Karlsruhe Institute of Technology

Non-parametric Methods for Correlation Analysis in Multivariate Data with Applications in Data Mining

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Hoang Vu Nguyen

aus Ho Chi Minh Stadt, Vietnam

Tag der mündlichen Prüfung:	06. Februar 2015
Erster Gutachter:	Prof. Dr.-Ing. Klemens Böhm
Zweite Gutachterin:	Prof. Dr. Katharina Morik
Dritter Gutachter:	Prof. Dr. rer. nat. Emmanuel Müller

DOI: 10.5445/IR/1000049548

I declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, 24th November 2014

.....

(**Hoang Vu Nguyen**)

Acknowledgments

I would like to express my deep appreciation to Prof. Klemens Böhm, my supervisor and the head of my chair. Prof. Böhm has given me numerous opportunities to pursue independent and practical research. He has also provided me timely feedback and invaluable advices that are very helpful for me in meeting my agenda. His professionalism and attentiveness are key elements leading to the completion of this thesis.

I would also like to express my deep thanks to Prof. Katharina Morik, my second reviewer. Prof. Morik not only agreed to act as a referee for my thesis but also gave me lots of useful comments to improve my thesis, my presentation skills, and especially my knowledge on graphical modeling and data mining.

I want to extend my thanks to Emmanuel Müller and Jilles Vreeken, two senior researchers who have helped me a lot in keeping up with latest developments in the literature. Their expertise and experience on scientific research are invaluable inputs for shaping the early directions of my work.

My sincere gratitude also goes to my colleagues, the secretaries, and the administrative staffs at my chair. In one or another way, they have helped me to move on with my research.

I would like to thank the students who I got the chance to work with. Their innovation and creativity have given me new insights in problem solving.

I am also extremely grateful to my friends who I either have known for a long time or I have got the chance to meet up during conference trips. Talking to them socially or scientifically is always a great joy to me.

Last but never least, I want to especially thank my wife and my son for their encouragement, love, and support in the entire course of this thesis. Carrying a doctoral study, I had to face lots of challenges, externally and internally. Without them, it would have been very tough for me to overcome the challenges and to keep things in perspective, especially during the hard time.

Abstract

Nowadays, many real-world applications are producing data that keeps increasing in both quantity and scale, making it impossible for human to manually analyze and understand such data. Thus, automatic and scalable solutions for extracting interesting and useful knowledge hidden in the data are required. Motivated by this need, data mining has been proposed as a general solution.

Traditional data mining techniques focus on the full space which consists of all given dimensions. However, due to the presence of noisy and/or irrelevant dimensions, mining the full space cannot lead to the discovery of interesting patterns. Thus, subspace mining has been introduced as an alternative paradigm to address the problem of high dimensionality. In short, subspace mining looks for novel patterns in multiple (possibly overlapping) subspaces. Recent development in this area has pointed out that knowledge discovery in subspaces whose dimensions are correlated will yield patterns that are of high quality and are easy to interpret. Hence, for an effective knowledge discovery in subspaces, it is important to first discover correlated subsets of dimensions, i.e., correlated subspaces. To achieve this, we need to have reliable methods to assess the correlation of any subset of dimensions. This in turn is the general goal of correlation analysis.

The notion of correlation is one of the key elements of modern statistics and has a wide impact in many areas of applied science. In short, correlation analysis studies the statistical relationships of two or more dimensions. Considering the fact that in today's real-world applications, data is collected in multivariate spaces having hundreds of dimensions, enabling efficient and effective correlation analysis for subspace mining enriches both of their applicability not only in data mining but also in any area dealing with high dimensional data.

In this thesis, we develop novel methods for correlation analysis in multivariate data, with a special focus on mining correlated subspaces. These methods handle the major open challenges arisen when combining correlation analysis with subspace mining. In particular, we propose novel techniques with new correlation measures for scalable mining of correlated subspaces in real-valued data as well as in relational databases with mixed data types. In addition to introducing new correlation measures, we propose methods for computing total correlation—a well-known multivariate measure. As total correlation has a widespread impact in many fields, our work opens various potential venues of applications. We also extend our research beyond traditional correlation analysis: We explore interaction-preserving discretization of multivariate data and multivariate causality analysis. Besides theoretical findings, we conduct thorough experiments on a variety of real-world data sets. The results validate the benefits of our methods.

Zusammenfassung

Heutzutage produzieren viele reale Anwendungen Daten, die sowohl in Quantität als auch in Umfang zunehmen, was es unmöglich für Menschen macht, die Daten manuell zu analysieren und zu verstehen. Deswegen werden automatische und skalierbare Lösungen für das Extrahieren vom interessanten und nützlichen Wissen, die in den Daten verborgen sind, erforderlich. Dieses Bedürfnis motiviert, Data-Mining als allgemeine Lösung vorzuschlagen.

Traditionelle Data-Mining-Techniken konzentrieren sich auf die Fullspace, die von allen angegebenen Dimensionen besteht. Allerdings, aufgrund von verrauschten und/oder irrelevanten Dimensionen führt Fullspace-Mining zur Entdeckung von interessanten Mustern nicht. Deswegen wurde Subspace-Mining als alternatives Paradigma eingefürht, um das Problem der hohen Dimensionalität zu behandeln. Kurz gesagt, sucht Subspace-Mining nach neuen Mustern in mehreren (möglicherweise überlappenden) Subspaces. Die jüngste Entwicklung in diesem Bereich hat gezeigt, dass Knowledge-Discovery in Subspaces, deren Dimensionen korreliert werden, Muster ergeben, die von hoher Qualität und leicht zu interpretieren sind. Daher ist es wichtig für ein effektives Knowledge-Discovery in Subspaces, zuerst korrelierte Untermengen von Dimensionen zu entdecken, d.h. korrelierte Subspaces. Um dies zu erreichen, müssen wir zuverlässige Methoden, um die Korrelation jeder Untermenge von Dimensionen bewerten zu können. Dies wiederum ist das allgemeine Ziel der Korrelationsanalyse.

Der Begriff der Korrelation ist eines der Schlüsselelemente der modernen Statistik und hat breite Auswirkungen in vielen Bereichen der angewandten Wissenschaft. Zusammengefasst, untersucht die Korrelationsanalyse die statistischen Beziehungen von zwei oder mehreren Dimensionen. Weil in den heutigen realen Anwendungen werden Daten in multivariaten Spaces mit Hunderten von Dimensionen gesammelt, das Ermöglichen einer effizienten und effektiven Korrelationsanalyse für Subspace-Mining steigt die Anwendbarkeit des Data-Minings und jedes Bereiches, der sich mit hoher dimensionaler Daten beschäftigt.

In dieser Thesis entwickeln wir neue Methoden zur Korrelationsanalyse in multivariaten Daten mit einem speziellen Fokus auf das Mining von korrelierten Subspaces. Diese Methoden behandeln die großen offenen Herausforderungen der Kombination von Korrelationsanalyse mit Subspace-Mining. Im Besonderen schlagen wir neuartige Techniken mit neuen Korrelationsmaßen für skalierbares Mining von korrelierten Subspaces in kontinuierlichen Daten, sowie in relationalen Datenbanken mit gemischten Datentypen, vor. Neben der Einführung neuer Korrelationsmaßnahmen schlagen wir Methoden zur Berechnung der Total-Korrelation vor—eine bekannte multivariate Maßnahme. Weil die Total-Korrelation weitverbreitete Auswirkungen in vielen Bereichen hat, öffnet unsere Arbeit verschiedene mögliche

Anwendungen. Wir erweitern auch unsere Forschung über die traditionelle Korrelationsanalyse: Wir erforschen die Interaction-Preserving Diskretisierung der multivariaten Datenanalyse und der multivariaten Kausalität. Neben theoretischen Erkenntnissen führen wir flächendeckende Experimente auf einer Vielzahl von realen Datensätzen aus. Die Ergebnisse bestätigen die Vorteile unserer Methoden.

Contents

List of Figures

List of Tables

1. Overview of Thesis

1.1. Introduction

Nowadays, many real-world applications are producing data that keeps increasing in both quantity and scale. This tremendous growth has made it increasingly more and more difficult for human to manually analyze and understand the data. Therefore, automatic and scalable solutions for extracting interesting and useful knowledge hidden in the data are required. The knowledge obtained in turn is important for users in multiple aspects, e.g., decision making and planning. Motivated by such a need, Knowledge Discovery in Databases (KDD) has been proposed and quickly become a promising approach towards tackling the challenge of data analysis. Roughly speaking, the KDD process takes in raw data, pre-processes it, extracts novel patterns from it, evaluates these patterns for their impact and validity, and finally presents the knowledge learned out of the most important patterns to users [HK01]. The step of identifying novel patterns, known as "data mining", is the key component of the KDD process.

Traditionally, data mining techniques focus on the full space which consists of all given dimensions. However, this space often obscures interesting patterns due to the presence of noisy and/or irrelevant dimensions [BGRS99]. Further, taking into account the full space with very many dimensions often causes the curse of dimensionality [LV07], especially when the number of data objects more than often is insufficient to fill up the full space. At first, global dimensionality reduction is proposed to alleviate the issue. Techniques of this type, e.g., Principle Component Analysis (PCA), aim at finding a global projection that best preserves some characteristic of the data; in the case of PCA, it is the variance that is preserved. However, by focusing on a single view, they miss local structure, e.g., locally clustered objects, embedded in different subsets of dimensions (a.k.a. subspaces). Thus, subspace mining has been introduced as an alternative paradigm to address the problem of high dimensionality. In short, subspace mining looks for multiple (possibly overlapping) subspaces where novel patterns are likely present.

For example, subspace clustering [AGGR98, SZ04, MAG+09, AKMS07, GFM+11] detects subspaces that likely exhibit clustering structure while subspace outlier detection [AY01, KSZK09, MSS11] uncovers subspaces where one or more objects exhibit some deviation from the remaining objects. However, most existing subspace mining methods tie the subspace mining process to a specific type of pattern, e.g., clusters or outliers. Hence, they produce subspaces that do not provide any guarantee on the detectability of other types of pattern, i.e., their subspaces lack generality. Recognizing this problem, recent studies [CFZ99, ZPWN08, CYZR10, KMB12] advocate for decoupling the process of subspace discovery and the mining of interesting patterns in these subspaces. Their proposals are based on the observation that knowledge discovery in subspaces where member dimensions are correlated will result in novel patterns that cannot be obtained by looking at each individual dimension. Furthermore, the results are often easier to interpret by, e.g., checking the characteristics of the dimensions involved with domain experts. Hence, for an effective knowledge discovery in subspaces, it is crucial to first discover correlated subspaces.

Example 1. *The facility management of KIT stores indicator values of buildings, such as electricity, heating, gas, and water consumption per time unit. Each dimension is one indicator of a specific building. In such data, not all indicators of all buildings are correlated with each other. Instead, there are different subsets of correlated indicators, e.g., the heating indicators of office buildings, the ones of the Chemistry department, and so on. Overlap among subsets is possible since buildings can both be office buildings and belong to the Chemistry department. In practice, detecting subsets of correlated indicators is important for facility managers. This is because they can understand the energy consumption of the university better from such subsets. For instance, they can apply specialized data analytics on just those subsets to find anomalous measurements. An example would be an abnormally high heating value among the office buildings. Clearly, one cannot observe such patterns when indicators are not correlated or data is distributed randomly. Further, the subsets can be exploited to construct prediction models to monitor energy consumption, and hence, save the management from investing in new energy smart meters.*

To mine correlated subspaces, in turn we need to have reliable methods to assess the correlation of any subspace, or more in particular, we need techniques to assess the correlation of any subset of dimensions. In fact, this is the general goal of correlation analysis.

For years, the notion of correlation has become one of the key elements of modern statistics. It also plays an important role in many areas of applied science, e.g., databases [IMH+04, ZHO+11], data mining [CFZ99, ZPWN08, CYZR10, KMB12], machine learning [AL13, AABL13, JHS10], signal processing [SRPP11, RSX+11], biology [CYZR10, RRF+11], to name a few. In a nutshell, correlation analysis is concerned with studying the statistical relationship of two (then the analysis is pairwise) or more (then the analysis is multivariate) dimensions [Ren59]. As such dependencies are reliable indicators of the existence of interesting patterns, correlation analysis can help to steer the focus of users to appropriate subspaces, and

hence, reduce their effort in dealing with a huge number of subspaces; most of which are irrelevant. This is an important requirement in the era of big data. Considering the fact that nowadays in several real-world applications, data is collected in increasingly multivariate spaces having hundreds of dimensions, enabling efficient and effective correlation analysis for subspace mining enriches both of their applications not only in data mining but also in various other areas having something to do with high dimensional data.

With such a motivation, in this thesis, we aim at developing novel methods for correlation analysis in multivariate data, with a special focus on mining correlated subspaces. In the following, we point out major open challenges that are in the way of achieving our goal. After that, we specify the scope of this thesis as well as its contributions.

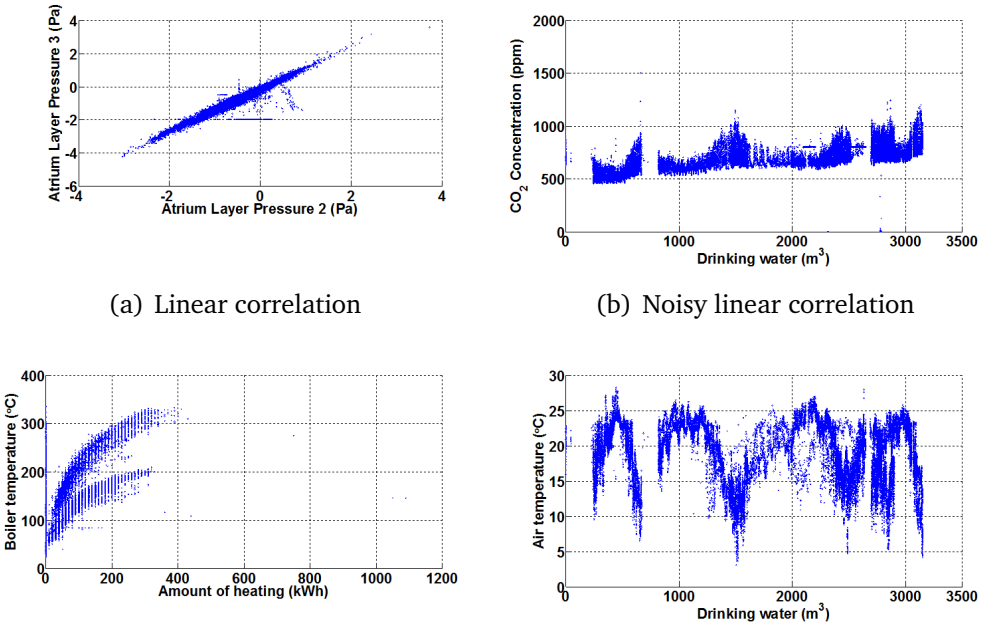
1.2. Challenges

There are many challenges associated with the general research of correlation analysis. As we mainly address the issue of mining correlated subspaces, in this section, we discuss five major issues specifically relevant for this task, namely: complex types of correlation, computability, efficiency, mixed data types, and redundancy of output subspaces. Chapters 3 to 7 focus on the development of effective techniques to tackle each of these challenges.

Challenge 1: Complex Types of Correlation

In general, real-world data contains different types of correlation: pairwise vs. multivariate and linear vs. non-linear. A correlation is pairwise if it is involved in two dimensions; otherwise, it is multivariate. Regarding linear correlations, the scatter plot of two dimensions (pairwise) that are linearly correlated features a straight line; see Figure 1.1(a) for an example. In general, linear correlations without noise are not common. Instead, data tends to be noisy; see Figure 1.1(b) for an example. On the other hand, non-linear correlations can be either functional (Figure 1.1(c)) or non-functional (Figure 1.1(d)). Again, functional and non-functional correlations can also be noisy.

In the setting of data mining, we often do not have much knowledge about the data at hand, including which correlations it contains. Thus, to ensure the generality of the mining process, the underlying correlation analysis method should be able to detect generic (i.e., linear as well as non-linear, functional as well as non-functional) correlations. Further, the correlations should not be restricted to pairwise only; they can be involved in any arbitrary number of dimensions. Unraveling multivariate generic correlations is particularly very challenging. In fact, there have been correlation measures proposed for such a purpose, such as those based on Shannon entropy [CT06]. However, as we explain in the next challenge, it is not always straightforward to reliably compute them on real-valued data.



(a) Linear correlation

(b) Noisy linear correlation

(c) Functional non-linear correlation

(d) Non-functional non-linear correlation

Figure 1.1.: Examples of different types of correlation.

Challenge 2: Computability

Traditional statistics usually assume that distributions of data are available and define correlation measures based on these distributions. This especially is the case for measures capturing complex types of correlation, e.g., the measures defined based on Shannon entropy (mutual information and total correlation). While such distributions can be easily obtained for discrete/categorical data, this in general is not the case for real-valued data. Thus, if we want to apply existing correlation measures on real-valued data, we usually need to first obtain data distributions. This could be done by choosing a distribution in advance and fit it to the data at hand, or by means of kernel methods [Sil86]. However, these workarounds not only cause biases to the choice of the distribution/kernel but also flag in power with the growing number of dimensions, i.e., the curse of dimensionality [LV07]. On the other hand, there are non-parametric approaches to computing correlation measures, such as discretization [CFZ99] and estimation based on nearest neighbor analysis [KSG04]. Yet, these non-parametric methods also suffer from some certain problems which will be explained in Section 1.3. Thus, in this thesis, we propose new correlation measures together with their robust non-parametric computation for an effective mining of data containing real-valued dimensions.

Challenge 3: Efficiency

The search space of potentially overlapping correlated subspaces, called the subspace lattice (see Figure 1.2), is exponential in the number of dimensions. For a data set with 40 dimensions, the total number of subspaces is 2^{40} (more than 1

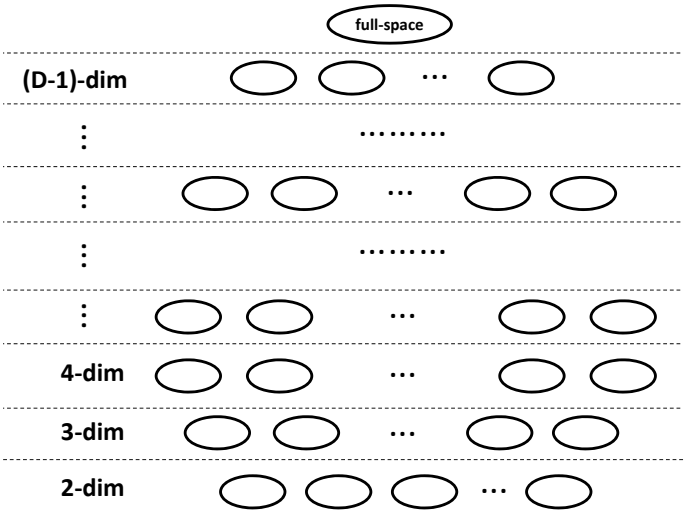


Figure 1.2.: Example showing the subspace lattice of multivariate data.

trillion). Considering the fact that real-world data usually contains a larger number of dimensions, the subspace lattice in practice is astronomically large. Setting aside the complexity of computing correlation measures, this huge search space presents a tremendous challenge in terms of computational cost. As we will point out later, state of the art techniques for subspace search mostly do not scale well due to inefficient search schemes. Further, in Chapter 4, we show that under an enhanced setting, the subspace search problem is NP-hard. Thus, it is very challenging how to perform multivariate correlation analysis in a scalable manner.

Challenge 4: Mixed Data Types

Real-world data may contain dimensions of heterogeneous data types (i.e., real-valued, discrete, and categorical) while traditional correlation analysis usually assumes that dimensions have the same data type. As a result, performing correlation analysis in data with mixed data types demands not only novel correlation measures but also novel methods to explore the subspace lattice. Solving this issue will enable us to include all possible dimensions, representing different sources of information, into the mining process, regardless of their data types. As opposed to limiting to a single data type during the process of correlation analysis, by considering heterogeneous data types, we are able to achieve a more comprehensive picture on the data.

Challenge 5: Redundancy of Output Subspaces

As the number of candidate subspaces is huge, the process of mining correlated subspaces may produce many subspaces which essentially convey similar information regarding the underlying data. This not only hinders manual inspection and post-analysis but also reduces the value of the results: Instead of benefiting from the output, users must spend additional (significant) effort to find out which subspace is valuable and which one is not. In other words, they are again faced with

the daunting task of extracting useful knowledge from a large amount of data, in this case, the large amount of output subspaces. Thus, it is of great importance that the number of output subspaces is reasonably small while still capturing the essential knowledge. This issue so far has not been addressed in the literature.

1.3. Related Work

In the following, we briefly review related work and explain why none of them can tackle all of the aforementioned challenges. To ease presentation, we first discuss well-known correlation measures, which form the core of correlation analysis. Then, we review methods that are capable of mining subspaces, namely feature selection, general subspace search, and subspace search for specific tasks. The details are as follows.

Correlation measures. Pearson's coefficient is a popular correlation measure widely used in data analysis, thanks to its easy computation on empirical data, i.e., it addresses Challenge 2. However, since it is only for two dimensions (pairwise) and mainly captures linear correlations, it does not address Challenge 1. Apart from Pearson's coefficient, there exist also other more advanced pairwise correlation measures, e.g., Spearman's rank coefficient [Spe87], Kendall's τ coefficient [Ken38], distance correlation [SR09]. However, it is often hard to generalize them for the multivariate setting [RRF+11]. Thus, they also do not address Challenge 1.

Correlation measures defined based on Shannon entropy in turn are able to capture generic correlations [CT06]. Arguably, two most well-known measures of this kind are mutual information (pairwise) and total correlation (multivariate). In fact, total correlation is a generalization of mutual information to the multivariate setting [Han78]. To compute both measures non-parametrically on real-valued data, we need to estimate data distribution by, e.g., discretization (converting real-valued data to discrete) or performing nearest neighbor analysis. Regarding the former, a common practice is to use equal-width or equal-frequency binning. However, such naïve discretization methods suffer from hard parameter setting, for instance, we do not know in general how many bins we should use and where we should place the bins. Further, they are often oblivious of data distribution, and hence, tend not to preserve correlations well. Recently, Reshef et al. [RRF+11] propose a method to find optimal discretization to compute mutual information on real-valued data. Yet, as we point out in Chapter 6, their technique suffers from some drawbacks on the pairwise case as well as cannot be generalized for total correlation. Other advanced discretization techniques have also been proposed, for instance, [Bay01, MPY05, KM07]. In Chapter 8 (in particular Section 8.3), we will explain in more details the weaknesses of existing discretization techniques. Overall, addressing the computability of mutual information and total correlation w.r.t. discretization is still an open issue, i.e., existing computation methods using discretization do not meet Challenge 2. Besides discretization, one could also estimate both mutual information and total correlation by nearest neighbor analysis [KSG04]. This approach can avoid discretizing the data. However, it requires

us to choose the number of nearest neighbors—a hard-to-set parameter as well. In addition, it relies on metric distances. The higher the dimensionality, the less reliable the distances are [BGRS99]. Therefore, the robustness of this approach tends to reduce in multivariate spaces, and hence, Challenge 1 might not be met. While there is no clear winner between discretization and nearest neighbor analysis, in this thesis, part of its contributions is to compute total correlation by discretization. The study of nearest neighbor analysis is left for future work.

Another class of correlation measures is the quadratic measures of dependency [AL97, Ach08, SGSS07]. Most of them, however, require kernel density estimation in their computation [RSX+11], i.e., they do not address Challenge 2. An exception to this is the measure recently proposed in [SRPP11]. Yet, it is pairwise in its current form. We will extend this measure in Chapter 4 and explain how to efficiently and reliably compute it on empirical data.

Please note that the measures aforementioned could be computed easily when the data distribution is known, or such an assumption is made. Nevertheless, such a parametric approach is not the focus of this thesis, and hence, is not further explored.

Regarding mixed data types (i.e., real-valued, discrete, and categorical), generally it is not straightforward to apply existing measures for this setting, i.e., many of them do not address Challenge 4. Exceptions to this are mutual information and total correlation. In particular, one could discretize real-valued data, and then, process discrete/categorical data and the original real-valued alike. However, as discussed above, finding a reliable discretization still is an open issue. On the other hand, one could tackle heterogeneous data types by combining measures specialized for each type. The subtlety here is to utilize measures that ensure no big "gap" between each world, e.g., to facilitate interpretability. Nevertheless, to the best of our knowledge, there currently exists no such solution. Thus, we address this issue in Chapter 5.

Feature selection. In a nutshell, feature selection (as well as feature transformation) aims at selecting a subset of dimensions (a.k.a. features, attributes) that are most suitable for the mining/learning task in consideration. Besides, feature selection is also a way to alleviate the curse of dimensionality. In the following, we review some feature selection methods and point out why they are not suited for our goal.

Principle Component Analysis (PCA) [LV07] transforms the data to a new space that best preserves its variance. This transformation however retains linear correlation only, and hence, does not satisfy Challenge 1.

Supervised feature selection [Hal00, PLD05, SSG+07, GN09, RLHEC10, SM11] aims at choosing a subset of dimensions that are correlated to the class label but otherwise not correlated to each other. Thus, its goal is different from ours.

Unsupervised feature selection [DCSL02, DB04, HCN05, GDJ11, ZL07] selects the smallest subset of features that best uncovers interesting patterns (e.g., clusters) from data. This approach tends to be biased w.r.t. to the chosen notion of

interesting pattern. This in turn could limit the types of correlation it can uncover [CFZ99, KMB12] (cf., Challenge 1).

Overall, a common major drawback of supervised/unsupervised feature selection techniques is that by mining a single subspace, they potentially miss other interesting correlated subspaces that also containing useful knowledge.

General subspace search. Some recent proposals are [CFZ99, KKKW03, CYZR10, KMB12]. Methods of this kind are capable of mining several overlapping correlated subspaces, abstracting from any concrete task. Thus, they can detect different types of correlation, i.e., they can address Challenge 1. The notions of correlation used in these techniques vary. For instance, ENCLUS [CFZ99] employs total correlation and computes it using equal-width discretization (a naïve discretization technique). HiCS [KMB12] on the other hand proposes a novel correlation measure quantifying the difference between marginal and the respective conditional distributions. In fact, our measures proposed in Chapter 3 are based on the measure of HiCS. That is, we also perform correlation analysis by examining the divergence between marginal and condition distributions. However, our measures in Chapter 3 are different from that of HiCS regarding (a) the conditional distributions to use, (b) the way the divergence is quantified, and (c) robustness to the curse of dimensionality. More details are provided in the respective chapter.

The general subspace search techniques mentioned above all explore the search space using a bottom-up search scheme, which is also known as the Apriori search. This search scheme imposes a monotonicity property, reminiscent of that used in itemset mining [AS94], to substantially reduce the search space. HiCS introduces a version of this scheme which eases parameterization. In Chapter 3, we adopt this version for mining correlated subspaces.

Nonetheless, as we will explain in Chapter 4, due to the monotonicity restriction, this search scheme tends to detect only low dimensional subspaces. Such subspaces in turn likely are different projections of the same high dimensional correlated subspaces. This causes redundancy that is well-known for most subspace mining models [MGAS09, MAG+09], i.e., Challenge 5 is not met. Besides, this search scheme also suffers from the scalability issue due to its expensive mining of correlated dimension pairs, and the levelwise search scheme which generates very many candidate subspaces, i.e., it does not address Challenge 3. As a major contribution of this thesis, we propose a highly scalable search scheme with post-removal of redundancy in Chapter 4 to tackle both Challenges 3 and 5. One last remark is that existing general subspace search techniques also do not meet Challenge 4 due to the lack of a correlation measure suitable for mixed data types (as explained above). We will study this issue in details in Chapter 5.

There is also work in the database area which can be adapted to mine subspaces [ZHO+11, IMH+04, SBHR06, AHS+09, PLP+10]. Yet, they are limited to disjoint partitioning of dimensions [ZHO+11, AHS+09] (i.e., overlapping of subspaces is not allowed), foreign key and primary key constraints [SBHR06], or pairwise correlation analysis [IMH+04, PLP+10]. We will get back to these meth-

ods in Chapter 5 where we propose a method for mining multivariate (possibly overlapping) correlated subspaces in relational databases.

Subspace clustering and outlier detection. Besides general subspace search methods, there are also subspace methods designed specifically for well-known mining tasks such as outlier detection [AY01, KKSZ12, MSS11, DAN+14, MNDA13] and clustering [AGGR98, NGC01, YM05, KKK04, KKRW05, AKMS07, AKMS08b, AKMS08a, MAK+09, MAG+09, MAGS11, DB14, HFK+14, GHP+14]. However, they are strongly coupled with a specific notion of outlier or cluster. As a result, they tend to have little effect on other tasks, e.g., detecting different types of pattern exhibiting different types of correlation. Thus, they in general do not address Challenge 1.

In addition, some of the methods mentioned above [DAN+14, MNDA13, GHP+14] detect one to two subspaces only, and hence, miss other important subspaces. Most of the remaining methods [AY01, MSS11, AGGR98, NGC01, YM05, KKK04, AKMS07, AKMS08b, AKMS08a] in turn can detect several subspaces. They however utilize the Apriori search scheme and have the scalability problem (cf., Challenge 3). In contrast, [KKRW05, MAK+09, MAG+09, MAGS11] perform jump search and achieve high efficiency. Our jump search in Chapter 4 in fact is inspired from these methods. But please note that their jump search mines multivariate regions of high density (e.g., with respect to the cluster notion of DBSCAN as in [MAGS11]). Our goal on the other hand is to mine correlated subspaces not bound to any specific local pattern, i.e., we perform global processing while [KKRW05, MAK+09, MAG+09, MAGS11] perform local processing.

As a final remark, most of the methods looking for patterns in several subspaces tend to produce redundant output (cf., Challenge 5). This issue is handled in [AKMS08b, MAG+09] which are able to mine non-redundant clusters. Their handling of redundancy is in-process. Our solution in turn is post-process. Studying in-process removal of redundancy is reserved for future work.

Correlation clustering. Methods of this category [ABK+07a, ABK+07b, AY00, AR10, BKKZ04, YWWY02, GFVS12] perform a more direct treatment of correlations during the clustering process. In particular, they aim to look for clusters whose member objects exhibit a high correlation. Roughly speaking, their focus is on local correlation while we are interested in global correlation. Yet, till now these methods are limited to linear correlations, i.e., Challenge 1 is not addressed. Extending them to more complex correlations is hard since both of their notions of correlated cluster and their mining processes are coupled with linear correlations. We identify local correlation analysis capturing multivariate and generic correlations to be an interesting direction for future research. In this thesis, we however focus specifically on the global aspect of correlation.

1.4. Contributions and Thesis Outline

There are many possible ways to advance the field of correlation analysis. In this thesis, the main aim is to address correlation analysis in multivariate data with

a special focus on mining correlated subspaces. Further, we embark on a non-parametric approach and propose methods to tackle all of the aforementioned challenges. In the following, we present the main contributions and the structure and of this thesis. Overall, the thesis is divided into four parts.

Part 1: Background

In this part, the basic concepts that are relevant for understanding the theories behind our novel methods are introduced. In particular, the notion of correlation measure is formalized and two properties a correlation measure should satisfy for effective data mining is specified. Some well-known measures widely used in data analysis are also reviewed and formal explanation on why they are unsuited for addressing the challenges mentioned in Section 1.2 is provided.

Part 2: Mining Correlated Subspaces in Real-Valued Data

In this part, the focus is on mining correlated subspaces in real-valued data. In particular, we propose two methods for tackling Challenges 1–3 and 5.

In Chapter 3, we study multivariate correlation analysis with cumulative entropy (CE) [RCVW04, CL09], a new type of entropy designed specifically for real-valued data. Since it is not readily clear how cumulative entropy can be used for correlation analysis, we make an important contribution by introducing two new multivariate correlation measures, CMI (for *cumulative mutual information*) and CMI++, which are based on this type of entropy. In fact, our measures could be considered as extensions of the measure proposed in HiCS; yet, we will later point out their major differences. In a nutshell, our measures do not make any assumption on the type of correlation. They are thus capable of capturing both linear and non-linear, functional and non-functional correlations (cf., Challenge 1). Besides addressing the good properties a correlation measure should possess (see Chapter 2), we prove that CMI and CMI++ possess all important properties that total correlation [CT06] has on discrete/categorical data. Further, we introduce effective and efficient methods for reliably computing CMI and CMI++ on real-valued data, and hence, address Challenge 2. As the main focus in this chapter is on correlation measures, to mine correlated subspaces, we simply plug our measures into the Apriori search scheme of HiCS. We evaluate the quality of subspaces detected by both CMI and CMI++ through clustering and outlier detection. The experiments show that our methods bring about quality improvement for both data mining tasks.

In Chapter 4, we point out that the Apriori search scheme suffers from four issues: (a) it is not highly scalable, (b) it tends to miss high dimensional correlated subspaces, (c) it fragments them into many redundant lower dimensional subspaces, and (d) it is prone to the curse of dimensionality. From this analysis, we propose a new scalable search scheme using jump search. Instead of traversing the subspace lattice level by level, this search scheme jumps to the relevant high dimensional subspaces by analyzing the statistics of their two dimensional projections. Our jump search in fact is inspired from that of [MAK+09]; yet, as aforementioned the

latter searches for multivariate regions of high density while we go for multivariate correlated subspaces. While the jump search is efficient, it alone is insufficient. In particular, to ensure scalability we have to mine the two dimensional subspaces efficiently. Hence, we study a correlation measure which belongs to the class of quadratic measures of (in)dependence [RSX+11, SRPP11] and which allows for an efficient pairwise computation. Similarly to CMI and CMI++, this measure also is based on cumulative distribution functions (cdfs) and does not assume a specific type of correlation (cf., Challenge 1). In addition, its computation on empirical data is fully in closed form, e.g., it permits straightforward application in practice (cf., Challenge 2). This closed form also allows an efficient approximation based on AMS Sketch [AMS96]. Combining the jump search scheme and the efficient method for mining two dimensional subspaces, the technique covered in this chapter, named 4S for *scalable subspace search scheme*, achieves high scalability in mining correlated subspaces in very high dimensional data (cf., Challenge 3). Last but not least, we incorporate into 4S an MDL-based merge of subspaces which ensures succinctness of output as well as retrieves fragmented high dimensional correlated subspaces (cf., Challenge 5). The subspace merge step is done using the algorithm in [MV13]; yet, our transformation of subspace search to a problem where the technique in [MV13] becomes applicable is a novel contribution.

Part 3: Mining Correlated Subspaces in Mixed Typed Data

Though 4S achieves scalability, it is not applicable to data with mixed data types. Thus, the goal in this part is to address Challenge 4.

In Chapter 5, we do not only handle heterogeneous data types: We take into account the fact that data may be stored in multiple relations as well. Thus, this chapter, in a more precise way to put, is about detecting groups of correlated columns (dimensions) in relational databases. Here, our solution, named DECOREL for *Detecting Column Correlations*, takes advantage of the knowledge gained from the CMI measures (which means CMI and CMI++) and the scalable search scheme of 4S. Considering the correlation measure, since we are going to adapt (not fully) the search scheme of 4S, we only need to deal with pairwise correlations. Hence, we propose a pairwise information-theoretic correlation measure, which is a blend between the pairwise version of the CMI++ measure and mutual information. Recalling that both are information-theoretic measures, we hence achieve a handling of mixed data types with no major change in the interpretation of correlation scores. Further, we show that this close relation between the CMI++ measure and mutual information enables a cross-data-type correlation computation. Besides, we note that our correlation measure proposed in this chapter is related to the one employed in [PLP+10]; yet, as we will point out their measure uses Shannon entropy and does not consider real-valued data. Regarding the search scheme, we adopt the jump search of 4S. However, we change some details to better fit mixed data types and the multi-relational nature of the data.

Part 4: Computing Total Correlation

Total correlation is a well-known multivariate correlation measure and defined based on Shannon entropy. One of the most famous ways in data mining to compute total correlation is to first discretize the data to obtain probability mass functions. However, naïve discretizations usually break correlations among dimensions, causing inaccurate computation of total correlation. Thus, the objective of this part is to develop a correlation-aware discretization technique to compute total correlation (cf., Challenge 2). By correlation-aware, we mean that the technique should preserve correlations in the data with respect to total correlation. Our solutions in this part in fact are inspired from the work of Reshef et al. [RRF+11], which finds optimal discretization for mutual information. We will show that their approach however is not for computing total correlation. Hence, new methods such as ours are required.

In Chapter 6, we propose such our solution for real-valued data. Yet, our contributions here are beyond merely computing total correlation. Instead, we propose multivariate maximal correlation analysis, which we generalize from various existing methods for correlation analysis. It serves as the main theme of the entire chapter. Our method for computing total correlation on empirical data, named MAC for ***M**ultivariate M**a**ximal **C**orrelation Analysis*, is an instantiation of this general notion.

In Chapter 7, we extend MAC to incorporate external information (e.g., class label) for the supervised setting. Further, we extend MAC to handle mixed typed data (cf., Challenge 4). These two extensions amend the applicability and potential impact of MAC in practice.

Part 5: Going beyond Correlation Analysis

In the previous four parts, we devise novel methods for scalable mining of correlated subspaces and for computing total correlation. This part slightly departs from the spirit of correlation analysis: We aim at exploring possible extensions of our research to other related venues.

In Chapter 8, we further our study on correlation-aware discretization. However, our new method, named IPD for *Interaction-Preserving Discretization*, is not restricted to any specific notion of correlation. Instead, we aim at preserving more general interactions among dimensions by analyzing their multivariate distributions in consecutive data regions. In fact, IPD belongs to and is inspired from the class of multivariate discretization techniques [Bay01, MPY05]. However, to our knowledge, we are first to propose an objective function for this task. Our objective function successfully balances between preserving interactions of dimensions and the detail of the dimension under discretization. We introduce two efficient algorithms for solving the objective function; one is optimal, the other is a 2-approximation of the optimal. Last but not least, we propose a novel distance function for assessing the difference between two multivariate distributions. IPD is a combination of our objective function and our distance measure. Extensive

experiments on both synthetic and real-world data demonstrate the superiority of IPD compared to state of the art methods.

In Chapter 9, we extend our research to causality analysis which goes beyond telling if two (groups of) dimensions are correlated. Instead, it tells us, under the assumption of causal sufficiency (no hidden confounders), which (group) of the two causes the other. In particular, given two multivariate random variables X and Y with some correlation relationship and with the same number of observations, we aim at efficiently inferring their causal direction. We accomplish this by proposing a new principle for causal inference that is based on Kolmogorov complexity and which makes use of the algorithmic Markov condition. We show that our principle generalizes various existing methods that rely on the principle of plausible Markov kernels. As Kolmogorov complexity is not computable, we present NOVUM (for *entropy divergence-based causal inference on multivariate and mixed typed data*), an efficient non-parametric implementation of our principle based on cumulative and Shannon entropy. As it will turn out, the computation of NOVUM is based on that of CMI++. Further, we show that our method is applicable to mixed typed data, as well as how to derive a scalable causal discovery framework. One important feature of our work is that we do not restrict the type of correlation between random variables, be it linear or non-linear, functional or non-functional. Extensive experiments on both synthetic and real-world data show NOVUM to yield both high accuracy and high efficiency on both deterministic and noisy data.

Part 6: Summary

In this part, we summarize all contributions of this thesis and discuss directions for future research.

Part I.

Background

2. Principles of Correlation Measures

In this chapter, we formalize the notion of correlation measure. We further show how some well-known measures instantiate this notion. Overall, the materials presented in the following are useful for understanding the the theories behind all methods proposed in this thesis.

To formalize the principle of correlation measures, consider d real-valued random variables $X_1, \ldots, X_d$. Let $p(X_i)$ be the probability density function (pdf) of X_i. Also, let $p(x_i)$ be a short form for $p(X_i = x_i)$. Let $P(X_i)$ stand for the cumulative distribution function (cdf) of X_i, and $P(x_i)$ be a short form for $P(X_i \leq x_i)$. To understand what constitutes correlation, we first define the condition under which the dimensions are mutually/statistically independent.

Definition 1. ***Statistical Independence**:*
$X_1, \ldots, X_d$ are statistically independent iff

$$p(X_1, \ldots, X_d) = \prod_{i=1}^{d} p(X_i) \quad .$$

Hence, in principle the correlation score of $X_1, \ldots, X_d$, denoted as $Corr(X_1, \ldots, X_d)$, quantifies to which extent their relationship deviates from the statistical independence condition, i.e., to which extent their joint probability distribution differs from the product of their marginal probability distributions. The larger the difference, the higher $Corr(S)$ is. Thus, we have:

$$Corr(X_1, \ldots, X_d) \sim diff\left(p(X_1, \ldots, X_d), \prod_{i=1}^{d} p(X_i)\right)$$

with $diff$ being an instantiation of a difference function. In general, $Corr$ is expected to possess the following two properties [Ren59]:

- **Property 1** (Positive score): $Corr(X_1, \ldots, X_d) > 0$ iff $p(X_1, \ldots, X_d) \neq \prod_{i=1}^{d} p(X_i)$.
- **Property 2** (Zero score): $Corr(X_1, \ldots, X_d) = 0$ iff $p(X_1, \ldots, X_d) = \prod_{i=1}^{d} p(X_i)$, i.e., $X_1, \ldots, X_d$ are statistically independent.

These two properties are to ensure the correctness of $Corr$ from the theoretical aspect. In addition, $Corr$ must address Challenges 1 and 2 to ensure its applicability in practical data mining where, in general, neither prior knowledge on correlation types nor pdfs are known; only the data is available.

In the following, for illustration purposes, we present some well-known correlation measures instantiating the above general notion of correlation measure. The first measure is the Pearson's coefficient.

Definition 2. ***Pearson's coefficient****:*
The Pearson's coefficient of two variables X_i and X_j, denoted by $\rho(X_i, X_j)$, is

$$\rho(X_i, X_j) = \frac{E(X_i X_j) - E(X_i)E(X_j)}{\sigma_{X_i}\sigma_{X_j}} \quad .$$

In the above equation, $E(X_i X_j)$ and $E(X_i)E(X_j)$ can be considered to represent $p(X_i, X_j)$ and $p(X_i)p(X_j)$, respectively. In this sense, $\rho(X_i, X_j)$ is the normalized difference between $p(X_i, X_j)$ and $p(X_i)p(X_j)$.

Pearson's coefficient is popular as it permits closed form computation on empirical data, i.e., it solves Challenge 2. However, it suffers from three drawbacks:

- That $\rho(X_i, X_j) = 0$ does not imply X_i and X_j are statistically independent, i.e., it meets neither Property 1 nor Property 2.
- It is only defined for pairwise correlation.
- It is only effective in capturing linear correlation [BF85], i.e., it does not address Challenge 1.

Mutual information [CT06] also is a pairwise measure. However, it is based on Shannon (differential) entropy and capable of capturing generic correlations. The formal definition of mutual information is as follows.

Definition 3. ***Mutual Information****:*
The mutual information of two random variables X_i and X_j is

$$I(X_i, X_j) = H(X_i) + H(X_j) - H(X_i, X_j)$$

where $H(.)$ is the Shannon entropy.

Here, $H(X_i)+H(X_j)$ represents $p(X_i)p(X_j)$ and $H(X_i, X_j)$ represents $p(X_i, X_j)$. In fact, $I(X_i, X_j)$ is the Kullback-Leibler (KL) divergence of $p(X_i, X_j)$ and $p(X_i)p(X_j)$:

$$I(X_i, X_j) = KL(p(X_i, X_j) \mid\mid p(X_i)p(X_j)) = \int p(x_i, x_j) \log \frac{p(x_i, x_j)}{p(x_i)p(x_j)} dx_i dx_j \quad .$$

This property will be used in Chapters 4 and 5 to explain the rationale of our methods. Further, mutual information possesses the following properties.

Lemma 1. $I(X_i, X_j) = H(X_i) - H(X_i \mid X_j) = H(X_j) - H(X_j \mid X_i)$.

Lemma 2. $I(X_i, X_j) \geq 0$ *with equality iff* X_i *and* X_j *are statistically independent.*

Lemma 2 implies that mutual information meets both Properties 1 and 2. Further, it can detect generic correlations [RRF+11]. However, mutual information is unsuited for real-valued columns in both theoretical and practical aspects. From a theoretical point of view, intuitively, the more X_i and X_j are correlated, the lower the conditional entropy terms $H(X_i \mid X_j)$ and $H(X_j \mid X_i)$ are, i.e., the higher their mutual information. Thus, a high mutual information score *often* indicates a correlation between X and Y. We use the word *often* because Shannon entropy, a constituent element of traditional mutual information, is only well defined for discrete/categorical data. Its continuous form, differential entropy, suffers from some unexpected properties, such as [RCVW04]:

- it can be negative, and
- that the differential entropy of X_i given X_j equals to zero does not imply that X_i is a function of X_j.

Thus, unlike the discrete/categorical case, a high mutual information score between real-valued X_i and X_j, though indicating that X_i and X_j are correlated, conveys less information on their actual correlation. In particular, it does not say if X_i is a function of X_j or vice versa.

From a practical point of view, to compute mutual information for the real-valued case, we need the pdfs, which usually are unavailable and need to be estimated [CT06], e.g., by discretization or by nearest neighbor analysis (the latter is not explored further in this thesis). Naïve discretization tends to produce overly simple or complex histograms due to the lack of knowledge on the number and width of histogram bins. More advanced discretization method for computing mutual information has been proposed [RRF+11]. However, it needs to fix either X_i or X_j to equal-frequency binning first before searching for the optimal discretization in the other dimension. Due to this issue, which will be discussed in more details in Chapter 6, [RRF+11] does not effectively address Challenge 2.

A well-known generalization of mutual information to more than two random variables is total correlation.

Definition 4. ***Total Correlation:***
The total correlation of $X_1, \ldots, X_d$ *is*

$$T(X_1, \ldots, X_d) = \sum_{i=2}^{d} H(X_i) - H(X_i \mid X_1, \ldots, X_{i-1}) \quad .$$

Similarly to mutual information, total correlation is the divergence of $p(X_1, \ldots, X_d)$ and $\prod_{i=1}^{d} p(X_i)$.

$$\begin{aligned} T(X_1, \ldots, X_d) &= KL(p(X_1, \ldots, X_d) \mid\mid \prod_{i=1}^{d} p(X_i)) \\ &= \int p(x_1, \ldots, x_d) \log \frac{p(x_1, \ldots, x_d)}{\prod_{i=1}^{d} p(x_i)} dx_1 \cdots dx_d \quad . \end{aligned}$$

Besides, we have:

Lemma 3. $T(X_1, \ldots, X_d) = \sum_{i=2}^{d} H(X_i) - H(X_i \mid X_1, \ldots, X_{i-1})$.

Lemma 4. $T(X_1, \ldots, X_d) \geq 0$ *with equality iff* $X_1, \ldots, X_d$ *are statistically independent.*

We will use the above properties to motivate/explain our methods in the later chapters. As with mutual information, total correlation also suffers from the issues of interpretability and computability on real-valued data. We will present our solutions w.r.t. discretization to handle these issues in Chapters 6 and 7.

Part II.

Mining Correlated Subspaces in Real-Valued Data

3. Subspace Mining with CE-based Correlation Measures

This chapter is based on our work originally published as [NMV+13]:

H. V. Nguyen, E. Müller, J. Vreeken, F. Keller, and K. Böhm, *CMI: An information-theoretic correlation measure for enhancing subspace cluster and outlier detection*, in SDM, 2013, pp. 198-206.

Here, the focus is on mining correlated subspaces on real-valued data and show their benefits for both clustering and outlier detection. In particular, we propose two novel correlation measures: CMI (for *cumulative mutual information*) and CMI++. In a nutshell, two measures are similar in that, for each subspace, its CMI score as well as its CMI++ score is defined based on cumulative entropy [RCVW04, CL09] and quantifies their mutual correlation accordingly. Our measures could be considered as extensions of the one in HiCS [KMB12]. However, as we will point out, our measures can better alleviate the curse of dimensionality. We also prove certain properties of both CMI and CMI++ which suggest that they are suitable for correlation analysis. In terms of their differences, CMI++ improves over CMI in three main aspects: (a) it provides an unbiased comparison of correlation scores of subspaces with different dimensionality, (b) it is not involved in the search of dimension permutation, and (c) it allows a more principled computation on empirical data. As the major focus here is to propose new correlation measures, to mine correlated subspaces, we model the search space as a lattice of subspaces and simply plug our measures into the Apriori search scheme in [KMB12] to explore this lattice. The details are as follows.

3.1. Basic Notions

Given a database $\mathbf{DB}$ of size N and dimensionality D. The full space of all dimensions is given by $F = \{X_1, \ldots, X_D\}$. Each dimension $i \in [1, D]$ is associated

with a random variable X_i that has a real-valued domain $dom(X_i) = \mathbb{R}$. We use the notion of density distribution $p(X_i)$ for the projected database on dimension i. Any non-empty subset $S \in \mathcal{P}(F)$ is called a subspace of $\mathbf{DB}$. The dimensionality of S is denoted as $dim(S)$. W.l.o.g., the d-dimensional subspace $\{X_1, \ldots, X_d\}$ is used as a representative for any d-dimensional subspace in our analysis.

As discussed in Chapter 2, a correlation measure $Corr$ quantifies as closely as possible the deviation of subspaces from the ones whose member dimensions are statistically independent. In particular, for a d-dimensional subspace S with dimensions $X_1, \ldots, X_d$, its correlation score depends on how much the difference between its joint distribution $p(X_1, \ldots, X_d)$ and the product of its marginal distributions $p(X_1) \cdots p(X_d)$ is:

$$Corr(S) \sim \mathit{diff}\left(p(X_1, \ldots, X_d), \prod_{i=1}^{d} p(X_i)\right) \quad . \tag{3.1}$$

As aforementioned, the higher the difference, the higher the correlation score of S. Correlation of one dimensional subspaces is undefined. Thus, we focus on two or higher dimensional subspaces.

3.2. Detailed Assessment of Subspace Search Methods

Looking at existing techniques, ENCLUS [CFZ99] instantiates the diff function by total correlation

$$\sum_{i=1}^{d} H(X_i) - H(X_1, \ldots, X_d)$$

where $X_1, \ldots, X_d$ are *discretized* versions of the original dimensions.

PODM [YLO09], like ENCLUS, also discretizes data to obtain probability mass functions. Different from ENCLUS, it instantiates diff as

$$\sum_{x_1 \in dom(X_1), \ldots, x_d \in dom(X_d)} \frac{1}{p(x_1, \ldots, x_d)}$$

where $p(x_1, \ldots, x_d) \neq 0$. In other words, PODM computes the sum of the inverse of the joint distribution $p(X_1, \ldots, X_d)$.

The instantiation of HiCS [KMB12] is done by averaging over multiple random runs of the form

$$\mathit{diff}\,(p(X_i), p(X_i \mid \{X_1, \ldots, X_d\} \setminus \{X_i\}))$$

where X_i is picked randomly. That is, HiCS computes the Monte Carlo sum of the difference between marginal distribution $p(X_i)$ and its conditional distribution $p(X_i \mid \{X_1, \ldots, X_d\} \setminus \{X_i\})$. The calculation of HiCS is not dependent on discretization, and hence, HiCS comes closer than both ENCLUS and PODM to addressing the problem of correlation analysis in the real-valued domain.

Yet, none of these techniques fulfills Properties 1 and 2, and Challenges 1 and 2. Considering Property 2, the measure of ENCLUS is unreliable because of the knowledge loss caused by naïve discretization. The use of joint probability mass function $p(x_1, \cdots, x_d)$ is also problematic. In particular, following the definition of Shannon entropy we have that

$$H(X_1, \ldots, X_d) = - \sum_{x_1 \in dom(X_1), \ldots, x_d \in dom(X_d)} p(x_1, \ldots, x_d) \log p(x_1, \ldots, x_d)$$

with $p(x_1, \ldots, x_d)$ measured by the relative number of points in the respective hypercube. For increasing d, most of the hypercubes are empty and the nonempty ones most likely contain only one data point each [AY01, LV07]. Taking into account that $\lim_{x \to 0} x \log x = 0$, we have

$$H(X_1, \ldots, X_d) \to - \sum_{i=1}^{N} \frac{1}{N} \log \frac{1}{N} = \log N \quad .$$

That is, the joint entropy $H(X_1, \ldots, X_d)$ approaches $\log N$, which is constant. Hence, when d is large enough and all X_i have similar distribution, e.g., uniformly dense, any d-dimensional subspaces S_1 and S_2 have very similar correlation scores: $Corr(S_1) \approx Corr(S_2)$. In other words, the measure of ENCLUS produces indifferent scores for high dimensional subspaces. In addition, since ENCLUS employs naïve discretization, it does not address Challenge 2.

PODM on the other hand fails to address both Properties 1 and 2 as well as Challenge 1 since its measure just relies on the joint probability, i.e., it does not measure correlation. Similarly to ENCLUS, it also does not meet Challenge 2. Further, since PODM relies on the joint probability $p(X_1, \ldots, X_d)$, so analogously to ENCLUS, it also suffers from the issue of indiscriminative correlation scores in high dimensional subspaces.

As for HiCS, the random choice of X_i might miss important information on correlation as some dimension might not be tested against the remaining ones. For instance, a zero correlation score assigned by HiCS does not imply statistically independence as there is no guarantee that all dimensions are assessed against the others at least once. Thus, HiCS principally does not address Properties 1 and 2. In addition, HiCS uses conditional probability distributions with $(d-1)$ conditions and might suffer from the empty space issue as ENCLUS and PODM do.

3.3. Our Correlation Measure CMI

We now introduce CMI. The organization of this section is as follows. First, we propose a new general notion of correlation measure that is more robust to the curse of dimensionality than the one in Equation (3.1). This new notion also forms the basis for as well as explains the intuition behind both CMI (covered in this Section and Section 3.4) and CMI++ (covered in Sections 3.5 and 3.6). Second, we introduce cumulative entropy (CE), which is used to instantiate CMI. Third, we present the actual CMI measure.

3.3.1. Correlation measure – another general form

Our goal is to have correlation measures that are robust to the curse of dimensionality. Towards achieving this goal, we make the observation that working with high dimensional joint distributions (as in Equation (3.1)) or conditional distributions with many dimensions in the conditional part (as in HiCS) exposes us to the empty space issue. Thus, one solution is to avoid such distributions. In particular, consider $S = \{X_1, \ldots, X_d\}$. We introduce another generalized form to implement $Corr(S)$, or $Corr(X_1, \ldots, X_d)$, which meets our objective:

$$Corr(X_1, \ldots, X_d) \sim \sum_{i=2}^{d} \mathit{diff}\left(p(X_i), p(X_i \mid X_1, \ldots, X_{i-1})\right). \tag{3.2}$$

Equation (3.2) can be considered as a factorized form of Equation (3.1). In particular, it computes the correlation $Corr(X_1, \ldots, X_d)$ of $X_1, \ldots, X_d$ by aggregating the difference between the marginal distribution $p(X_i)$ and the conditional distribution $p(X_i \mid X_1, \ldots, X_{i-1})$ for $i \in [2, d]$. In this way, loosely speaking $Corr(X_1, \ldots, X_d)$ is the sum of the correlation scores of subspaces

$$(X_1, X_2), \ldots, (X_1, \ldots, X_i), \ldots, (X_1, \ldots, X_d)$$

if we consider $\mathit{diff}(p(X_i), p(X_i \mid X_1, \ldots, X_{i-1}))$ to be the correlation score of the subspace $(X_1, \ldots, X_i)$. The advantage of using lower dimensional subspaces is that the correlation measure is more robust to the empty space phenomenon. Thus, by avoiding the joint distribution $p(X_1, \ldots, X_d)$ as well as conditional distributions $p(X_i \mid \{X_1, \ldots, X_d\} \setminus \{X_i\})$, Equation (3.2) is able to mitigate the curse of dimensionality in correlation computation.

Please also note that Equation (3.2) is inspired from and hence related to the correlation measure of HiCS mentioned in Section 3.2. In particular, both quantify correlation through the divergence of marginal and the respective conditional distributions. However, for each d-dimensional subspace, while HiCS uses conditional distributions with $(d-1)$ conditions, Equation (3.2) uses conditional distributions with 1 to $(d-1)$ conditions, which helps to mitigate the curse of dimensionality when d is high. Further, while HiCS randomly picks the dimensions X_i to form marginal and conditional distributions, Equation (3.2) makes sure that all dimensions are included in the correlation assessment.

In addition, our factorized form of correlation measure in Equation (3.2) has its link to the total correlation. In particular, from Chapter 2, we have that $T(X_1, \ldots, X_d) = KL(p(X_1, \ldots, X_d) \mid\mid \prod_{i=1}^{d} p(X_i))$. Following [CT06], we have a factorization property of the KL divergence:

$$\begin{aligned} &KL(p(X_1, \ldots, X_d) \mid\mid \prod_{i=1}^{d} p(X_i)) \\ &= KL\left(p(X_2 \mid X_1) \mid\mid p(X_2)\right) + \ldots + KL\left(p(X_d \mid X_1, \ldots, X_{d-1}) \mid\mid p(X_d)\right) \quad . \end{aligned}$$

In other words, $T(X_1, \ldots, X_d)$ is the summation of the KL distances between one-dimensional (conditional) pdfs. This means that the correlation of $X_1, \ldots, X_d$ can be represented as the summation of the difference between one-dimensional (conditional) pdfs. Thus, our factorized form of correlation measure is justified.

However, unlike Equation (3.1), the correlation measure in Equation (3.2) may be variant to the way we form the factorization, i.e., the permutation of dimensions used. To build a general notion, our goal here is to eliminate such dependence. Thus, we derive a permutation-free version of Equation (3.2) as follows. Let $\mathcal{F}_d$ be the set of bijective functions $\sigma : \{1, \ldots, d\} \to \{1, \ldots, d\}$.

Definition 5. ***Factorized correlation measure**:*
The correlation score of $\{X_1, \ldots, X_d\}$ is

$$Corr(X_1, \ldots, X_d) = \max_{\sigma \in \mathcal{F}_d} \sum_{i=2}^{d} \mathit{diff}\left(p(X_{\sigma(i)}), p(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})\right). \quad (3.3)$$

Equation (3.3) eliminates the dependence on any specific permutation by taking the maximum score over all permutations. By considering the maximum value, we aim at uncovering the best correlation score of the dimensions involved. This design choice also is in line with maximal correlation analysis [BF85, RSX+11]; see Chapter 6 for more information.

In principle, one could instantiate diff in Equation (3.3) by KL divergence, and $Corr$ becomes the total correlation. However, due to the issues differential entropy on real-valued data (cf., Chapter 2), this turns out not to be a good choice. Our goal instead is to work with *cumulative entropy*, a new notion of entropy designed for real-valued data.

3.3.2. Cumulative entropy

In short, cumulative entropy (CE) captures the information content, i.e., complexity, of a probability distribution. However, different from Shannon entropy, it works with (conditional) cdfs, which turn out to facilitate computation on real-valued data.

Definition 6. ***Cumulative Entropy (CE)**:*
The cumulative entropy of a continuous random variable X, denoted as $h(X)$, is:

$$h(X) = -\int_{dom(X)} P(X \leq x) \log P(X \leq x) dx.$$

Similarly to Shannon entropy, the cumulative entropy of X captures the amount of uncertainty contained in X. Different from Shannon entropy, it is defined based on the cumulative distribution $P(X \leq x)$. From $0 \leq P(X \leq x) \leq 1$, we obtain $h(X) \geq 0$. This means that CE is always non-negative just like Shannon entropy defined on discrete data.

The notion of cumulative entropy in Definition 6 is based on [RCVW04, CL09]. However, it is more general since it is not restricted to non-negative random variables. Furthermore, following [RCVW04, CL09] we extend the notion of CE to conditional cumulative entropy and show that it maintains some important properties of conditional Shannon entropy on discrete/categorical data, as follows.

Definition 7. ***Conditional Cumulative Entropy****:*
The conditional CE of any real-valued random variable X knowing that some random vector $V \in \mathbb{R}^B$ (with B being a positive integer) takes the value v is defined as:

$$h(X \mid v) = -\int_{dom(X)} P(X \leq x \mid v) \log P(X \leq x \mid v) dx.$$

The CE of X conditioned by V is:

$$E_V[h(X \mid V)] = \int_{dom(V)} h(X \mid v) p(v) dv.$$

Just like the usual conditional entropy, we denote $E_V[h(X \mid V)]$ as $h(X \mid V)$ for notational convenience. The conditional CE has two important properties given by the following theorems.

Theorem 1. $h(X \mid V) \geq 0$ *with equality iff there exists a function*

$$f : dom(V) \rightarrow dom(X)$$

such that $X = f(V)$.

Proof. See [RCVW04, CL09]. □

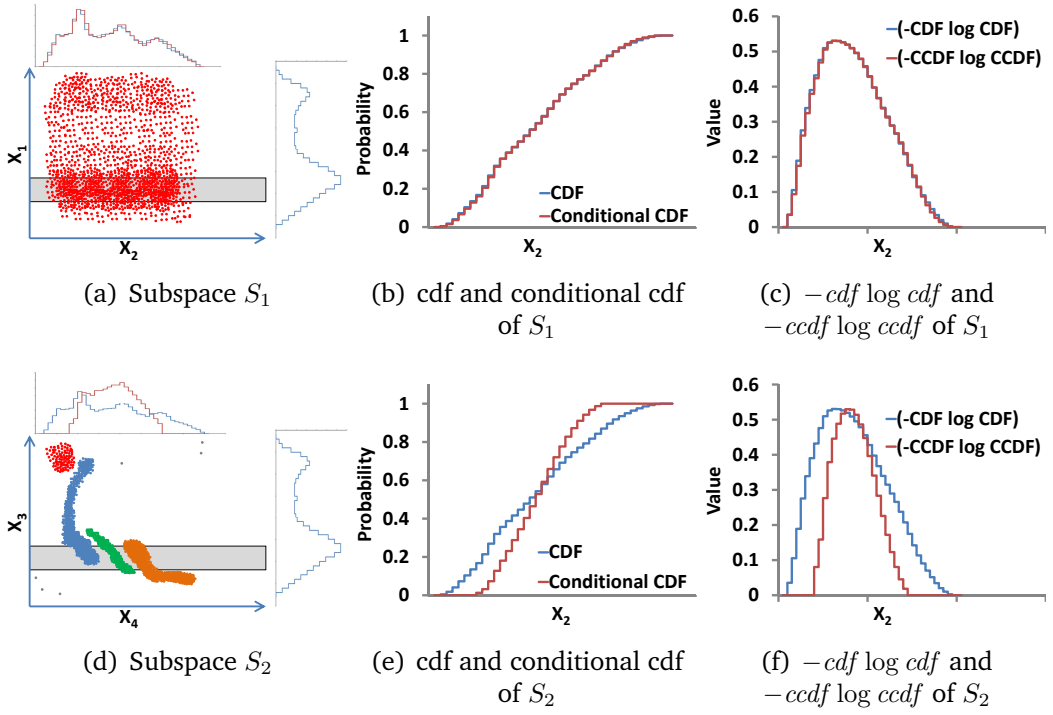
Theorem 2. $h(X \mid V) \leq h(X)$ *with equality iff X is independent of V.*

Proof. See [RCVW04, CL09]. □

In other words, Theorem 1 tells us that the conditional cumulative entropy $h(X \mid V)$ is non-negative and it is zero iff X is a functional of V. Theorem 2 in turn tells us that the conditional cumulative entropy $h(X \mid V)$ does not exceed the unconditional term $h(X)$, and equality takes place iff X is independent of V. In fact, these two properties of conditional CE match those of conditional Shannon entropy. Taking into account the fact that the unconditional CE has similar properties to unconditional Shannon entropy, we can see that CE preserves the good properties of Shannon entropy, which corroborates its suitability for data analysis.

3.3.3. Cumulative mutual information

Our goal here is to realize Equation (3.3) to create correlation measures that can alleviate the empty space issue. Hence, we create CMI measure instantiating Equation (3.3) by means of CE and conditional CE. In particular, using CE, we set $diff\,(p(x), p(x \mid \ldots))$ to $h(X) - h(X \mid \ldots)$. Thus, we have:



(a) Subspace S_1 (b) cdf and conditional cdf of S_1 (c) $-cdf \log cdf$ and $-ccdf \log ccdf$ of S_1

(d) Subspace S_2 (e) cdf and conditional cdf of S_2 (f) $-cdf \log cdf$ and $-ccdf \log ccdf$ of S_2

Figure 3.1.: Example subspaces with low and high correlation having different CMIs

Definition 8. ***Cumulative Mutual Information:***
*The cumulative mutual information (*CMI*) of real-valued random variables* $X_1, \ldots, X_d$ *is:*

$$\text{CMI}(X_1, \ldots, X_d) = \max_{\sigma \in \mathcal{F}_d} \sum_{i=2}^{d} h(X_{\sigma(i)}) - h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})$$

where $h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})$ *is* $h(X_{\sigma(i)} \mid V)$ *with* $V = (X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})$ *being a random vector whose domain is* $dom(X_{\sigma(1)}) \times \cdots \times dom(X_{\sigma(i-1)})$.

Intuitively, the more correlated $X_1, \ldots, X_d$ are, the smaller the conditional CE terms are, i.e., the larger CMI is. Thus, CMI is able to capture subspace correlation (Property 1). To further clarify this property, we use the toy example in Figure 3.1, which is based on that of [KMB12]. It depicts the scatter plots, cdf plots, and plots of the function $-P(X \leq x) \log P(X \leq x)$, namely $-cdf \log cdf$, of two subspaces S_1 and S_2 (ccdf means conditional cdf). The blue lines stand for the marginal distribution of the corresponding dimension. The red lines on the other hand feature the conditional distribution of one dimension obtained by selecting a range of the remaining dimension (gray stripes). One can see that S_2 has a higher score than S_1, and hence,

$$\text{CMI}(X_3, X_4)_{\text{selected range}} = 4.344 > \text{CMI}(X_1, X_2)_{\text{selected range}} = 0.113.$$

Further, even when high-order conditional CEs may be impacted by the curse of dimensionality, CMI still yields distinguishable correlation scores for high dimensional subspaces thanks to its member low-order conditional CE terms. Thus, CMI is more robust to the curse of dimensionality compared to existing correlation measures (Property 1). As CMI is based on the factorized form of correlation measure in Equation (3.2), its first difference to the measure of HiCS follows correspondingly (cf., Section 3.3.1). Another difference comes from the fact that the measure of HiCS quantifies the divergence between marginal and conditional distributions using statistical tests (e.g., Welch's t-test or Kolmogorov-Smirnov test) while CMI employs cumulative entropy. Our experiments later will further reveal the difference between CMI and HiCS in terms of performance.

Regarding additional characteristics of CMI, we can see that if $X_1, \ldots, X_d$ are m-wise independent, then CMI$(X_1, \ldots, X_d)$ is low as $h(X_i) - h(X_i \mid \ldots)$ vanishes for $i \leq m$ (Property 3). Moreover, we prove that CMI $= 0$ iff $X_1, \ldots, X_d$ are mutually independent (Property 2).

Theorem 3. CMI$(X_1, \ldots, X_d) \geq 0$ *with equality iff* $X_1, \ldots, X_d$ *are statistically independent.*

Proof. We have:

$$\text{CMI}(X_1, \ldots, X_d) \geq \sum_{i=2}^{d} h_{CE}(X_i) - h_{CE}(X_i \mid X_1, \ldots, X_{i-1}) \quad .$$

As conditioning reduces CE (cf., Theorem 2), it holds that $h_{CE}(X_i) - h_{CE}(X_i \mid X_1, \ldots, X_{i-1}) \geq 0$. Thus CMI$(X_1, \ldots, X_d) \geq 0$. Furthermore, the equality occurs when X_i is independent from $(X_1, \ldots, X_{i-1})$ for $2 \leq i \leq d$. This implies that $p(X_1, \ldots, X_d) = \prod_{i=1}^{d} p(X_i)$.

Conversely, when $X_1, \ldots, X_d$ are mutually independent, X_i is independent from $(X_1, \ldots, X_{i-1})$ for $2 \leq i \leq d$. Thus, we have:

$$\sum_{i=2}^{d} h_{CE}(X_i) - h_{CE}(X_i \mid X_1, \ldots, X_{i-1}) = 0 \quad .$$

This also holds for other permutation. Thus, CMI$(X_1, \ldots, X_d) = 0$. □

As another result, we observe that:

Lemma 5. CMI$(X_1, \ldots, X_d) \leq \max_{\sigma \in \mathcal{F}_d} \sum_{i=2}^{d} h(X_{\sigma(i)})$.

Proof. The result follows from Theorem 1. □

Lemma 5 essentially says that CMI$(X_1, \ldots, X_d)$ is upper-bounded by sums of marginal (unconditional) CE terms. We will use this lemma in Section 3.5 for normalization purposes, which leads to the creation of CMI++.

Heuristic choice of permutation. To compute CMI, ideally we have to find the optimal permutation $\sigma^* \in \mathcal{F}_d$. There are $d!$ possible cases in total. Together with the exponential number of subspaces, a brute-force approach is impractical. We therefore propose a heuristic to obtain a permutation that approximates σ^*. In particular, we first pick a pair of dimensions X_a and X_b ($1 \leq a \neq b \leq d$) such that $h(X_b) - h(X_b \mid X_a)$ is maximal among the possible pairs. Then, we continue selecting the next dimension X_c ($c \neq a$ and $c \neq b$) such that $h(X_c) - h(X_c \mid X_a, X_b)$ is maximal among the remaining dimensions. Likewise, at each step, assuming that $I = \{X_{p_1}, \ldots, X_{p_k}\}$ is the set of dimensions already picked and $R = \{X_{r_1}, \ldots, X_{r_{d-k}}\}$ is the set of remaining ones, we select the dimension $X_{r_i} \in R$ such that $h(X_{r_i}) - h(X_{r_i} \mid I)$ is maximal. The process goes on until no dimension is left. We let σ^* be the permutation obtained by our strategy. Our experiments will show that this heuristic works well in practice.

3.4. Computing CMI

To compute CMI, we need to compute CE and conditional CE.

3.4.1. Computing unconditional CE

Let $X[1] \leq \ldots \leq X[N]$ be realizations of X.

Theorem 4. *We have:*

$$h(X) = -\sum_{i=1}^{N-1} (X[i+1] - X[i]) \frac{i}{N} \log \frac{i}{N}.$$

Proof. See [RCVW04, CL09]. □

From Theorem 4, we can see that the unconditional CE can be computed in closed form on empirical data. Compared to unconditional Shannon entropy, unconditional CE thus is more suited to practical data analysis on real-valued domains.

3.4.2. Computing conditional CE

In contrast to the straightforward computation of unconditional CE, it is not as simple to calculate the conditional CE in an accurate and efficient way. In the following, we first point out that, due to limited data, sticking to the exact formula of conditional CE may lead to inaccurate results. We then propose a strategy to resolve the issue.

Consider d dimensions $X_1, \ldots, X_d$ and assume that we want to compute $h(X_1 \mid X_2, \ldots, X_d)$. From Definition 7, we have:

$$h(X_1 \mid X_2, \ldots, X_d) = \int_{dom(X_2)} \cdots \int_{dom(X_d)} h(X_1 \mid x_2, \ldots, x_d) p(x_2, \ldots, x_d) dx_2 \cdots dx_d.$$

Further:

$$h(X_1 \mid x_2, \ldots, x_d) = \lim_{\epsilon \to 0^+} h(X_1 \mid x_2 - \epsilon \leq X_2 \leq x_2 + \epsilon, \ldots, x_d - \epsilon \leq X_d \leq x_d + \epsilon).$$

This means $h(X_1 \mid x_2, \ldots, x_d)$ is computed based on the data in the hypercube $[x_2 - \epsilon, x_2 + \epsilon] \times \cdots \times [x_d - \epsilon, x_d + \epsilon]$ where ϵ can be arbitrarily small. Since N is finite, the expected number of data points falling into this hypercube approaches 0 as $\epsilon \to 0^+$ [LV07]. When d is large, the problem is exacerbated as one faces the empty space phenomenon. With empty hypercubes (or even hypercubes of one data point), $h(X_1 \mid x_2, \ldots, x_d)$ vanishes. Hence, $h(X_1 \mid X_2, \ldots, X_d)$ becomes 0. We thus encounter a paradox: *If sticking to the exact formula of conditional CE, we may end up with an inaccurate result*.

To alleviate this problem, instead of computing conditional CE for each individual tuple $(x_2, \ldots, x_d)$ that is 0 due to the empty space phenomenon, we switch to combinations of different tuples. This is to increase the likelihood that we have enough points for a meaningful computation. Therefore, we resort to data summarization by clustering.

In principle, clustering summarizes the data by means of clusters. Since the number of clusters is generally much less than the original data size, we may have more data points in each cluster. Hence, the issue of limited data is mitigated. Assuming that a clustering algorithm $\mathcal{C}$ is used on $\mathbf{DB}$ projected to $\{X_2, \ldots, X_d\}$ resulting in Q clusters $\{C_1, \ldots, C_Q\}$ (the support of C_i is $|C_i|$), we propose to estimate $h(X_1 \mid X_2, \ldots, X_d)$ by $\sum_{i=1}^{Q} \frac{|C_i|}{N} h(X_1 \mid C_i)$. That is, we consider each cluster to be a condition on X_1. As each cluster potentially contains many data points, this helps us to overcome the empty space problem. Note that if M is kept small enough, we will have enough points for a meaningful computation of $h(X_1 \mid C_i)$ regardless of the dimensionality d.

As our cluster-based approach does not rely on any specific cluster notion, it can be instantiated by any method. To ensure efficient computation of CMI, we use the one-pass K-means clustering strategy introduced in [OO04] with $K = Q$. Also following this paper, for $Q > 1$ the j^{th} centroid ($1 \leq j \leq M$) is first initialized to $\vec{\mu} \pm \frac{j-1}{(d-1)(Q-1)} \vec{\lambda}$ where the $\pm$ sign is picked randomly with probability $1/2$, and $\vec{\mu}$ and $\vec{\lambda}$ are the vector mean and standard deviation of $(X_2, \ldots, X_d)$, respectively (which are precomputed). That is, each centroid deviates from the vector mean

$\vec{\mu}$ by small fraction of $\vec{\lambda}$. Each incoming data point is assigned to the nearest centroid. The centroids are updated periodically after every $\sqrt{N}$ points. To reduce dependency on the order of data points, data perturbation is performed prior to clustering.

After doing clustering, we obtain Q clusters summarizing the data. For the parameter Q, if it is set too high, we may end up with high runtime and not enough data in each cluster for a reliable estimation of conditional CE. If it is instead set to 1, i.e., no clustering at all, $h(X_1 \mid \cdots)$ becomes $h(X_1)$, i.e., there is a loss of information. In our preliminary experiments, we find that $Q = 10$ yields a reasonably good balance between quality and efficiency, so we set Q to this value. Using clustering, applying the same proof as the one of Theorem 2, we conclude that the conditional CE is less than or equal to its respective unconditional CE.

3.4.3. Time Complexity Analysis

We now analyze the time complexity of computing CMI$(X_1, \ldots, X_d)$. According to Section 3.3.3, we first need to select a pair $\{X_a, X_b\} \subset \{X_1, \ldots, X_d\}$ such that $h(X_b) - h(X_b \mid X_a)$ is maximal. Following [OO04], computing $h(X_b) - h(X_b \mid X_a)$ for every pair $\{X_a, X_b\}$ costs $O(QN)$, i.e., $O(N)$ as Q is relatively small compared to N. Thus, processing all pairs costs $O(d^2N)$.

At each step, $I = \{X_{p_1}, \ldots, X_{p_k}\}$ is the set of dimensions already picked and $R = \{X_{r_1}, \ldots, X_{r_{d-k}}\}$ is the set of remaining ones. According to our heuristic to select the permutation of dimensions, we need to compute $h(X_{r_i}) - h(X_{r_i} \mid I)$ for every $X_{r_i} \in R$. To do so, we first cluster the data in the subspace formed by the dimensions of I. This costs $O(kQN)$, which can be simplified to $O(kN)$. Computing $h(X_{r_i}) - h(X_{r_i} \mid I)$ then costs $O(N)$ (we sort all dimensions only once). Thus, the total cost at this step is $O(kN + (d-k)N)$, i.e., $O(dN)$.

Overall, the total complexity of computing CMI$(X_1, \ldots, X_d)$ is

$$O(d^2N) + O\left(\sum_{k=2}^{d-1} dN\right),$$

which is equivalent to $O(d^2N)$.

3.5. CMI++: Improving over CMI and Its Implementation

As we have shown so far, CMI satisfies the properties for a reliable correlation analysis. Further, it permits robust and efficient computation on empirical data. However, CMI suffers from three issues. First, as we are going to use CMI in subspace search, we may need to compare CMI scores of subspaces with different dimensionality. For a fair comparison, those scores should be on the same scale. Yet, CMI currently does not guarantee this. Second, to compute the CMI score of

each subspace, we need to search for the optimal permutation of its dimensions. Though we have proposed a heuristic to tackle this, the approximate permutation is dependent on how well we estimate the conditional CE terms. As we currently compute these terms by data clustering, this essentially boils down to how good the selected clustering algorithm is. On the other hand, choosing a suitable clustering method is non-trivial. Thus, to eliminate that search and the issues just mentioned, we aim at a correlation measure defined on a fixed permutation of dimensions, i.e., no search is required. Third, to avoid data clustering in correlation computation, another approach to compute the conditional CE terms is required.

To address all of these three issues, we propose CMI++. In the following, we focus on the former two issues and postpone the third one to Section 3.6.

3.5.1. Normalized CMI

As mentioned above, for a fair comparison, the correlation scores of subspaces with different dimensionality should be on the same scale. In particular, we aim at normalizing the scores of CMI such that they fall into the range $[0, 1]$ with 0 being no correlation at all.

To this end, we observe that simply normalizing CMI$(X_1, \ldots, X_d)$ by d (i.e., normalizing the score of each d-dimensional subspace by its dimensionality) would not give correlation scores of the same range. This is because if we normalized CMI$(X_1, \ldots, X_d)$ by d, following Lemma 5, the normalized score would be upper-bounded by $\frac{\max_{\sigma \in \mathcal{F}_d} \sum_{i=2}^{d} h(X_{\sigma(i)})}{d}$. Since $\max_{\sigma \in \mathcal{F}_d} \sum_{i=2}^{d} h(X_{\sigma(i)})$ varies for different subspaces, the normalized score would not bring about a fair comparison of subspaces with different dimensionality.

Instead, we perform normalization based on the observation that for each permutation σ of dimensions $X_1, \ldots, X_d$, the term $\sum_{i=2}^{d} h(X_{\sigma(i)}) - h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})$ is bounded above by $\sum_{i=2}^{d} h(X_{\sigma(i)})$. In particular, we have:

Lemma 6. *It holds that:*

$$\sum_{i=2}^{d} h(X_{\sigma(i)}) - h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)}) \leq \sum_{i=2}^{d} h(X_{\sigma(i)})$$

with equality iff $X_{\sigma(2)}, \ldots, X_{\sigma(d)}$ are functions of $X_{\sigma(1)}$.

Proof. The result follows from Theorem 1. □

From Lemma 6, for unbiased comparison of the correlation scores of different subspaces, we propose the normalized CMI as follows.

Definition 9. ***Normalized Cumulative Mutual Information:***
The normalized CMI *of real-valued random variables* $X_1, \ldots, X_d$ *is:*

$$\text{CMI}_n(X_1, \ldots, X_d) = \max_{\sigma \in \mathcal{F}_d} \frac{\sum_{i=2}^{d} h(X_{\sigma(i)}) - h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})}{\sum_{i=2}^{d} h(X_{\sigma(i)})}$$

with the convention that $\frac{0}{0} = 0$.

That is, $\text{CMI}_n(X_1, \ldots, X_d)$ is the maximum *normalized* correlation over all permutation of $X_1, \ldots, X_d$. As one can see, we derive $\text{CMI}_n(X_1, \ldots, X_d)$ based on Lemma 6. To show that CMI_n is indeed suited to unbiased correlation analysis, we prove some of its good properties below.

Lemma 7. *We have:*

- $0 \leq \text{CMI}_n(X_1, \ldots, X_d) \leq 1$.
- $\text{CMI}_n(X_1, \ldots, X_d) = 0$ *iff* $X_1, \ldots, X_d$ *are statistically independent.*
- $\text{CMI}_n(X_1, \ldots, X_d) = 1$ *iff there exists* X_i *such that each* $X_j \in \{X_1, \ldots, X_d\} \setminus \{X_i\}$ *is a function of* X_i.

Proof. The results are derived from Theorem 3 and Lemma 6. □

Based on Lemma 7, CMI_n can be used to compare the correlation scores of different subspaces, regardless of their dimensionality. In addition, its value is bounded on both sides, which according to [Ren59] yields better interpretability than unbounded values. It is straightforward to verify that CMI_n satisfies Properties 1–3.

3.5.2. Practical CMI

To compute CMI_n, we still need to look for the permutation that maximizes the score. Our goal is to avoid this search. We therefore propose a practical (heuristic) version of CMI_n that achieves this goal. In other words, our strategy heuristically fixes a permutation for correlation computation, and hence, saves time. The details are as follows.

Definition 10. ***Practical Cumulative Mutual Information:***
The practical CMI *of real-valued random variables* $X_1, \ldots, X_d$ *is:*

$$\text{CMI}_p(X_1, \ldots, X_d) = \frac{\sum_{i=2}^{d} h(X_{\sigma(i)}) - h(X_{\sigma(i)} \mid X_{\sigma(1)}, \ldots, X_{\sigma(i-1)})}{\sum_{i=2}^{d} h(X_{\sigma(i)})}$$

where $\sigma \in \mathcal{F}_d$ is a permutation such that $h(X_{\sigma(1)}) \geq \ldots \geq h(X_{\sigma(d)})$.

In other words, CMI_p chooses the permutation corresponding to the sorting of dimensions in descending order of CE values. The intuition behind this choice is as follows. To compute CMI_n, we must find the permutation $\pi \in \mathcal{F}_d$ that maximizes

$$\frac{\sum_{i=2}^{d} h(X_{\pi(i)}) - h(X_{\pi(i)} \mid X_{\pi(1)}, \ldots, X_{\pi(i-1)})}{\sum_{i=2}^{d} h(X_{\pi(i)})}.$$

To maximize the above term, we should minimize the denominator and maximize the numerator. To minimize the former, it would most likely help to exclude $h(X_{\sigma(1)})$—the largest unconditional CE term. Thus, we expect a permutation where $X_{\sigma(1)}$ appears first to be good.

For the numerator, we have the following observation. Assume that $h(X_i) \geq h(X_j)$, i.e., X_i is more random than X_j. Then $h(X_k \mid X_i)$ tends to be smaller than $h(X_k \mid X_j)$ [RCVW04]. For instance, if X_j is deterministic, $h(X_k \mid X_i) \leq h(X_k \mid X_j) = h(X_k)$. So $h(X_k \mid \ldots)$ will get further away from $h(X_k)$ as the condition "$\ldots$" becomes more random, and vice versa.

Now assume that $h(X_k) \geq h(X_i)$. If X_k is after X_i in the permutation, $h(X_k)$ will appear in the numerator. However, $h(X_k \mid \ldots)$ will get close to $h(X_k)$ as "$\ldots$" containing X_i is less random, i.e., $h(X_k) - h(X_k \mid \ldots)$ tends to be small.

If X_k instead is before X_i in the permutation in turn, we will have $h(X_i)$ in the numerator. However, $h(X_i \mid \ldots)$ gets further away from $h(X_i)$, i.e., $h(X_i) - h(X_i \mid \ldots)$ tends to be relatively large.

All in all, these suggest that dimensions with large CE values should be placed before those with small CE values to maximize the numerator. Though we do not have a rigorous proof for our intuition, our experiments reveal that CMI_p works very well in practice. To show that CMI_p is also suited to unbiased correlation analysis, we prove some of its good properties below.

Lemma 8. *It holds that:*

- $0 \leq \text{CMI}_p(X_1, \ldots, X_d) \leq 1$.
- $\text{CMI}_p(X_1, \ldots, X_d) = 0$ *iff* $X_1, \ldots, X_d$ *are statistically independent.*
- $\text{CMI}_p(X_1, \ldots, X_d) = 1$ *iff* $X_{\sigma(2)}, \ldots, X_{\sigma(d)}$ *are functions of* $X_{\sigma(1)}$.

Proof. The results are derived from Theorem 3 and Lemma 6. □

Similarly to CMI_n, CMI_p also meets Properties 1–3. In addition to the original CMI measure, we will use CMI_p in the rest of this paper and call it CMI++. In summary, compared to CMI, CMI++ allows for unbiased correlation analysis

of subspaces with different dimensionality. In addition, its computation does not require searching for an optimal permutation. Please note that the aforementioned differences between CMI and HiCS also hold for CMI++. In the next section, we explain how to compute CMI++ on empirical data.

3.6. Computing CMI++

In this section, w.l.o.g., we assume that

$$\text{CMI++}(X_1, \ldots, X_d) = \frac{\sum_{i=2}^{d} h(X_i) - h(X_i \mid X_1, \ldots, X_{i-1})}{\sum_{i=2}^{d} h(X_i)}.$$

That is, the permutation of dimensions in $\text{CMI++}(X_1, \ldots, X_d)$, which is fixed (see Section 3.5.2), is assumed to be $X_1, \ldots, X_d$ for ease of presentation.

For $X \in \{X_1, \ldots, X_d\}$, let $X[1] \leq \ldots \leq X[N]$ be realizations of X. According to Theorem 4, we have

$$h(X) = -\sum_{i=1}^{N-1} (X[i+1] - X[i]) \frac{i}{N} \log \frac{i}{N}.$$

On the other hand, from Definition 7, we have

$$h(X_i \mid X_1, \ldots, X_{i-1}) = \int_{dom(X_1)} \cdots \int_{dom(X_{i-1})} h(X_i \mid x_1, \ldots, x_{i-1}) p(x_1, \ldots, x_{i-1}) dx_1 \cdots dx_{i-1}.$$

We propose to compute the conditional CE terms of $\text{CMI++}(X_1, \ldots, X_d)$ by what we refer to as *correlation-aware discretization*. We go for discretization as it also is an approach to summarize data/group values. However, existing simple discretization methods (e.g., equal-width) are oblivious of subspace correlation: They form bins without considering the dependencies among dimensions, and hence, they break correlation in the data. On the contrary, we formulate the computation of the conditional CE terms in CMI++ as optimization problems where we search for the discretization maximizing $\text{CMI++}(X_1, \ldots, X_d)$. That is, we search for the bins that preserve as well as reveal hidden correlation of the data. The details are as follows.

- *Computing $h(X_2 \mid X_1)$:* The value of $h(X_2 \mid X_1)$ depends on how we estimate the distribution of X_1, or in other words, how we discretize the realizations of X_1. Here, our goal is to maximize $h(X_2) - h(X_2 \mid X_1)$, i.e., maximizing $\text{CMI++}(X_1, \ldots, X_d)$. As $h(X_2)$ is fixed given the realizations of X_2, maximizing $h(X_2) - h(X_2 \mid X_1)$ is equivalent to minimizing $h(X_2 \mid X_1)$. To solve the latter problem, we propose to search for the discretization of X_1 such that

$h(X_2 \mid X_1)$ is minimal. As we will show in the following, we can efficiently tackle this optimization problem by dynamic programming.

- *Computing $h(X_i \mid X_1, \ldots, X_{i-1})$ for $i \geq 3$:* Ideally, one would search for the optimal discretizations of $X_1, \ldots, X_{i-1}$ that minimize $h(X_i \mid X_1, \ldots, X_{i-1})$. However, this is very computationally expensive. We overcome this issue by observing that, to this end, we have already discretized $X_1, \ldots, X_{i-2}$, and the resulting discretizations are for maximizing CMI++$(X_1, \ldots, X_d)$. Hence, we choose to search for the discretization of X_{i-1} only. To this end, we will also show that we can efficiently find the discretization of X_{i-1} that minimizes $h(X_i \mid X_1, \ldots, X_{i-1})$ by dynamic programming. By not re-discretizing any dimension already processed, we are able to reduce the computational cost and permit large-scale processing.

Overall, to compute CMI++$(X_1, \ldots, X_d)$, we take advantage of the fact that the order of dimensions has been fixed. From there, we compute each conditional CE term appearing in CMI++$(X_1, \ldots, X_d)$ (in that order) optimally using dynamic programming. To this end, our solution indeed is inspired from [RRF+11]. Yet, they focus on Shannon entropy and we focus on cumulative entropy.

We now prove that the discretization at each step can be searched efficiently by dynamic programming. In particular, assume that I is the set of dimensions we have already discretized. Further, assume that we want to compute $h(X' \mid I, X)$, i.e., we need to search for the discretization of X in the current step.

Let $X[1] \leq \ldots \leq X[N]$ be realizations of X. We write $X[j, u]$ for $\{X[j], X[j+1], \ldots, X[u]\}$ where $j \leq u$. We note that $X[1, N]$ is in fact X. Further, we write $\langle X[j, u] \rangle$ as a bin of X, which contains points of $\mathbf{DB}$ falling into the range from $X[j]$ to $X[u]$. Accordingly, we let $h(X' \mid I, \langle X[j, u] \rangle)$ denote $h(X' \mid I)$ computed using points of $\mathbf{DB}$ falling into that bin. To show that the optimal discretization of X minimizing $h(X' \mid I, X)$ can be searched by dynamic programming, we introduce the following formulation which will subsequently lead to the solution of our problem. In particular, for $1 \leq l \leq u \leq N$, we write

$$f_{X',I,X}(u, l) = \min_{g:|g|=l} h(X' \mid I, X^g[1, u])$$

where g is a discretization of $X[1, u]$, $|g|$ is its number of bins, and $X^g[1, u]$ is the set of bins g forms on $X[1, u]$. That is, $f_{X',I,X}(u, l)$ is the minimum $h(X' \mid I, X[1, u])$ over all discretization g of $X[1, u]$ into l bins. For brevity, we simply write $f_{X',I,X}$ as f. In the following, we show a recursive formula for $f(u, l)$ which inspires the use of dynamic programming to efficiently compute it, and hence, to efficiently solve our problem of minimizing $h(X' \mid I, X)$.

Theorem 5. *Let l, u be two integers such that $1 < l \leq u \leq N$. We have*

$$f(u, l) = \min_{j \in [l-1, u)} \mathcal{A}_j$$

where

$$\mathcal{A}_j = \frac{j}{u} f(j, l-1) + \frac{u-j}{u} h(X' \mid I, \langle X[j+1, u] \rangle).$$

Proof. Let $g^* = \arg\min_{g:|g|=l} h_{CE}(X' \mid B, X^g[1, u])$. We denote l bins that g^* generates on X as $b(X)_1, \ldots, b(X)_l$. We write $|b(X)_t|$ as the number of values of X in $b(X)_t$. For each $X'_i \in I$, we denote its bins as $b(X'_i)_1, \ldots, b(X'_i)_{n'_i}$.

Further, let $c_z = \sum_{i=1}^{z} |b(X)_i|$. Note that each bin of X is non-empty, i.e., $c_z \geq z$. We use $h_{CE}(X' \mid I, b_t)$ to denote $h_{CE}(X' \mid I)$ computed using the points of $\mathbf{DB}$ corresponding to the realizations of X in $b(X)_t$, projected onto X' and I. We write $|(t, t_1, \ldots, t_k)|$ as the number of points in the cell made up by bins $b(X)_t, b(X'_1)_{t_1}, \ldots, b(X'_{|I|})_{t_{|I|}}$.

We have: $f(u, l)$

$$\begin{aligned}
&= \sum_{t=1}^{l} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_{|I|}=1}^{n'_{|I|}} \frac{|(t, t_1, \ldots, t_k)|}{u} \times h_{CE}(X' \mid b(X)_t, b(X'_1)_{t_1}, \ldots, b(X'_{|I|})_{t_{|I|}}) \\
&= \sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_{|I|}=1}^{n'_{|I|}} \frac{|(t, t_1, \ldots, t_k)|}{u} \times h_{CE}(X' \mid b(X)_t, b(X'_1)_{t_1}, \ldots, b(X'_{|I|})_{t_{|I|}}) \\
&\quad + \frac{|b_l|}{u} h_{CE}(X' \mid I, b_l) \\
&= \frac{c_{l-1}}{u} \sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_{|I|}=1}^{n'_{|I|}} \frac{|(t, t_1, \ldots, t_k)|}{u} \times h_{CE}(X' \mid b(X)_t, b(X'_1)_{t_1}, \ldots, b(X'_{|I|})_{t_{|I|}}) \\
&\quad + \frac{u - c_{l-1}}{u} h_{CE}(X' \mid I, \langle X[c_{l-1}+1, u] \rangle) \\
&= \frac{c_{l-1}}{u} f(c_{l-1}, l-1) + \frac{u - c_{l-1}}{u} h_{CE}(X' \mid I, \langle X[c_{l-1}+1, m] \rangle) \quad .
\end{aligned}$$

In the last line,

$$\sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_{|I|}=1}^{n'_{|I|}} \frac{|(t, t_1, \ldots, t_k)|}{u} \times h_{CE}(X' \mid b(X)_t, b(X'_1)_{t_1}, \ldots, b(X'_{|I|})_{t_{|I|}})$$

is equal to $f(c_{l-1}, l-1)$ because otherwise, we could decrease $f(u, l)$ by choosing a different discretization of $X[1, c_{l-1}]$ into $l-1$ bins. This in turn contradicts our definition of $f(u, l)$. Since $c_{l-1} \in [l-1, u)$ and $f(u, l)$ is minimal over all $j \in [l-1, u)$, we arrive at the final result. □

Theorem 5 shows that the optimal discretization of $X[1, u]$ can be derived from

that of $X[1, j]$ with $j < u$. This allows us to design a dynamic programming algorithm to find the discretization of X that minimizes $h(X' \mid I, X)$.

We note that we have to impose a maximum number of bins on any discretization g considered. This is because in the extreme case when all realizations of X are distinct and $|g| = N$, $h(X' \mid I, X)$ will be zero. This is also known as the empty space issue [LV07]. Therefore, inspired by [RRF+11], we restrict that $|g| < N^\epsilon$ where $\epsilon \in (0, 1)$.

Efficiency Consideration. For each dimension, a cut point is a mid value of two of its consecutive distinct values. When discretizing a dimension at each step, if we consider the data at highest granularity (i.e., we use the original set of cut points of that dimension), the time complexity of dynamic programming is $O(N^3)$. This is too restrictive for large data sets. We overcome this problem by reducing the number of candidate cut points. In particular, when discretizing a dimension with maximum grid size max_grid (here, $max\_grid = N^\epsilon$), we limit its number of cut points to $c \times max\_grid$ with $c > 1$. Similarly to [RRF+11], we do this using equal-frequency discretization on the dimension with the number of bins equal to $(c \times max\_grid + 1)$. More elaborate pre-processing is possible, but beyond the scope of this work.

Regarding ϵ and c, the larger these are, the more candidate discretizations we consider. Hence, we can achieve a better result, but the computational cost becomes higher. Our preliminary empirical analysis shows that $\epsilon = 0.3$ and $c = 4$ offers a good balance between quality and efficiency, and we will use these values in the experiments.

Following an analysis similar to that of [RRF+11], the cost of discretizing a dimension is $O(c^3 \times max\_grid^3)$, which is equivalent to $O(N^{0.9})$ as $max\_grid = N^{0.3}$ and $c = 4$ is small. The total complexity of computing CMI++$(X_1, \ldots, X_d)$ is therefore $O(dN^{0.9})$.

Remarks. In fact, one also could use correlation-aware discretization to compute CMI, not only CMI++. However, in CMI, we need to search for the optimal permutation by the heuristic search (see Section 3.3.3). Thus, if correlation-aware discretization was used, at each step of the heuristic search, we would have to solve the optimization problem for all dimensions not yet selected at this step. This potentially incurs a high computational cost and hinders large-scale processing.

3.7. Subspace Search Scheme

For a D-dimensional data set, there are $2^D - 1$ candidate subspaces to examine. The exponential number of subspaces makes a brute-force search impractical. To address this issue, we adopt the Apriori search scheme proposed in HiCS [KMB12]. It is an approximate, yet scalable, levelwise subspace search framework.

This search scheme relies on the intuition that a correlated high dimensional subspace likely has its correlation reflected in its lower-dimensional projections. In

the field of subspace clustering, there is an analogous observation: High dimensional subspace clusters tend to have their data points clustered in all lower dimensional projections [AGGR98, KKRW05, AKMS07]. One could then apply a levelwise search to mine subspaces having correlation scores larger than a pre-specified value. However, to facilitate parameterization, the search scheme in HiCS avoids imposing direct thresholds on correlation scores.

Instead, it employs a beam search strategy to obtain efficiency. Starting with two-dimensional subspaces, in each step it uses the top M subspaces of high correlation scores to generate new candidates in a levelwise manner. A newly generated candidate is only considered if all of its child subspaces have high correlation scores. First, this requirement permits tractable time complexity. Second, it takes into account interaction among subspaces of different dimensionality, and selects subspaces that are ensured to have high correlation scores. Third, it avoids redundancy; if $T \subseteq S$ and S has a higher correlation score than T then T is excluded from the final result. Experiments in Section 3.8 demonstrate that this search scheme yields good results when it comes to clustering and outlier analysis.

3.8. Experiments

We compare our methods, CMI and CMI++, to two existing methods for mining correlated subspaces: ENCLUS and HiCS. As further baselines, we include FB [LK05] which selects subspaces randomly, and PCA [LV07]. The methods discussed so far select subspaces not bound to any specific data mining task. As specialized techniques, we consider HOD [AY01] for subspace outlier detection, and CLIQUE [AGGR98] and PROCLUS [APW+99] for subspace clustering.

For our methods, we use $Q = 10$ for CMI, and $\epsilon = 0.3$ and $c = 4$ for CMI++ unless stated otherwise. For the Apriori search scheme, we set $M = 400$. For each competitor, we tried to find its optimal parameter setting. As FB and HOD are non-deterministic, we record 5 runs for each of them and give the average (standard deviations were recorded, but negligible).

We evaluate how correlated subspaces improve the result quality of outlier detection and clustering techniques. Therefore, LOF [BKRTN00] and DBSCAN [EKSX96], two well-established methods for pattern detection, are selected on top of the approaches tested. To ensure comparability, we use the same parameter value of $MinPts$ for LOF across *all* experiments. This in turn is not possible with DBSCAN as the values of its parameters, especially the neighborhood distance threshold, are known to vary from one data set to another [MGAS09, GFM+11]. Thus, instead of using a global parameter setting for DBSCAN, we customize it for each data set. Yet, for each data set, we apply the same setting when processing the subspaces output by different subspace search methods.

To ensure succinct sets of subspaces that allow for post-analysis, only the best 100 subspaces of each technique are utilized for clustering and outlier detection. As a standard practice [LK05, KMB12, MSS11], we assess outlier detection results by

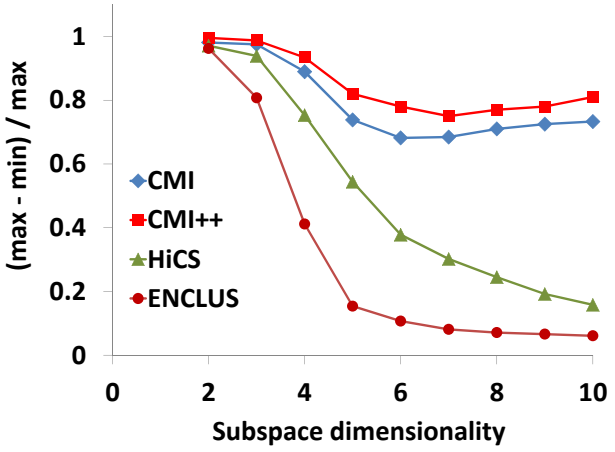


Figure 3.2.: Higher is better] Correlation score vs. dimensionality.

the Area Under the ROC Curve (AUC). Similarly, we evaluate clustering results by means of F1 and Accuracy [MAG+09, MGAS09, GFM+11].

All techniques were implemented in Java (to ensure fair runtime comparison) and all experiments were conducted on an Intel® i5-2520M Processor with 8GB memory.

3.8.1. Impact of dimensionality

To illustrate that our methods are robust to increasing dimensionality of subspaces, we evaluate them on a synthetic data set of 20 dimensions and 5120 instances, generated according to [MSS11]. In this data set, we embed subspace clusters in randomly selected 2–10 dimensional subspaces. Additionally, 120 outliers are created deviating from these clusters. Please note that in this experiment, we perform an exhaustive search without any pruning. Because of the large total number of subspaces ($2^{20}-1$), we only experiment up to $d = 10$ to avoid excessive runtime. We record $\frac{\max A_d - \min A_d}{\max A_d}$ where A_d is the set of correlation scores of all d-dimensional subspaces. For $2 \leq d \leq 10$, $\min A_d \approx 0$ as there are uncorrelated d-dimensional subspaces, and $\max A_d \neq 0$ as there are correlated d-dimensional subspaces with clusters and outliers. Hence, ideally $\frac{\max A_d - \min A_d}{\max A_d} = 1$ for $2 \leq d \leq 10$. For each correlation measure and each value of d, the closer $\frac{\max A_d - \min A_d}{\max A_d}$ to 1, the better.

The results are in Figure 3.2. We can see that CMI and CMI++ are relatively robust to dimensionality: They yield discriminative correlation scores for high dimensional subspaces with CMI++ having better performance. This is because both CMI and CMI++ work with factorized distributions, i.e., they can mitigate the empty space issue. In addition, CMI++ computes conditional CE terms in a more principled and accurate way. Thus, it can better capture correlation of subspaces.

On the other hand, HiCS and ENCLUS produce non-discriminative correlation scores for high dimensional subspaces. Specifically, when dimensionality increases, subspaces tend to have similar correlation scores under their schemes, i.e., their correlation measures are overshadowed by the curse of dimensionality. Hence, ENCLUS starts assigning somewhat indifferent correlation scores starting from dimensionality 5 while that of HiCS is dimensionality 9. Thus, using high dimensional conditional and joint probability distributions in correlation measures may lead to unreliable correlation analysis.

3.8.2. Synthetic data: clustering and outlier detection

Based on the method described in [MSS11], we generate synthetic data sets with 5120 data points and 20, 40, 80, and 120 dimensions. Each data set contains subspace clusters embedded in randomly chosen 2-6 dimensional subspaces of the full space and 120 outliers deviating from these clusters. Higher dimensional subspaces are not used to ensure reliability of correlation scores produced by HiCS and ENCLUS.

Quality for outlier detection. The quality of subspaces is evaluated by inspecting how they enhance outlier detection compared to LOF without subspace selection as the baseline. Since we are experimenting with outlier mining, we include HOD as another competitor. The results are given in Figure 3.3. Overall, CMI++ achieves the best outlier detection results and CMI comes in second. Further, both are more stable than competitors with increasing dimensionality.

The performance of LOF degrades with increasing dimensionality of data, underscoring the problem of using the noisy full spaces. FB is affected by random selection of subspaces with little to no correlation. HiCS and ENCLUS in turn have low AUC values due to the problems we point out in Section 3.2. HOD considers objects falling into sparse hypercubes in multivariate subspaces as outliers. Thus, it is prone to the empty space issue, which explains its degrading performance with respect to increasing dimensionality. PCA shows the worst performance due to its inability to capture complex correlations in multivariate subspaces. As subsequent evaluation confirmed this trend, we exclude PCA in the following experiments.

Next, we briefly illustrate the parameterization of both CMI and CMI++. Regarding CMI, we study the impact of setting Q (i.e., choosing a suitable clustering setting) on its computation. For CMI++, we examine its performance w.r.t. different values of ϵ and c. As a representative case, in Figure 3.4, we display the outlier detection results of CMI, with Q varied, on the synthetic data set of 5120 data points and 20 dimensions. Using the same data set, Figure 3.5(a) features the outlier detection results of CMI++ with $c = 4$ and ϵ varied, while Figure 3.5(b) shows its results with $\epsilon = 0.3$ and c varied. We skip the results on data sets of other size and dimensionality as they exhibit a similar trend. Going over the results, we see that the quality of CMI is impacted when Q changes. Further, there is no value of Q, i.e., no clustering setting, that clearly stands out in terms of quality. CMI++ in turn is not involved in such an issue: Its performance is relatively stable w.r.t.

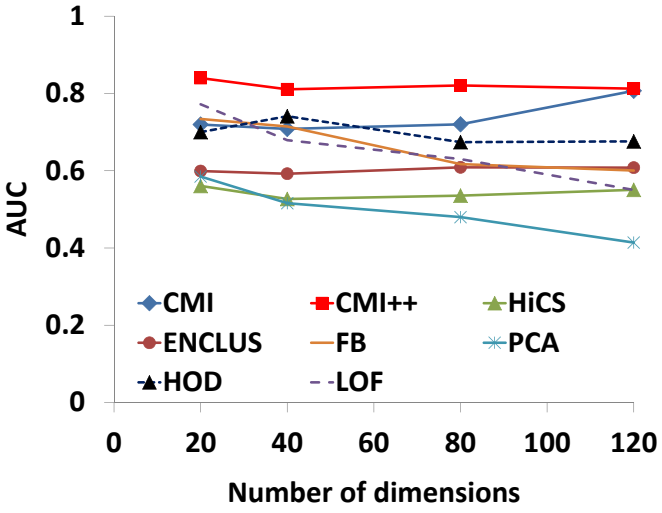


Figure 3.3.: [Higher is better] Outlier detection results on synthetic data: AUC vs. dimensionality.

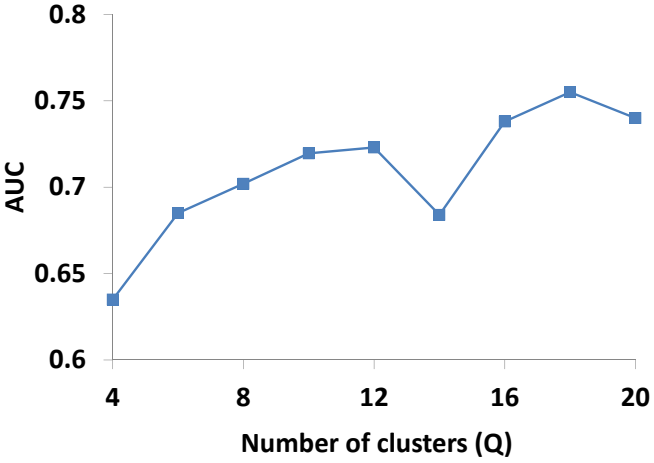
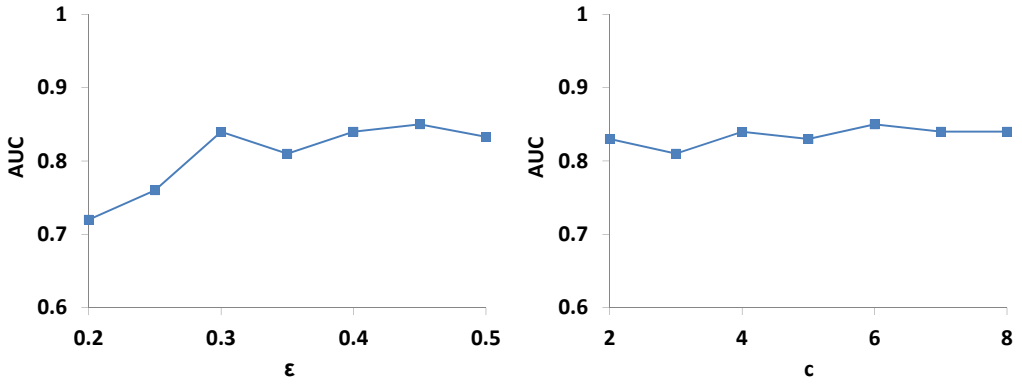


Figure 3.4.: [Higher is better] Sensitivity of CMI to the selection of clustering setting (Q) on a synthetic data set with $N = 5120$ and $D = 20$.

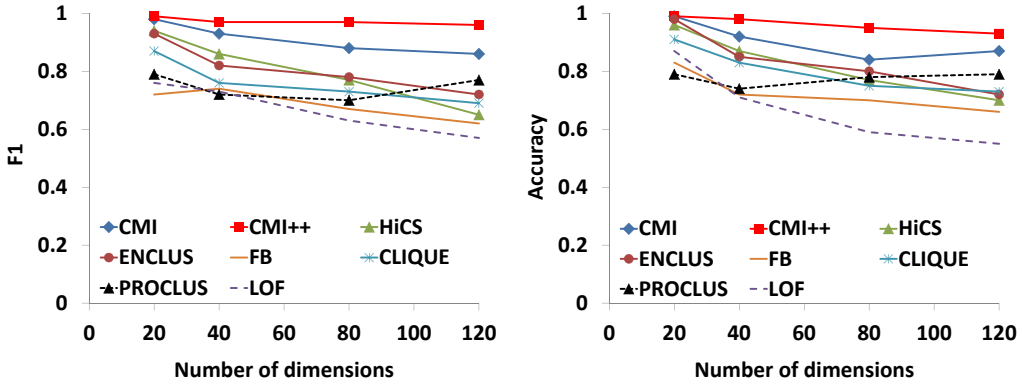
both ϵ and c. This shows that by allowing a clustering-free computation on empirical data, CMI++ allows easier parameterization. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable parameterization might be required.

Quality for clustering. In this experiment, we assess the quality of subspaces by clustering results. We use DBSCAN as the baseline clustering algorithm. For each subspace search method tested, we reduce redundancy in its set of discovered clusters following the scheme proposed in [AKMS07]. Besides, we include CLIQUE and PROCLUS as further baselines. The results are in Table 3.6. We again see that CMI++ and CMI achieve respectively the best and second best quality in terms



(a) AUC vs. ϵ (b) AUC vs. c

Figure 3.5.: [Higher is better] Sensitivity of CMI++ to ϵ and c on a synthetic data set with $N = 5120$ and $D = 20$.



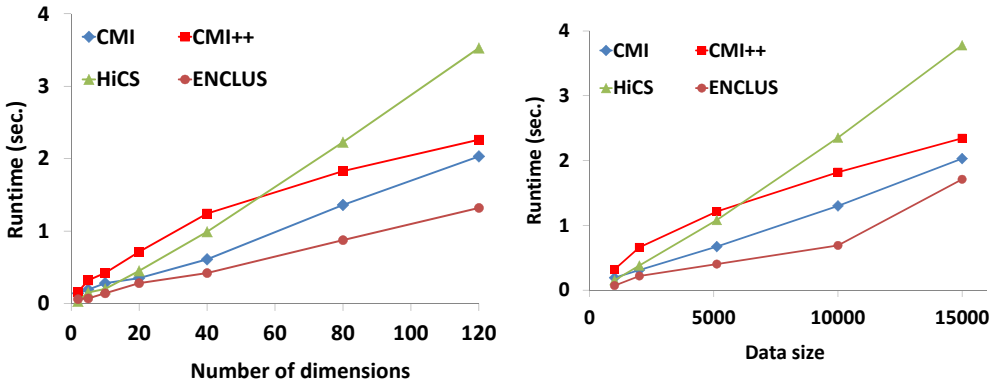
(a) F1 vs. dimensionality (b) Accuracy vs. dimensionality

Figure 3.6.: [Higher is better] Clustering results on synthetic data.

of both F1 and Accuracy values. They also scale better with dimensionality than other methods.

Overall, the experiments on synthetic data point out that CMI++ and CMI perform well in selecting subspaces containing clusters and outliers. We skip the experiment on quality against data size as the results show similar tendency of all methods. Furthermore, real-world data sets included in our empirical study already have various sizes that can be served as an assessment of quality against data size.

Runtime scalability. We now study the runtime scalability of our methods. We observe that different methods may find subspaces of different dimensionality, i.e., they reach different levels of the subspace lattice. Thus, it does not make sense to compare the time subspace search methods spend for mining subspaces. Instead, we assess their runtime in computing the correlation score of the same data space with the same number of objects. Our objective is to compare the scalability of all



(a) Runtime vs. dimensionality (b) Runtime vs. data size

Figure 3.7.: [Lower is better] Scalability results on synthetic data.

methods with regard to the dimensionality and the data size of the space. To this end, we employ synthetic data sets. In particular, for scalability in dimensionality, we generate synthetic data set of size 1000 and the number of dimensions varying from 2 to 120. For scalability in data size, we generate synthetic data set of dimensionality 5 and the number of objects varying from 1000 to 15000. Note that for all data sets, the correlation computation is limited to the full space. The results are in Figure 3.7.

We see that ENCLUS has the best scalability due to its simple computation of total correlation. CMI has the second best scalability. CMI++ scales better than HiCS. Besides, the scalability of CMI++ approaches that of CMI with increasing dimensionality and data size. Overall, CMI and CMI++ have linear scalability in both dimensionality and data size. They both take less than 2.5 seconds to process a data set with 120 dimensions and a data set with 15000 data points. Combining with the results on quality, we conclude that our methods yield high accuracy for correlation assessment at low computational cost.

3.8.3. Evaluation on real-world data

All real world data sets used in our experiments are from the UCI Machine Learning Repository and have been used as benchmarks in recent publications [LK05, KMB12, GFM+11, MAG+09]. The characteristics of the data sets are summarized in Table 3.1.

Quality for outlier detection. We evaluate the performance of all subspace search methods with outlier detection on real world data. We perform experiments on 11 benchmark datasets, using the minority class as ground truth for the evaluation of detected outliers. In some of these data sets, e.g., Pendigits, all classes have identical support and we down-sample one class to 10% of its original size—a procedure commonly used in outlier evaluation [LK05, KMB12, MSS11]. The results are in Table 3.2. In each row, we highlight the top two best values. Overall, CMI++ achieves the best AUC score in 8 out of 11 data sets and is the second best in 2

Data set	Attributes	Size
Ann-Thyroid	21	3772
Arrhythmia	129	420
Diabetes	8	768
Glass	9	214
Ionosphere	34	351
Lymphography	18	148
Madelon	500	600
Pendigits	16	7494
Segment	19	2310
Shape	17	160
Vowel	10	990
WBC	33	198
WBCD	30	569

Table 3.1.: Characteristics of real world data sets.

other data sets. CMI has the best AUC score in 3 data sets and is the second best in 2 data sets.

By applying a Friedman test [Dem06] at significance level $\alpha = 0.05$, we find that the observed differences in AUC values of all methods are significant. By performing a post-hoc Nemenyi test, we learn that CMI++ significantly outperforms HiCS, ENCLUS, FB, HOD, and LOF. Using a Wilcoxon signed rank test with $\alpha = 0.05$ to compare CMI++ and CMI, we find CMI++ to be significantly better.

The overall conclusion is that our methods, especially CMI++, provide the best quality improvement for LOF.

Quality for clustering. Here, we pick 5 data sets that are frequently used to assess subspace clustering methods [MGAS09]. According to the results in Table 3.3 (in each row, the top two best values are highlighted), CMI and CMI++ provide the best quality improvement in DBSCAN.

By applying a Friedman test at $\alpha = 0.05$, we find the observed differences between the methods to be significant. A Nemenyi test in the post-hoc analysis shows that CMI++ performs significantly better than HiCS, ENCLUS, FB, CLIQUE, PROCLUS, and LOF. A Wilcoxon signed rank test between CMI++ and CMI shows CMI++ to be significantly better.

3.9. Conclusion

So far, we have introduced CMI and CMI++, two new correlation measures for analyzing multivariate data. In short, they are based on cumulative entropy of subspaces and are robust to the curse of dimensionality. In addition, they are not restricted to pairwise analysis, and capture mutual dependencies among dimensions.

Data set	CMI	CMI++	HiCS	ENCLUS	FB	HOD	LOF
Ann-Thyroid	0.75	0.80	**0.84**	0.75	**0.81**	0.71	0.75
Arrhythmia	**0.63**	**0.65**	0.58	0.59	0.60	0.58	0.57
Diabetes	0.60	**0.62**	**0.61**	**0.61**	0.58	0.55	0.53
Glass	0.50	**0.52**	0.51	0.50	0.50	0.51	**0.52**
Ionosphere	0.59	**0.61**	0.59	0.59	**0.60**	0.57	0.59
Lymphography	**0.83**	**0.85**	0.80	0.81	0.81	0.76	0.80
Madelon	0.82	**0.86**	0.79	0.83	0.82	0.83	**0.87**
Pendigits	0.97	**0.98**	0.77	0.94	0.91	0.89	**0.98**
Segment	**0.64**	**0.63**	0.61	0.60	0.57	0.60	0.59
WBC	**0.94**	**0.94**	0.93	0.92	0.92	0.77	0.69
WBCD	**0.44**	**0.44**	0.42	0.43	0.40	0.41	0.40
Average	**0.70**	**0.72**	0.68	0.69	0.68	0.65	0.66

Table 3.2.: [Higher is better] Outlier detection results (AUC values) on real world data. Top two best values in each row are in **bold**. CMI++ significantly outperforms HiCS, ENCLUS, FB, HOD, and LOF under a Friedman test with $\alpha = 0.05$. CMI++ significantly outperforms CMI under a Wilcoxon signed rank test with $\alpha = 0.05$.

Furthermore, they allow efficient and reliable computation on empirical data. To find correlated subspaces, we employ the Apriori search scheme in [KMB12] that can explore the exponential search space of subspaces in practical runtime. Experiments on various synthetic and real world data sets show that compared to existing techniques, our methods deliver better improvement for *both* clustering and outlier detection.

In this chapter, we focus on information-theoretic correlation measures and the Apriori search scheme for mining correlated subspaces in real-valued data. This opens a research question: Is there any better way to detect correlated subspaces in multivariate data? In the course of tackling this question, we find out that the method proposed in this chapter tends not to detect *high dimensional* correlated subspaces. This is because the Apriori search scheme is biased against high dimensionality. In particular, the higher the level of the subspace lattice, the more restrictions are placed on the respective subspaces. As a consequence, the Apriori search scheme tends to produce subspaces that are fragments of some high dimensional correlated subspaces, which cause redundancy. Motivated by this observation, we have devised a new scalable subspace search scheme (inspired from [MAK+09]) coupled with a quadratic measure of dependence that is able to handle data sets with millions of records and thousands of dimensions. An analysis on the drawbacks of the Apriori search scheme together with a scalable search technique are given in the next chapter.

	CMI	CMI++	HiCS	ENCLUS	FB	CLIQUE	PROCLUS	DBSCAN
				WBC				
F1	**0.71**	**0.79**	0.64	**0.71**	0.48	0.67	0.49	0.61
Acc.	0.70	**0.75**	0.69	0.65	**0.74**	0.68	0.70	0.66
				Shape				
F1	0.81	**0.82**	0.79	**0.84**	0.76	0.61	0.79	0.32
Acc.	0.73	**0.79**	0.74	0.61	0.71	0.72	**0.75**	0.47
				Pendigits				
F1	**0.74**	**0.78**	0.69	0.68	0.71	0.55	0.70	0.62
Acc.	**0.72**	**0.75**	0.71	0.68	0.68	0.60	0.67	0.63
				Diabetes				
F1	**0.65**	**0.68**	0.64	0.64	0.62	0.51	0.62	0.61
Acc.	**0.65**	**0.69**	0.63	**0.65**	0.62	0.61	0.64	0.58
				Glass				
F1	**0.47**	**0.57**	0.44	0.41	0.44	0.45	0.44	0.41
Acc.	**0.55**	**0.61**	0.50	0.44	0.46	0.52	0.54	0.44

Table 3.3.: [Higher is better] Clustering results (F1 and Accuracy values) on real world data. Top two best values in each row are in **bold**. CMI++ significantly outperforms HiCS, ENCLUS, FB, CLIQUE, PROCLUS, and LOF under a Friedman test with $\alpha = 0.05$. CMI++ significantly outperforms CMI under a Wilcoxon signed rank test with $\alpha = 0.05$.

4. Subspace Mining with Quadratic Measure of Dependence

This chapter is based on our work originally published as [NMB13] and [NMB14]:

H. V. Nguyen, E. Müller, and K. Böhm, *4S: Scalable subspace search scheme overcoming traditional apriori processing*, in BigData Conference, 2013, pp. 359-367.

H. V. Nguyen, E. Müller, and K. Böhm, *A near-linear time subspace search scheme for unsupervised selection of correlated features*, Big Data Research, vol. 1, pp. 37-51, 2014.

The focus still is on mining correlated subspaces for multivariate real-valued data. However, instead of an information-theoretic correlation measure, we now propose a quadratic measure of (in)dependence for correlation assessment, which is an extension of that of [SRPP11]. Similarly to the measures proposed in the previous chapter, our measure here also is based on cumulative distribution functions (cdfs). However, its computation on empirical data is fully in closed form. With the new measure, we broaden our study to another class of correlation measure. Regarding the search for correlated subspaces, we will point out several drawbacks of the Apriori search scheme through a formal analysis of its characteristics. Hence, we depart from this search scheme and propose a novel jump search, reminiscent of the one in [MAK+09], with theoretical justifications. We will explain how our jump search alleviates the issues of the Apriori search scheme. Overall, the method covered in this chapter, named 4S for *scalable subspace search scheme*, enables a truly scalable solution towards efficient search of correlated subspaces in high dimensional data.

The road map of this chapter is as follows. In Section 4.1, we introduce the main notions used. In Section 4.2, we propose our own measure. In Section 4.3, we formally review the Apriori search scheme. In Section 4.4, we give an overview of 4S, with details on mining pairwise correlations in Section 4.5, mining higher

dimensional subspaces in Section 4.6, and subspace merge in Section 4.7. We study the speed up of 4S in Section 4.8, followed by our extensive experiments in Section 4.9. We conclude the chapter in Section 4.10.

4.1. Preliminaries

Consider a database $\mathbf{DB}$ of size N and dimensionality D. The set of dimensions is denoted as the full space $F = \{X_1, \ldots, X_D\}$. Each dimension X_i has a continuous domain $dom(X_i)$ and w.l.o.g, we assume $dom(X_i) = [-v, v] \subseteq \mathbb{R}$ with $v \geq 0$. We write $p(X_i)$ for the probability density function (pdf) of X_i. We also write $p(x_i)$ as a short form for $p(X_i = x_i)$. We let $P(X_i)$ stand for the cumulative distribution function (cdf) of X_i, and write $P(x_i)$ as a short form for $P(X_i \leq x_i)$.

A subspace S is a non-empty subset of F. Its dimensionality is written as $|S|$. The subspace lattice of $\mathbf{DB}$ consists of $D-1$ layers $\{\mathcal{L}_i\}_{i=2}^{D}$. Single dimensional subspaces are excluded since one is interested in correlations of two or more dimensions. Every layer $\mathcal{L}_i$ contains $\binom{D}{i}$ subspaces, each having i dimensions.

We aim at mining subspaces across all lattice layers whose member dimensions are highly correlated. Note that the search space is huge. For a dataspace with D dimensions the total number of possible subspaces is $O(2^D)$. For one subspace, one needs at least $O(D \cdot N)$ time to process, e.g., to compute the correlation. An overall complexity of $O(D \cdot N \cdot 2^D)$ makes brute-force search impractical. Even more sophisticated search schemes have severe scalability problems (see Section 4.3). Hence, we will propose a new scalable solution (see Section 4.4).

4.2. Correlation Measure

For correlation assessment, we here also aim at a correlation measure which facilitates computation on empirical data. In the following, we propose such a new measure. It is based on cdfs, is non-parametric (no prior assumption on the data distribution is required), and permits computation on empirical data in closed form. Our measure belongs to the class of quadratic measure of (in)dependence and is an extension of the one in [SRPP11] (specializing in pairwise correlations) to multivariate data. It is defined as follows.

Definition 11. ***Correlation Measure*** $Corr$*:*
The correlation score of $S = \{X_1, \ldots, X_d\}$ *is*

$$Corr(X_1, \ldots, X_d) = \int_{-v}^{v} \ldots \int_{-v}^{v} \left(P(x_1, \ldots, x_d) - P(x_1) \cdots P(x_d)\right)^2 dx_1 \cdots dx_d \quad .$$

Essentially, our measure quantifies the correlation of $X_1, \ldots, X_d$ by computing the squared difference between their joint cdf and the product of their marginal cdfs. This in fact is an instantiation of the general notion of correlation measure presented in Chapter 2. In particular, the joint cdf stands for the joint pdf and marginal cdfs stand for marginal pdfs. To show that our measure indeed is suitable for correlation analysis, we prove the below lemma.

Lemma 9. *$Corr(X_1, \ldots, X_d) \geq 0$ with equality iff $p(X_1, \ldots, X_d) = p(X_1) \cdots p(X_d)$.*

According to Lemma 9, our correlation measure meets Properties 1 and 2 of a good correlation measure. In addition, it does not make any assumption on the types of correlation, and hence, satisfies Challenge 1. Further, we prove that it can be computed in closed form on empirical data, i.e., it addresses Challenge 2. In particular, let $X_i(1), \ldots, X_i(N)$ be the realizations of X_i. We have:

Theorem 6. $Corr(X_1, \ldots, X_d) =$

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\prod_{k=1}^{d}(v - \max(X_k(i), X_k(j))) - \frac{2}{N^{d+1}}\sum_{i=1}^{N}\prod_{k=1}^{d}\sum_{j=1}^{N}(v - \max(X_k(i), X_k(j))) + \frac{1}{N^{2d}}\prod_{k=1}^{d}\sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(X_k(i), X_k(j))) \quad .$$

Proof. Our proof is based on [SRPP11], which has proved for the pairwise case. In particular, let $ind(\alpha)$ be an indicator function with value 1 if α is true and 0 otherwise. It holds that

$$P(a_1, \ldots, a_d) = \int_{-v}^{v} \ldots \int_{-v}^{v} ind(x_1 \leq a_1) \cdots ind(x_d \leq a_d) p(x_1, \ldots, x_d) dx_1 \cdots dx_d \quad .$$

Using empirical data, Eq. (4.2) becomes: $P(a_1, \ldots, a_d) = \frac{1}{N}\sum_{i=1}^{N}\prod_{k=1}^{d} ind(X_k(i) \leq a_k)$.

Likewise: $P(a_k) = \frac{1}{N}\sum_{i=1}^{N} ind(X_k(i) \leq a_k)$. Therefore, $Corr(X_1, \ldots, X_d)$ equals to:

$$\int_{-v}^{v} \ldots \int_{-v}^{v} \left(\frac{1}{N}\sum_{i=1}^{N}\prod_{k=1}^{d} ind(X_k(i) \leq x_k) - \prod_{k=1}^{d}\frac{1}{N}\sum_{i=1}^{N} ind(X_k(i) \leq x_k) \right)^2 dx_1 \cdots dx_d \quad .$$

Expanding Eq. (4.2), we have:

$$\int_{-v}^{v} \ldots \int_{-v}^{v} \left(\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\prod_{k=1}^{d} ind\left(\max(X_k(i), X_k(j)) \leq x_k\right) - \frac{2}{N^{d+1}}\sum_{i=1}^{N}\prod_{k=1}^{d}\sum_{j=1}^{N} ind\left(\max(X_k(i), X_k(j)) \leq x_k\right) + \frac{1}{N^{2d}}\prod_{k=1}^{d}\sum_{i=1}^{N}\sum_{j=1}^{N} ind\left(\max(X_k(i), X_k(j)) \leq x_k\right) \right) dx_1 \cdots dx_d$$

Bringing the integrals inside the sums, we obtain:

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\prod_{k=1}^{d}\int_{-v}^{v} ind\left(\max(X_k(i),X_k(j))\leq x_k\right)dx_k$$
$$-\frac{2}{N^{d+1}}\sum_{i=1}^{N}\prod_{k=1}^{d}\sum_{j=1}^{N}\int_{-v}^{v} ind\left(\max(X_k(i),X_k(j))\leq x_k\right)dx_k$$
$$+\frac{1}{N^{2d}}\prod_{k=1}^{d}\sum_{i=1}^{N}\sum_{j=1}^{N}\int_{-v}^{v} ind\left(\max(X_k(i),X_k(j))\leq x_k\right)dx_k$$

by which we arrive at the final result. □

Using Theorem 6, computing our measure $Corr$ on empirical data is straightforward. Thus, we will use $Corr$ as the correlation measure in 4S. In fact, one can also plug $Corr$ into existing subspace search schemes. However, in the next section we point out why these search schemes are not suited to big data applications.

4.3. Existing Search Schemes

Existing methods explore the search space based on the Apriori principle (APR) using a correlation measure for subspace assessment, e.g., total correlation as used in [CFZ99].

For APR, one can either keep a top number of subspaces at each layer (beam-based) or impose a threshold on the subspace correlation (threshold-based). Recently, [KMB12] point out that the beam-based scheme allows more intuitive parameterization than the threshold-based one. Thus, for better presentation, we stick to the former. However, our discussion is also applicable to the threshold-based scheme [CFZ99, KKKW03].

We illustrate the lattice exploration of APR in Figure 4.1. Its pseudocode is in Algorithm 1. APR starts at layer $\mathcal{L}_2$ (Line 1). For each layer $\mathcal{L}_i$ visited, APR computes the total correlation $T(S)$ for each candidate subspace $S \in \mathcal{L}_i$. The top $min(MAX\_NUM, \binom{D}{i})$ subspaces $CAND_i$ with the highest total correlation are selected (Lines 1 and 6). MAX_NUM is the beam size. $CAND_i$ is also used to determine which subspaces to examine in the next layer $\mathcal{L}_{i+1}$. In particular, a subspace S^{i+1} in $\mathcal{L}_{i+1}$ is considered iff all of its i-dimensional projections are in $CAND_i$ (Line 5). This is known as the monotonicity restriction, which causes redundant processing: To reach one subspace, one needs to generate and examine all of its lower dimensional projections, even though not all of them are relevant.

APR stops when either there is no more layer to explore, or the set of candidate subspaces in the current layer is empty. Assume that MAX_NUM is set such that APR reaches layer $\mathcal{L}_k$. We have:

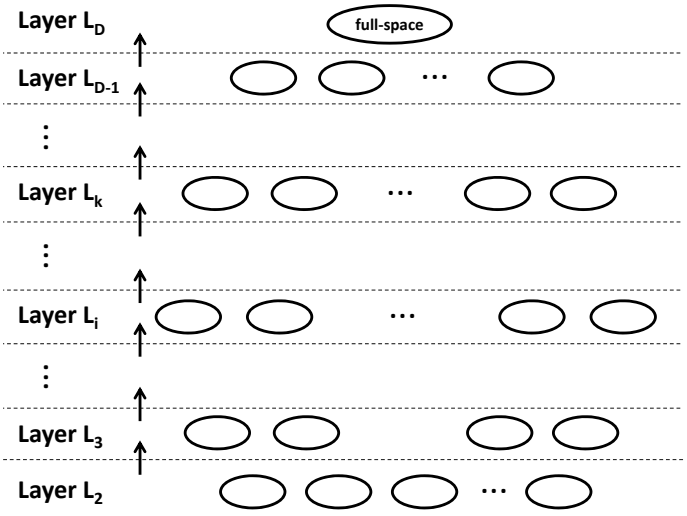


Figure 4.1.: Example showing the subspace lattice exploration of Apriori approach APR.

Algorithm 1: APR

1 $CAND_2 =$ Set of $min(MAX\_NUM, \binom{D}{2})$ subspaces in $\mathcal{L}_2$ with the highest correlation
2 $OUT = CAND_2$
3 $i = 2$
4 **while** $CAND_i \neq \emptyset$ **do**
5 $\quad CAND_{i+1} = \{S \in \mathcal{L}_{i+1} : \forall S' \subset S \wedge |S'| = i \Rightarrow S' \in CAND_i\}$
6 $\quad CAND_{i+1} =$ Top $min(MAX\_NUM, |CAND_{i+1}|)$ subspaces of $CAND_{i+1}$ with the highest correlation
7 $\quad OUT = OUT \cup CAND_{i+1}$
8 $\quad i = i + 1$
9 Return OUT

Lemma 10. *The time complexity of APR is* $O(\Delta \cdot \sum_{i=2}^{k} \binom{D}{i})$ *where* Δ *is the cost of computing the correlation of each subspace.*

Proof. For each layer $\mathcal{L}_i$ ($i \geq 2$) with $\binom{D}{i}$ subspaces, the worst case time complexity to compute the correlation for all of its subspaces is $O(\Delta \cdot \binom{D}{i})$. Thus, the overall time complexity is $O(\Delta \cdot \sum_{i=2}^{k} \binom{D}{i})$. □

Regarding Δ, we have $\Delta = \Theta(N)$ with total correlation [CFZ99], and $\Delta = \Theta(N^2)$ with our $Corr$ measure.

Since the monotonicity property imposes strict restrictions on high-level layers (i.e., high k), APR tends not to reach high dimensional subspaces. To resolve the

issue, MAX_NUM must be very large. However, this causes APR to process many candidate subspaces at each layer visited. Further, to process a subspace, APR requires to examine exponentially many lower dimensional projections to ensure that they all have high correlation. These cause its runtime to become very high. Even when MAX_NUM is kept low, APR still suffers from poor scalability due to its expensive mining of $\mathcal{L}_2$, in particular, $O(D^2 \cdot \Delta)$. Further, setting MAX_NUM to low values fails to offset the monotonicity restriction. This prevents APR from discovering high dimensional subspaces. Only lower dimensional fragments of correlated subspaces are detected. Thus, the quality of subspaces is impacted.

Another drawback of using APR is that the higher the layer visited, the more likely it is that the curse of dimensionality occurs. This is because most existing multivariate correlation measures, including our $Corr$ measure, suffer from reduction of discriminative power in high dimensional spaces—a phenomenon which has been demonstrated empirically in Chapter 3.

In summary, APR (a) is inefficient, (b) tends to miss high dimensional correlated subspaces, (c) fragments them into many redundant lower dimensional subspaces, and (d) is prone to the curse of dimensionality.

4.4. Overview of 4S Processing

We illustrate the lattice exploration of 4S in Figure 4.2 and contrast it to the APR scheme depicted in Figure 4.1. To avoid the exponential runtime in the data dimensionality, 4S does not explore the subspace lattice in a levelwise manner. Instead, 4S initially mines subspaces of high correlations in $\mathcal{L}_2$. They are then combined to directly create higher dimensional subspaces. In short, 4S works in three steps. First, we compute the correlation of each pair of dimensions and only keep the top K pairs (i.e., subspaces of $\mathcal{L}_2$) with the largest correlations. Setting K is explained in Section 4.6.

Second, we construct an undirected correlation graph $\mathcal{G}_\mathcal{D}$ representing our search space of subspaces. Its nodes are the dimensions, connected by an edge iff their correlation is in the top K values. Following our new notion of correlated subspace, we apply the algorithm in [ELS10] to mine maximal cliques of this correlation graph. The maximal cliques serve as candidate subspaces. We also prove that these candidate subspaces are likely mutually correlated. The toy example in Figure 4.3 displays a correlation graph for a 10-dimensional data set. There are 45 possible subspaces in $\mathcal{L}_2$; $K = 10$ of which are picked to construct $\mathcal{G}_\mathcal{D}$. From $\mathcal{G}_\mathcal{D}$, 4S finds three maximal cliques (subspaces): $S_1 = \{1, 2, 3, 4\}$, $S_2 = \{1, 3, 4, 5\}$, and $S_3 = \{7, 8\}$.

Third, mining maximal cliques on $\mathcal{G}_\mathcal{D}$ may also produce subspaces that are projections of the same subspaces due to the restriction on pairwise correlations (i.e., through K). For instance, in Figure 4.3, dimension 5 is connected to all dimensions in S_1 except for dimension 2. This leads to the detection of two separate subspace fragments S_1 and S_2 that have high overlap with each other. It would

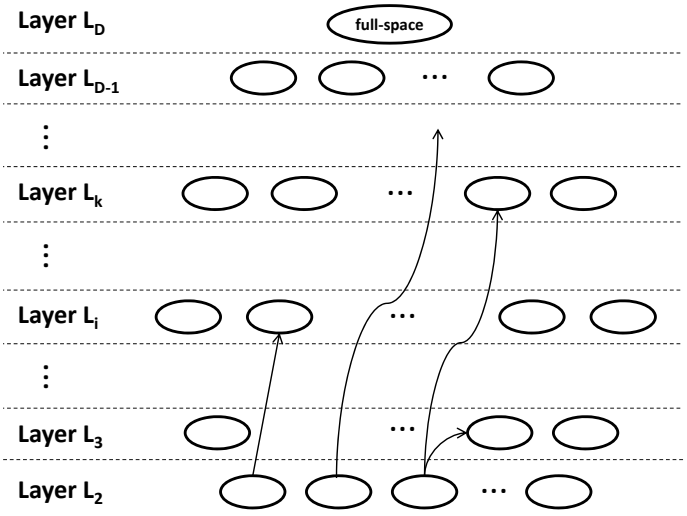


Figure 4.2.: Example showing the subspace lattice exploration of 4S.

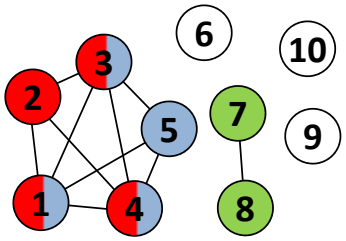


Figure 4.3.: Example of correlation graph.

make sense to merge S_1 and S_2 to create the larger subspace $\{1, 2, 3, 4, 5\}$. This also helps us to cope with real-world data where perfect pairwise correlation between dimensions of correlated subspaces may not always be fulfilled. Thus, we propose to merge similar subspaces using the MDL-based approach in [MV13]. Following this step, we obtain higher dimensional subspaces with minimal redundancy.

Overall, in contrast to APR, we can reach high dimensional correlated subspaces with our scalable search scheme, which consists of: (a) scalable computation of $\mathcal{L}_2$, (b) scalable mining of $\mathcal{L}_k$ with $k > 2$, and (c) subspace merge. While APR needs to impose the Apriori monotonicity restriction on all layers for the sake of efficiency, we only require that dimensions of subspaces are pairwise correlated (i.e., restriction on $\mathcal{L}_2$), which facilitates a jump search strategy.

To this end, we note that our jump search is related to that of DensEst [MAK+09]. In particular, DensEst mines multivariate regions of high density by (a) collecting density statistics of 2-dimensional histograms and (b) jumps directly to such regions using the 2-dimensional statistics. The second step is done by first constructing an independence graph of dimensions, using χ^2 test as the correlation measure. Then, based on this independence graph DensEst factorizes the joint distribution into 2-dimensional distributions. Hence, DensEst is able to estimate the density of each multivariate region based on the densities of its 2-dimensional

projections. Our goal in this chapter is different from that of DensEst. More specifically, instead of mining high density regions, we in turn mine correlated subspaces by collecting statistics on pairwise correlations. DensEst thus can be perceived as a local approach while ours is a global approach. Further, as DensEst relies on χ^2 test which is for discrete/categorical data, its success on real-valued data highly depends on the discretization technique used. Choosing the right one w.r.t. χ^2 test and multivariate density estimation, however, is a non-trivial issue. Hence, with naïve discretization DensEst may fall prey to unreliable performance, like ENCLUS [CFZ99]. Our method 4S in contrast can avoid this issue.

There is another remark that we want to highlight regarding 4S. That is, the subspace search under our problem transformation (i.e., after the computation of pairwise correlations in $\mathcal{L}_2$) is NP-hard. We note that the input of the subspace search problem for database $\mathbf{DB}$ consists of: (a) the set of all dimensions $\mathcal{F}$, and (b) the set of dimension pairs $\mathcal{P} \subseteq \mathcal{F} \times \mathcal{F}$ with the largest correlations to be kept. We prove the NP-hardness of the subspace search problem by giving a polynomial reduction of the classic Maximal Cliques Mining (MCM) problem, which is a NP-hard problem [JP09], to subspace search (SS): MCM $\leq_p$ SS.

Theorem 7. *The subspace search problem under the setting of* 4S *is NP-hard.*

Proof. First, we map the input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of MCM to an input $(\mathcal{F}, \mathcal{P})$ of SS. The mapping is straightforward: For each node $t \in \mathcal{V}$, we add a dimension X_t to the set of dimensions $\mathcal{F}$. After that, for each edge $e = (t, t') \in \mathcal{E}$, we add the pair $(X_t, X_{t'})$ into $\mathcal{P}$. Obviously, our mapping is done in polynomial time. Now we have to show that the subspace search result (under our notion of correlated subspace) on $(\mathcal{F}, \mathcal{P})$ corresponds to a solution of MCM in $\mathcal{G}$. This is easy to show as by our convention, each correlated subspace $S \subseteq \mathcal{F}$ has to satisfy:

(a) $\forall (X_t, X_{t'}) \subset S \Rightarrow (X_t, X_{t'}) \in \mathcal{P}$

(b) no proper superset of S is a correlated subspace, i.e., S is maximal

In other words, S represents a maximal clique of $\mathcal{G}$. Thus, solving SS for the constructed instance $(\mathcal{F}, \mathcal{P})$ leads to a valid solution of MCM for $\mathcal{G}$. □

Following Theorem 7, without making any assumption on the structure of the search space, i.e., the correlation graph $\mathcal{G}_\mathcal{D}$, SS is NP-hard. To overcome the issue, we instead impose restrictions on $\mathcal{G}_\mathcal{D}$. In particular, we heuristically make $\mathcal{G}_\mathcal{D}$ sparse by implicitly limiting the maximal degree of its nodes. We accomplish this by carefully setting K (more details are in Section 4.6). By enforcing $\mathcal{G}_\mathcal{D}$ to be sparse, we are able to apply exact and efficient algorithms [JP09], which have good performance on sparse graphs. Next, we introduce the details of 4S (see Sections 4.5–4.8), including our analysis showing that 4S reliably identifies correlated subspaces and is more general than APR in Section 4.6. We empirically show that 4S produces subspaces of higher quality than existing methods in Section 4.9.

4.5. Scalable Computation of $\mathcal{L}_2$

In $\mathcal{L}_2$, we need to compute the correlation score of all pairs of dimensions. To this end, for two dimensions X and Y, we have:

Lemma 11. $Corr(X,Y) =$

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(v-\max(x_i,x_j))(v-\max(y_i,y_j))$$
$$-\frac{2}{N^3}\sum_{i=1}^{N}\left(\sum_{j=1}^{N}(v-\max(x_i,x_j))\right)\left(\sum_{j=1}^{N}(v-\max(y_i,y_j))\right)$$
$$+\frac{1}{N^4}\sum_{i=1}^{N}\sum_{j=1}^{N}(v-\max(x_i,x_j))\sum_{i=1}^{N}\sum_{j=1}^{N}(v-\max(y_i,y_j)) \quad .$$

Proof. The proof can be obtained by either applying Theorem 6 for $d = 2$, or by following that in [SRPP11] which specializes in the pairwise case. □

Following Lemma 11, we need to compute three terms, referred to as T_1, T_2, and T_3, and

$$Corr(X,Y) = \frac{1}{N^2}T_1 - \frac{2}{N^3}T_2 + \frac{1}{N^4}T_3 \quad .$$

To compute $Corr(X,Y)$, we need $O(N^2)$ time. For D-dimensional data sets, the total runtime required to explore layer $\mathcal{L}_2$ becomes $O(D^2N^2)$. This is a serious problem for any data set. To tackle the issue, we introduce two new approaches, *MultiPruning* and *Sketching*, to boost efficiency regarding both N and D. *MultiPruning* calculates the exact correlation. However, it still has issues regarding efficiency for large data sets. *Sketching* in turn trades accuracy for efficiency. Yet it is still better than APR (see Section 4.9). Note that $Corr$ deploys the same estimator as other quadratic measures of independence [RSX+11], such as [Ach08, SGSS07]. The difference only lies in different kernels employed. Thus, the ideas of *MultiPruning* and *Sketching* are also applicable to other measures of the same category. In other words, our method is not limited to one correlation measure.

4.5.1. MultiPruning

MultiPruning aims at reducing the runtime by applying pruning rules for $Corr(X,Y)$ based on two upper bounds of T_1. It uses the fact that we only keep the top K pairs of dimensions with the largest correlation. Let $\{(x_{s(i)}, y_{s(i)})\}_{i=1}^{N}$ be $\{(x_i,y_i)\}_{i=1}^{N}$ sorted in descending order w.r.t. X. The upper bounds of T_1 are as follows.

Theorem 8. [CAUCHY-SCHWARZ BOUND]

$$T_1 \leq \sum_{i=1}^{N}\sqrt{\sum_{j=1}^{N}(v-\max(x_i,x_j))^2\sum_{j=1}^{N}(v-\max(y_i,y_j))^2} \quad .$$

Proof. Applying the Cauchy-Schwarz inequality, we have that for each $i \in [1, N]$:

$$\left(\sum_{j=1}^{N}(v - \max(x_i, x_j))(v - \max(y_i, y_j))\right)^2$$
$$\leq \sum_{j=1}^{N}(v - \max(x_i, x_j))^2 \sum_{j=1}^{N}(v - \max(y_i, y_j))^2 \quad .$$

Taking the square root for each side of Eq. (4.5.1) and summing up over all $i \in [1, N]$, we arrive at the final result. □

To derive the second bound, we observe that in T_1, each term $(v - \max(x_i, x_j))(v - \max(y_i, y_j))$ indeed appears twice. This can be avoided by sorting, for instance, with respect to the values of X. This leads to another formula of T_1, as follows:

Lemma 12. *It holds that*

$$T_1 = \sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)}) + 2\sum_{i=1}^{N}(v - x_{s(i)}) \sum_{j=i+1}^{N}(v - \max(y_{s(i)}, y_{s(j)})) \quad .$$

Proof. We have:

$$\begin{aligned}
T_1 &= \sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(x_i, x_j))(v - \max(y_i, y_j)) \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(x_{s(i)}, x_{s(j)}))(v - \max(y_{s(i)}, y_{s(j)})) \\
&= \sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)}) \\
&\quad + 2\sum_{i=1}^{N}\sum_{j=i+1}^{N}(v - \max(x_{s(i)}, x_{s(j)}))(v - \max(y_{s(i)}, y_{s(j)})) \\
&= \sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)}) + 2\sum_{i=1}^{N}(v - x_{s(i)}) \sum_{j=i+1}^{N}(v - \max(y_{s(i)}, y_{s(j)})) \quad .
\end{aligned}$$

□

We can see that after sorting with respect to X, for each $(x_{s(i)}, y_{s(i)})$, we only need to compute $(v - \max(x_{s(i)}, x_{s(j)}))(v - \max(y_{s(i)}, y_{s(j)}))$, which in fact equals to $(v - x_{s(i)})(v - \max(y_{s(i)}, y_{s(j)}))$, for $x_{s(j)} \leq x_{s(i)}$. Lemma 12 leads to the following theorem on the second upper bound of T_1.

Algorithm 2: Computing the statistics of the Cauchy-Schwarz bound on X

$\{(x_{s(1)}, org\_pos_1), \ldots, (x_{s(N)}, org\_pos_N)\} \leftarrow$ Sort $\{x_1, \ldots, x_N\}$ in descending order
$sum = 0$
for $i = 1 \rightarrow N$ **do**
  $ret(org\_pos_i) = sum + (v - x_{s(i)})^2 \cdot (N - i + 1)$
  $sum = sum + (v - x_{s(i)})^2$
Return $\{ret(1), \ldots, ret(N)\}$

Theorem 9. [SORTED-BASED BOUND]

$$T_1 \leq \sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)}) + 2\sum_{i=1}^{N}(v - x_{s(i)}) \sum_{j=i+1}^{N}(v - y_{s(j)}) \quad .$$

Proof. The proof is derived directly from Lemma 12 and the fact that

$$v - \max(y_{s(i)}, y_{s(j)}) \leq v - y_{s(j)} \quad .$$

□

The statistics required for the Cauchy-Schwarz bound, e.g., $\sum_{j=1}^{N}(v - \max(x_i, x_j))^2$ for $1 \leq i \leq N$, can be pre-computed for each dimension in $O(N \log N)$ time. This is from our observation that $\sum_{j=1}^{N}(v - \max(x_i, x_j))^2 = \sum_{x_j \geq x_i}(v - x_j)^2 + \sum_{x_j < x_i}(v - x_i)^2 = \sum_{x_j \geq x_i}(v - x_j)^2 + (v - x_i)^2 \cdot |\{x_j : x_j < x_i\}|$. That is, for each dimension, we first sort its data in descending order. Then, we loop through the data *once* in that order and pre-compute the required statistics. We illustrate our point by giving in Algorithm 2 a sample pseudocode, which computes the statistics of the Cauchy-Schwarz bound on X. We note that org_pos_i stands for the original position (before sorting) of $x_{s(i)}$. A corresponding numerical example is described below.

Example 2. *Consider three data points $P_1 = (1, -1)$, $P_2 = (-1, 1)$, and $P_3 = (0, 0)$ (i.e., $dom(X) = [-1, 1]$). To compute the statistics for X, we sort X in descending order and obtain $\{1, 0, -1\}$. Then, we compute $\sum_{j=1}^{3}(1 - \max(x_i, x_j))^2$ ($1 \leq i \leq 3$) by looping through the sorted list once and obtain: 0 for P_1, 5 for P_2, and 2 for P_3. Similarly, for $\sum_{j=1}^{3}(1 - \max(y_i, y_j))^2$ ($1 \leq i \leq 3$), we obtain: 5 for P_1, 0 for P_2, and 2 for P_3. We calculate the Cauchy-Schwarz bound by looping through the stored statistics once, and achieve: $\sqrt{0 \cdot 5} + \sqrt{5 \cdot 0} + \sqrt{2 \cdot 2} = 2$.*

The statistics required to *exactly* compute the second term T_2 of $Corr(X, Y)$, which is $\sum_{j=1}^{N}(v - \max(x_i, x_j))$ for $1 \leq i \leq N$, can be pre-computed similarly. The statistics

of the third term T_3, which is $\sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(x_i, x_j))$, is also computed during this phase by incrementally summing up the statistics of T_2 (per dimension).

During the pairwise correlation computation, we maintain the top K values seen so far. For a new pair of dimensions (X, Y), we first compute the bounds. This computation is in $O(N)$ (see Algorithm 2). Similarly, the exact value of the second term T_2 is computed. Similarly to Algorithm 2, we obtain the sorted-based bound in Theorem 9 in $O(N)$ time. The details are as follows. We loop through the data sorted w.r.t. X. For each point $(x_{s(i)}, y_{s(i)})$, we compute $(v - x_{s(i)})(v - y_{s(i)})$ and $(v - x_{s(i)})\sum_{j=i+1}^{N}(v - y_{s(j)})$. We do so by taking into account that $\sum_{j=i+1}^{N}(v - y_{s(j)}) = \sum_{j=1}^{N}(v - y_{s(j)}) - \sum_{j=1}^{i}(v - y_{s(j)})$. That is, $\sum_{j=i+1}^{N}(v - y_{s(j)})$ can be incrementally computed using the data values encountered so far in the loop.

The sorted-based bound can also be computed w.r.t. Y. So in fact, we have two versions of this bound, one for X and one for Y. The exact value of T_3 is computed in just $O(1)$ time using its pre-computed statistics, which are $\sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(x_i, x_j))$ and $\sum_{i=1}^{N}\sum_{j=1}^{N}(v - \max(y_i, y_j))$.

If any upper bound of $Corr(X, Y)$ is less than the K^{th} largest value so far, we can safely stop computing its actual value. Otherwise, we compute T_1, and hence, $Corr(X, Y)$ (and update the top K correlation values), using Lemma 12. That is, for each $x_{s(i)}$, we search for $y_{s(i)}$ in the list of values of Y sorted in descending order. For each value $y > y_{s(i)}$ encountered, we add $2(v - x_{s(i)})(v - y)$ to T_1. Once $y_{s(i)}$ is found, the search stops. Suppose that the position found is p, and the list has e elements. We add $2(e - p + 1)(v - x_{s(i)})(v - y_{s(i)})$ to T_1. We remove $y_{s(i)}$ from the list and proceed to $x_{s(i+1)}$. This helps us to avoid scanning the whole list and, hence, reduces the runtime. We note that $\sum_{i=1}^{N}(v - x_{s(i)})(v - y_{s(i)})$ is already computed during the sorted-based bound computation.

By means of pruning rules and efficient computation heuristics, *MultiPruning* is able to achieve efficiency in practice. However, as there is no theoretical guarantee on the effectiveness of the pruning rules, the worst-case complexity of *MultiPruning* is still $O(D^2N^2)$. This motivates us to introduce *Sketching*, which trades accuracy for further improvement in scalability.

4.5.2. Sketching

To better address the scalability issue (i.e., quadratic in N), we propose *Sketching* as an alternative solution. First, we see that T_3 is computed in only $O(1)$ time using its pre-computed statistics. Thus, our main intuition is to convert the terms T_1 and

T_2 to forms similar to that of T_3. We observe that T_1 and T_2 can be perceived as dot products of vectors. In particular, T_1 is the product of vectors

$$(v - \max(x_1, x_1), \ldots, v - \max(x_1, x_N), \ldots, v - \max(x_N, x_1), \ldots, v - \max(x_N, x_N))$$

and

$$(v - \max(y_1, y_1), \ldots, v - \max(y_1, y_N), \ldots, v - \max(y_N, y_1), \ldots, v - \max(y_N, y_N)) \quad .$$

Likewise, T_2 is the product of vectors

$$\left(\sum_{j=1}^{N} (v - \max(x_1, x_j)), \ldots, \sum_{j=1}^{N} (v - \max(x_N, x_j)) \right)$$

and

$$\left(\sum_{j=1}^{N} (v - \max(y_1, y_j)), \ldots, \sum_{j=1}^{N} (v - \max(y_N, y_j)) \right) \quad .$$

Vector products in turn can be efficiently estimated by AMS Sketch [AMS96]. AMS Sketch provides rigorous theoretical bounds for its estimation and can outperform other sketching schemes [RD08]. However, to our knowledge, we are first to use this theory to efficiently compute pairwise correlations of continuous random variables.

Our general idea is to use AMS Sketch to derive unbiased estimators of T_1 and T_2 that have forms similar to T_3. The estimators are unbiased since their expected values equal to their respective true values. We will prove that the estimators are close to the true values of T_1 and T_2, respectively. Overall, *Sketching* reduces the time complexity of computing $Corr(X, Y)$ to $O(N \log N)$.

Sketching approximates $Corr(X, Y)$ through unbiased estimators by projecting X and Y onto random 4-wise independent vectors. Let $u, w \in \{\pm 1\}^N$ be two such vectors which are independent to each other. We estimate T_1 as follows:

Theorem 10. *Let Z be a random variable that equals to*

$$\sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(x_i, x_j)) u_i w_j \sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(y_i, y_j)) u_i w_j$$

then $E(Z) = T_1$ and $Var(Z) \leq 8[E(Z)]^2$.

Likewise, we estimate T_2 as:

Theorem 11. *Let W be a random variable that equals to*

$$\sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(x_i, x_j)) u_i \sum_{i=1}^{N} \sum_{j=1}^{N} (v - \max(y_i, y_j)) u_i$$

then $E(W) = T_2$ and $Var(W) \leq 2[E(W)]^2$.

We derive Theorems 10 and 11 based on [AMS96]. These theorems tell us that T_1 and T_2 can be approximated by unbiased estimators, i.e., Z and W respectively. Further, these estimators have forms similar to that of T_3. Hence, $Corr(X, Y)$ can be approximated in $O(N \log N)$ time by pre-computing the statistics required in a way similar to *MultiPruning*. Please note that, we also need to ensure that the estimators concentrate closely enough around their respective mean. To accomplish this, we apply Chebychev's inequality. The variance of Z is upper-bounded by $8[E(Z)]^2$. By averaging over s_1 different values of u and w, the variance is reduced to at most $\frac{8[E(Z)]^2}{s_1}$. Using Chebychev's inequality, we have

$$P(|Z - E(Z)| > \epsilon E(Z)) \leq \frac{8}{s_1 \epsilon^2} \quad .$$

That is, the probability of high estimation error can be upper bounded, i.e., estimation error can be controlled probabilistically. In fact, if we repeat the averaging $s_2 = O(1/\delta)$ times and take the median of these averages, the relative error of Z w.r.t. $E(Z)$ is at most ϵ with probability at least $1 - \delta$, as proven in [AMS96].

Similarly, by averaging over s_1 different values of u, the variance of W is reduced to at most $\frac{2[E(W)]^2}{s_1}$. Applying Chebychev's inequality, we have

$$P(|W - E(W)| > \epsilon E(W)) \leq \frac{2}{s_1 \epsilon^2} \quad .$$

We again boost the estimation accuracy by repeating the averaging $s_2 = O(1/\delta)$ times.

Sorting all dimensions costs $O(DN \log N)$. For each random vector and each dimension, it costs the same amount of time as that of T_3 to pre-compute the statistics, which is $O(N)$ (see Section 4.5.1). For all vectors and all dimensions, the total cost of pre-computing statistics is $O(s_1 s_2 DN)$. Since $s_1 s_2$ must be large enough to guarantee estimation accuracy, the cost of pre-computing statistics dominates that of data sorting. For each pair of dimensions, the cost to calculate its (estimated) correlation is $O(s_1 s_2)$. Thus, computing the correlations for all dimension pairs and maintaining the top values cost $O(s_1 s_2 D^2 + D^2 \log K)$, with $O(s_1 s_2 D^2)$ dominating. Therefore, the total time complexity of *Sketching* is $O(s_1 s_2 DN + s_1 s_2 D^2)$. In our experiments, $D < N$, i.e., the time complexity becomes $O(s_1 s_2 DN)$, a considerable improvement from $O(D^2 N^2)$. We note that the factor $s_1 s_2$ does not contribute much to the overall runtime, and in practice *Sketching* scales linearly in both N and D.

4.6. Scalable Mining of $\mathcal{L}_k$

Based on the set of 2-dimensional subspaces found in $\mathcal{L}_2$, denoted as $\mathcal{S}_2$, we now explain how to mine subspaces in higher-level layers. According to our notion, a subspace has a high correlation if its member dimensions are all pairwise correlated. We now point out that subspaces fulfilling our notion likely have a high total

correlation. We also formally prove that our new notion of correlated subspace is more general than that of APR. That is, given the same correlation measure, all subspaces found by APR are also discovered by our mining scheme. Further, we will demonstrate empirically later on that, with our notion, 4S produces better subspaces than APR. First, let us consider a subspace S with all pairs $\{X_i, X_j\} \in \mathcal{S}_2$. W.l.o.g., assume that $S = \{X_1, \ldots, X_d\}$. We have:

Lemma 13. *The total correlation score of $X_1, \ldots, X_d$ (i.e., of S) is lower-bounded by*

$$T(X_1, \ldots, X_d) \geq \sum_{i=2}^{d} H(X_i) - H(X_i \mid X_{i-1}) \quad .$$

Proof. As conditioning reduces entropy [CT06], we have:

$$H(X_i \mid X_{i-1}) \geq H(X_i \mid X_1, \ldots, X_{i-1}) \quad .$$

Using Definition 4, we arrive at the result. □

That is, the total correlaiton score of S is lower bounded by the mutual information scores of its dimension pairs. By definition, every pair $\{X_{i-1}, X_i\} \in \mathcal{S}_2$ has a high correlation. Following Definition 11, this means that $P(X_{i-1}, X_i)$ and $P(X_{i-1})P(X_i)$ deviate from each other. Thus, the joint density function $p(X_{i-1}, X_i)$ of X_{i-1} and X_i deviates from the product of their marginal density functions, which is $p(X_{i-1})p(X_i)$ [SRPP11]. Consequently, $H(X_i) - H(X_i \mid X_{i-1})$, which equals to the Kullback-Leibler divergence of $p(X_{i-1}, X_i)$ and $p(X_{i-1})p(X_i)$, is high. Based on Lemma 13, we conclude that: $T(X_1, \ldots, X_d)$ is high. Lemma 13 also holds for any permutation of $X_1, \ldots, X_d$. Hence, under any permutation of the dimensions of S, S has a high total correlation. This also means: The difference between the joint density function of S and the product of its marginal density functions is high w.r.t. the Kullback-Leibler divergence. Hence, subspaces fulfilling our notion likely are mutually correlated, not just pairwise correlated. Since correlation measures define mutual correlation based on the difference between the joint distribution and the product of marginal distributions (see Chapter 2), our subspaces are also likely mutually correlated under other correlation measures.

We now prove that our new notion of correlated subspace is more general than that of APR:

Theorem 12. *Let S be a subspace detected by APR using $Corr$ as correlation measure and given $MAX\_NUM \leq K$, then all of its pairs $\{X_i, X_j\} \in \mathcal{S}_2$.*

Proof. We use induction:
Let $S = \{X_1, \ldots, X_d\}$ be a subspace mined by APR.

Basis: When $d = 2$, since $MAX\_NUM \leq K$, we have that $S \in \mathcal{S}_2$.

Hypothesis: Suppose that Theorem 12 holds for $d = n \geq 2$.

Inference: We prove that Theorem 12 also holds for $d = n + 1$, i.e., we prove $\forall X_i \neq X_j \in S : \{X_i, X_j\} \in \mathcal{S}_2$. This is straightforward. For $X_i \neq X_j$, there exists an n-dimensional subspace $U \subset S$ such that $X_i, X_j \in U$ and U is included by APR in the output (cf., monotonicity property). Hence, $\{X_i, X_j\} \in \mathcal{S}_2$ according to the hypothesis. □

Theorem 12 also holds for other correlation measures, e.g., the ones in [CFZ99, KMB12] or in Chapter 3, with $\mathcal{S}_2$ being formed according to the measure used. It implies that, given the same correlation measure and $MAX\_NUM \leq K$, all subspaces included in the final output of APR are also discovered by our mining scheme. This is because any two of their dimensions are pairwise correlated, i.e., they form cliques in the correlation graph. This shows that our mining scheme is more general than APR and, hence, can discover subspaces missed by APR. Note that a subspace satisfying the pairwise condition is not necessarily included in the final output of APR. Also, the monotonicity restriction imposed by APR is only to reduce the runtime, and does not guarantee the quality of subspaces. Our empirical study also confirms this.

Having formally analyzed the theoretical properties of our notion of correlated subspace, we now map the problem of mining subspaces in higher-level layers to maximal clique mining in the correlation graph. Consider an undirected correlation graph $\mathcal{G}_\mathcal{D}$ with nodes being the dimensions. An edge exists connecting two dimensions X_i and X_j iff $\{X_i, X_j\} \in \mathcal{S}_2$. A subspace of our interest then forms a clique in $\mathcal{G}_\mathcal{D}$. To avoid redundancy, we propose to mine only maximal cliques, i.e., subspaces are not completely contained in each other. To efficiently find all maximal cliques in $\mathcal{G}_\mathcal{D}$, we apply the algorithm proposed in [ELS10]. We regard maximal cliques of $\mathcal{G}_\mathcal{D}$ as the result of this step.

Interestingly, our consideration of maximal cliques in $\mathcal{G}_\mathcal{D}$ fits with undirected graphical modeling [WJ08]. In particular, let us denote $\mathcal{C}$ as the set of maximal cliques of $\mathcal{G}_\mathcal{D}$. Under undirected graphical modeling, the joint distribution of $X_1, \ldots, X_D$ is factorized as follows:

$$p(X_1, \ldots, X_D) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(C)$$

where Z is the normalization factor and ψ_C is the *compatibility function* of the dimensions in C. That is, the undirected graphical model $\mathcal{G}_\mathcal{D}$, also known as a *Markov random field*, factorizes the joint distribution of all dimensions into the product of the distributions of the maximal cliques. Thus, the maximal cliques of $\mathcal{G}_\mathcal{D}$ are very important for large-scale analysis of $p(X_1, \ldots, X_D)$ [WJ08]. This in turn supports our design choice in mining these maximal cliques as candidate subspaces.

Given D dimensions, the worst-case complexity to find all maximal cliques is $O(3^{D/3})$. To ensure the practicality of 4S, we need a way to construct $\mathcal{G}_\mathcal{D}$ which can help to reduce the runtime. To achieve this, we rely on a recent finding [AC10]. It states that the properties of a data set (e.g., distances between data points) are

preserved after dimensionality reduction as long as the number of dimensions kept is $O(\log N)$. As a result, we set $K \leq D \log N$, i.e., $O(D \log N)$. Hence, the expected maximal degree of each node in $\mathcal{G}_\mathcal{D}$ is $O(\log N)$, i.e., each dimension can be part of subspaces (maximal cliques) with expected maximal dimensionality $O(\log N)$. This implies that the expected degeneracy of $\mathcal{G}_\mathcal{D}$ is $O(\log N)$. Following [ELS10], we obtain the following result:

Theorem 13. *The expected time complexity of mining maximal cliques is* $O(DN^{1/3} \log N)$. *The expected number of maximal cliques is* $O((D - \log N)N^{1/3})$.

According to Theorem 13, our strategy of constructing $\mathcal{G}_\mathcal{D}$ enables us to efficiently and directly mine high dimensional subspaces with reduced knowledge loss. Further, we achieve this without traversing the subspace lattice in a levelwise manner. Note that our scheme is different from approaches imposing the maximal dimensionality of subspaces. This is because the maximal dimensionality is *implicitly* embedded in 4S (by setting K), rather than explicitly. Further, 4S is not constrained by the $O(\log N)$ bound in practice. This is due to the MDL-based merge of subspaces described next, which reconstructs high dimensional correlated subspaces from fragments.

4.7. Subspace Merge

We denote the set of dimensions, each belonging to at least one maximal clique, as $\{X_{r(j)}\}_{j=1}^{l}$. Also, $\{C_i\}_{i=1}^{m}$ is the set of maximal cliques. Due to the pairwise restriction of our subspace notion, subspaces (maximal cliques) obtained by mining $\mathcal{G}_\mathcal{D}$ may be projections of the same higher-dimensional correlated subspaces (see Figure 4.3). To reconstruct such subspaces and to remove redundancy in the output, we merge subspaces into groups such that the new set of subspaces *guarantees completeness and minimizes redundancy*. To accomplish this, we first construct a binary matrix $\mathcal{B}$ with l rows and m columns. The rows are dimensions, and the columns are cliques. $\mathcal{B}_{ij} = 1$ iff X_i is in C_j, and 0 otherwise.

Example 3. *The binary data set* $\mathcal{B}$ *of the example in Figure 4.3 is as follows:*

		Columns		
		S_1	S_2	S_3
Rows	X_1	1	1	0
	X_2	1	0	0
	X_3	1	1	0
	X_4	1	1	0
	X_5	0	1	0
	X_7	0	0	1
	X_8	0	0	1

The columns of $\mathcal{B}$ *correspond to the three subspaces detected. The cells are colored according to the color of its respective subspace.*

We transform the subspace merge to grouping similar columns of $\mathcal{B}$, each final group constituting one subspace. We aim at achieving the task without having to define any distance function among the dimensions of $\mathcal{B}$. Thus, we apply the merge technique proposed in [MV13] which uses the Minimum Description Length (MDL) principle. This technique is as follows.

Given a set of models $\mathcal{M}$, MDL identifies the best model $M \in \mathcal{M}$ as the one that minimizes

$$L(\mathcal{B}, M) = L(M) + L(\mathcal{B} \mid M) \quad .$$

Here, $L(M)$ is the length in bits of the description of the model M, and $L(\mathcal{B} \mid M)$ is the length of the description of the data $\mathcal{B}$ encoded by M. That is, MDL helps select a model that yields the best balance between goodness of fit and model complexity.

In our problem, each model is a grouping of maximal cliques $\{C_i\}_{i=1}^m$. Each candidate grouping $G = \{A_1, \ldots, A_k\}$ under consideration is a partitioning of $\{C_i\}_{i=1}^m$, i.e., it must satisfy three properties: (a) $\bigcup_{i=1}^k A_i = \{C_i\}_{i=1}^m$, (b) for $i \neq j$: $A_i \cap A_j = \emptyset$, and (c) for every i: $A_i \neq \emptyset$. To find the optimal grouping G, we need to formulate $L(\mathcal{B}, G)$ to enable the optimization process. To this end, we rely on [MV13] for this derivation.

In particular, each $A_i \in G$ can be described by a *code table* CT_i. This table has an entry for each possible value a from $dom(A_i)$. The left-hand column of CT_i contains the value, and the right-hand column contains the corresponding code (the code assigned by MDL encoding). The frequency of $a \in CT_i$ is defined as its support relative to the number of rows l of the binary data set $\mathcal{B}$: $fr(A_i = a) = supp(A_i = a)/l$. The total encoding cost $L(\mathcal{B}, G)$ is given as [MV13]:

Definition 12. ***Total Encoding Cost**:*
Let a grouping $G = \{A_1, \ldots, A_k\}$ be given. We have: $L(\mathcal{B}, G) = L(G) + L(\mathcal{B} \mid G)$, where

- $L(\mathcal{B} \mid G) = l \sum_{i=1}^k H(A_i)$,
- $L(G) = \log B_n + \sum_{i=1}^k L(CT_i)$,
- $L(CT_i) = \sum_{a \in dom(A_i)} |A_i| + \log\log l - \log fr(A_i = a)$

with $fr(A_i = a) \neq 0$, and B_n being the Bell number.

Definition 12 formulates all components that constitute the total encoding cost $L(\mathcal{B}, G)$, giving way for us to mine the grouping G that minimizes it. Note that the search space is $O(2^m)$ and unstructured. Thus, a heuristic algorithm is employed [MV13]. This algorithm starts with each attribute forming its own group. Then, it progressively picks two groups whose merge leads to the largest reduction in the total encoding cost, and it merges them. This practice indeed ensures that two most similar groups are merged at each step. The algorithm terminates when either there are no more groups to merge, or when the current step does not reduce the total encoding cost any more. The result below follows:

Theorem 14. *The subspace merge guarantees completeness and minimizes redundancy.*

That is, using the technique in [MV13], we ensure that the merge process outputs subspaces containing information of all the subspaces produced by the second step (completeness). This stems from the fact that MDL guarantees a *lossless compression* [Grü07]. Thus, the original set of subspaces is compressed while guaranteeing that no information loss occurs. Besides, the heuristic algorithm of [MV13] selects the grouping of subspaces that minimizes the overall compression cost. For instance, if a grouping contains two very similar subspaces (i.e., redundant ones), the algorithm would not pick it since the merge of two subspaces can result in a better grouping with a lower encoding cost. Hence, redundancy is minimized.

According to [MV13], the total time complexity of this step is $O(lm^3)$, which is $O((D - \log N)^3 lN)$. Nevertheless, the runtime in practice is much smaller because (a) the number of cliques is much smaller than the one stated in Theorem 13, (b) the number l of dimensions left is small compared to D, and (c) the merge algorithm tends to terminate early. Our experiments also point out that the runtime of this step is negligible compared to the first step. While APR can also apply this subspace merge, it does not achieve the same quality as 4S since it hardly ever reaches high dimensional subspaces.

4.8. Overall Complexity Analysis

Table 4.1 summarizes the time complexity reduction achieved by 4S for each step. The computation of $\mathcal{L}_2$ (using *Sketching*) costs $O(DN)$. The mining of $\mathcal{L}_k$ costs $O(DN^{1/3} \log N)$. The subspace merge costs $O((D - \log N)^3 lN)$. Thus, the worst-case complexity of 4S is $O((D - \log N)^3 lN)$. However, our experiments point out that the most time-consuming step is the computation of $\mathcal{L}_2$, which accounts for nearly 90% of the overall runtime. Hence, overall, we for our part conclude that 4S has $O(DN)$ average-case complexity. Our experiments also confirm that 4S has near-linear scalability with both N and D.

Considering the high time complexity of APR where k is the highest layer reached (see Lemma 10), one can see that the speed-up 4S achieves is a significant improvement: It enables a near-linear time heuristic search to explore an exponential search space. Further, it eliminates the efficiency bottleneck of exploring $\mathcal{L}_k$. Nevertheless, we note that 4S does not push the envelop. That is, from the linear scalability of 4S, there is in fact still much room of improvement towards a truly scalable solution for big data. For instance, we could use parallelization: Parallelizing the computation in $\mathcal{L}_2$ is straightforward; parallelizing the search in $\mathcal{L}_k$ can be done by using special techniques, such as [SSTP09].

4.9. Experiments

We write 4S-M and 4S-S as 4S with *MultiPruning* and *Sketching*, respectively. We compare 4S with the following methods: FS as baseline in full space;

Step	Brute-force	4S
Computation of $\mathcal{L}_2$	$O(D^2N^2)$	$O(DN)$
Mining of $\mathcal{L}_k$	$O(3^{D/3})$	$O(DN^{1/3}\log N)$
Subspace merge	$O(2^{(D-\log N)N^{1/3}})$	$O((D-\log N)^3 lN)$
Overall complexity	$O(2^{(D-\log N)N^{1/3}})$	$O((D-\log N)^3 lN)$

Table 4.1.: Overview of complexity reduction.

FB [LK05] using random subspaces for outlier mining; ENCLUS [CFZ99], CMI, and HICS [KMB12] representing the APR-style methods; DensEst [MAK+09] (plugged into the mining framework of [MAGS11]) representing jump search for local processing; FEM [DB04] representing the unsupervised feature selection approaches. Note that DensEst applied to the framework of [MAGS11] is for density-based subspace clustering. So to use it for other tasks (e.g., outlier detection) we extract the subspaces out of the clusters it produces. Besides, we include Krimp [VvLS11] and CompreX [ATVF12], two methods for compressing data by itemset mining. In short, Krimp and CompreX mine itemsets that compress the data well. Similarly to DensEst, we extract the subspaces out of these itemsets. Further, as DensEst, Krimp, and CompreX work on discrete data, we follow [NMVB14] to discretize the data.

For all of these methods, we try several parameter combinations in order to find the optimal parameter setting for each data set. Finally, we use the results of the parameter combination showing the best quality results. We note that we attempt to find the most relaxing parameters for ENCLUS, CMI, and HICS that can return good results within five days. As shown later, this practice, however, is not always possible due to the high complexity of APR methods. This is also the reason why we exclude CMI++ here: With restricted parameter setting, its results are about the same as those of CMI.

For our 4S-M and 4S-S, following Theorem 13, we set $K = D\log N$ to ensure a reasonable tradeoff between quality and efficiency. For 4S-S, we need to set s_1 and s_2. Regarding the former, we fix $s_1 = 10000$ as large values of s_1 yield high estimation accuracy [PP12]. For the latter, we fix $s_2 = 2$ following the observation that smaller values for s_2 generally result in better accuracy [DGGR02].

We study the quality of subspaces produced by the methods tested in: outlier detection with LOF [BKRTN00], clustering with DBSCAN [EKSX96], and classification with the C4.5 decision tree. The first two areas are known to yield meaningful results when the subspaces selected have high correlations, i.e., include few or no irrelevant dimensions [CFZ99, KMB12, LK05, MGAS09, NMV+13]. Hence, they are good choices for evaluating the quality of correlated subspaces. The third area is to show that correlated subspaces found by 4S are also useful for the supervised domain. For each method, LOF, DBSCAN, and C4.5 are applied on its detected subspaces and the results are combined (following [LK05] for LOF, [CFZ99] for DBSCAN, and [Ho98] for C4.5), and judged using corresponding well-known per-

Data set	Size	Attributes	Classes
Gisette	13500	5000	2
HAR	10299	561	6
KIT	48501	540	2
Mutant1	16592	5408	2
Mutant2	31159	5408	2
PAMAP1	1686000	42	15
PAMAP2	1662216	51	18

Table 4.2.: Characteristics of real-world data sets. Each of them has more than 1 trillion subspaces.

formance metrics. Note that when using DensEst for clustering purposes, this procedure is not applied; that is, we do not perform DBSCAN on the subspaces produced by DensEst and instead directly evaluate its subspace clusters (also obtained by means of DBSCAN [MAGS11]). Regarding parameter settings: We set $MinPts$ of LOF to about $\min(N \cdot 0.005, 100)$. For DBSCAN, we set $MinPts$ from 2 to 10, and ϵ from 0.01 to 0.04. For C4.5, we use the default parameter setting in WEKA.

For this quantitative assessment of subspaces, we use synthetic data and 6 real-world labeled data sets from the UCI Repository: the Gisette data about handwritten digits; HAR, PAMAP1, and PAMAP2 all sensor data sets with physical activity recordings; Mutant1 and Mutant2 containing biological data used for cancer prediction. Further, we use the facility management's database of our university (KIT) with energy indicators recorded from 2006 to 2011 (described in Chapter 1). More details are in Table 4.2. Note that each of them has more than 1 trillion subspaces. This features a challenging search space w.r.t. dimensionality for all methods.

Besides, we also further our study on the performance of 4S by experimenting it with a real-world unlabeled data set on climate and energy consumption. Our objective here is to qualitatively assess the subspaces detected by 4S, e.g., if they make sense to our domain expert. Lastly, we investigate how well the subspace merge of 4S compresses the output subspaces, or in other words, how succinct the output of 4S is.

4.9.1. Experiments on synthetic data

Quality on outlier detection. We have created 6 synthetic data sets of 10000 records and 100 to 1000 dimensions. Each data set contains subspace clusters with dimensionality varying from 8 to 24 and we embed 20 outliers deviating from these clusters. Our performance metric is the Area Under the ROC Curve (AUC), as in [LK05, KMB12, KKSZ12]. From Table 4.3, one can see that 4S-M overall has the best AUC on all data sets. 4S-S in turn achieves the second-best performance. In fact, in most cases, 4S-M correctly discovers all embedded subspaces. Though 4S-S does not achieve that, its subspaces are close to the best ones, and it has

Data set	4S-M	4S-S	FS	FB	ENCLUS	CMI	HICS	DensEst	FEM	Krimp	CompreX
D100	**1.00**	**1.00**	**1.00**	0.65	0.90	0.46	0.43	0.88	0.50	0.92	0.98
D200	**1.00**	**1.00**	0.99	0.50	0.85	0.47	0.46	0.85	0.48	0.97	0.94
D400	**0.99**	0.98	0.96	0.51	0.83	0.46	0.45	0.87	0.63	0.94	0.95
D600	**0.99**	0.98	0.77	0.54	0.76	0.42	0.29	0.83	0.54	0.86	0.91
D800	**0.99**	0.87	0.75	0.61	0.74	0.43	0.40	0.77	0.59	0.82	0.84
D1000	**0.99**	0.92	0.81	0.47	0.75	0.46	0.40	0.80	0.64	0.84	0.87

Table 4.3.: AUC on outlier mining for synthetic data sets. Highest values are in **bold**.

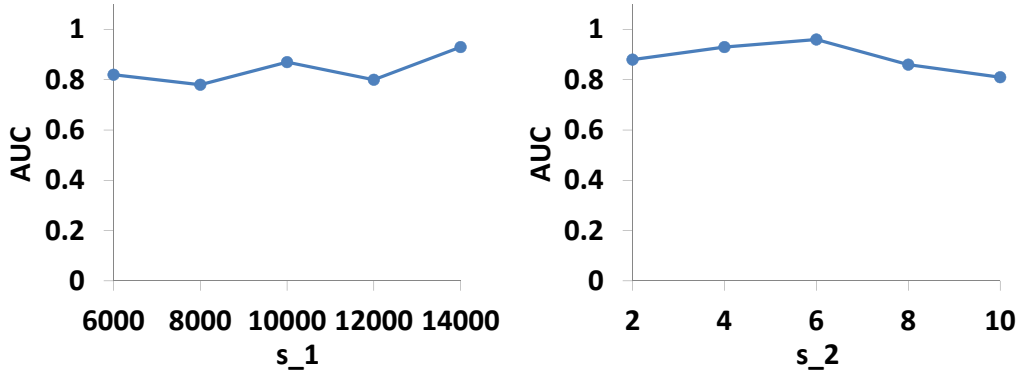
better performance than other methods. We are better than FS, which focuses on the full space where noisy dimensions likely hinder the detection of outliers. Our methods outperform FB, which highlights the utility of our correlated subspaces compared to random ones. DensEst, Krimp, and CompreX detect some correlated subspaces but they miss others, probably due to data discretization. Examining the subspaces found by APR-style methods (ENCLUS, CMI, and HICS), we see that they are either irrelevant, or they are low dimensional fragments of relevant subspaces. This explains their poor performance. FEM has low AUC since it only mines a single subspace, and hence, misses other important correlated subspaces where outliers are present.

We now study the sensitivity of 4S-S to different parameter settings. In particular, to assess how its performance is impacted by the setting of s_1, we fix $s_2 = 2$. On the other hand, to assess its sensitivity to s_2, we fix $s_1 = 10000$. We discuss the results on a data set with $N = 10000$ and $D = 800$ for illustration purposes. We note that the results on data sets of other dimensionality convey a rather similar information. From Figure 4.4, we notice that the performance of 4S-S (its AUC score) fluctuates as both parameters change, especially in the case of s_1. We observe that 4S-S has the best performance at $s_1 = 14000$, following by $s_1 = 10000$. As the second value yields better runtime (see Section 4.5.2), we choose it as our setting for s_1 in the remaining experiments. In turn, we see that the performance of 4S-S does not change much with $2 \leq s_2 \leq 6$. Thus, for efficiency reasons, we fix $s_2 = 2$ in other experiments. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable parameterization might be required.

Quality on clustering. Synthetic data sets with 100 to 1000 dimensions are used again. Our performance metric is the F1 measure, as in [MGAS09, MAG+09]. Table 4.4 displays clustering results of all methods. One can see that 4S-M and 4S-S have the best performance on all data sets tested. This again highlights the quality of subspaces found by our methods.

From the outlier detection and clustering experiments, we can see that 4S-S is a good approximation of 4S-M.

APR using subspace merge. For illustration, we only present the outlier detection and clustering results on the synthetic data set with 10000 records and 1000 dimensions. From Table 4.5, by applying the subspace merge, APR-style methods



(a) AUC vs. s_1 (b) AUC vs. s_2

Figure 4.4.: [Higher is better] Sensitivity of 4S-S to s_1 and s_2 on a synthetic data set with $N = 10000$ and $D = 800$.

Data set	4S-M	4S-S	FS	FB	ENCLUS	CMI	HICS	DensEst	FEM	Krimp	CompreX
D100	**0.99**	**0.99**	0.72	0.95	0.67	0.50	0.80	0.90	0.76	0.92	0.95
D200	**0.89**	**0.89**	0.67	0.66	0.67	0.50	0.80	0.81	0.76	0.84	0.83
D400	**0.85**	0.83	0.67	0.81	0.67	0.80	0.77	0.77	0.75	0.76	0.79
D600	**0.96**	0.95	0.67	0.66	0.67	0.67	0.83	0.83	0.53	0.83	0.85
D800	**0.99**	0.93	0.67	0.67	0.67	0.67	0.83	0.81	0.74	0.84	0.88
D1000	**0.91**	0.88	0.67	0.67	0.83	0.67	0.74	0.78	0.75	0.81	0.85

Table 4.4.: F1 on clustering for synthetic data sets. Highest values are in **bold**.

achieve better AUC and F1 values than without merge. Yet, our methods outperform all of them. This is because APR-style methods already face severe issue with reaching high dimensional subspaces. Thus, applying subspace merge in their case does not bring much improvement.

Scalability. Since FS and FB do not spend time for finding subspaces, we only analyze the runtime of the remaining methods. To test scalability to dimensionality, we use data sets with 10000 data points and dimensionality of 100 to 1000. Based on Figure 4.5, we see that DensEst and 4S-S have respectively the best and second best scalability. FEM scales better than 4S-M because it only searches for a single subspace. Krimp and CompreX also have good scalability due to their effective pruning schemes. Overall, 4S-S has near-linear scalability to dimensionality, thanks to our efficient search scheme.

Task	4S-M	4S-S	ENCLUS	CMI	HICS
Outlier Mining (AUC)	**0.99**	0.92	0.76	0.49	0.44
Clustering (F1)	**0.91**	0.88	0.84	0.70	0.76

Table 4.5.: Comparison with APR using subspace merge on the synthetic data set with 10000 records and 1000 dimensions. Highest values are in **bold**.

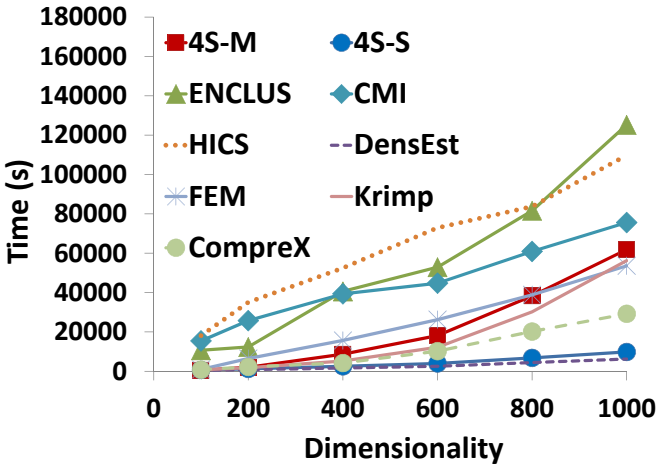


Figure 4.5.: Runtime vs. dimensionality on synthetic data.

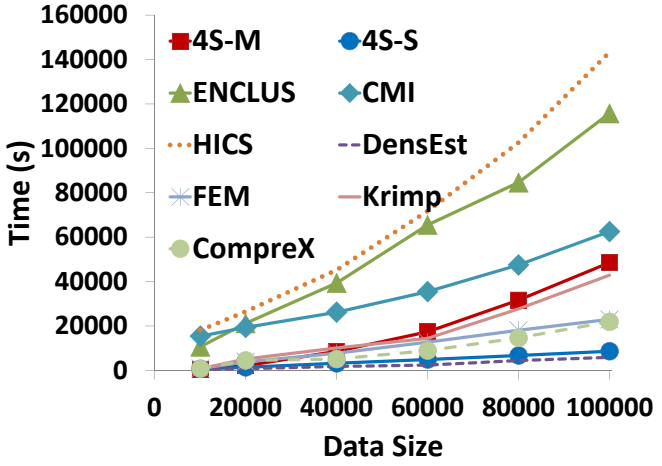


Figure 4.6.: Runtime vs. data size on synthetic data.

For scalability to data size, we use data sets with 100 dimensions and sizes of 10000 to 100000. From Figure 4.6, we see that 4S-S scales linearly and is more efficient than 4S-M. This agrees with our theoretical analysis.

We also note that the runtime of the first step in our methods dominates the other two steps. For example, on the data set of 10000 records and 1000 dimensions, 4S-S takes about 150 minutes for the first step and only 14 minutes for the remaining two steps. These costs in turn are negligible compared to the cost of performing clustering and outlier detection on the subspaces detected. For instance, LOF requires about 4 days to process this data set.

From the results obtained, we can conclude that 4S-S achieves the efficiency goal while still ensuring high quality of subspaces found. Though DensEst is faster, it misses certain important correlated subspaces, which explains why it is inferior

Data set	4S	FS	FB	ENCLUS	CMI	HICS	DensEst	FEM	Krimp	CompreX
Gisette	**0.77**	0.67	0.60	0.73	0.74	0.74	0.72	0.68	0.72	0.73
HAR	**0.67**	0.42	0.53	0.27	0.65	0.15	0.62	0.53	0.62	0.60
KIT	**0.73**	0.36	0.51	0.33	0.55	0.55	0.64	0.44	0.67	0.69
Mutant1	**0.62**	0.58	0.55	0.56	0.58	0.57	0.55	0.55	0.56	0.58
Mutant2	**0.64**	0.57	0.53	0.55	0.58	0.59	0.58	0.56	0.57	0.60
PAMAP1	**0.86**	0.54	0.47	*	*	*	0.75	0.48	0.77	0.83
PAMAP2	**0.87**	0.53	0.45	*	*	*	0.72	0.41	0.75	0.84

Table 4.6.: AUC on outlier mining for real-world data sets. Highest values are in **bold**. (*) means the result is unavailable due to excessive runtime.

to 4S-S in terms of quality on both clustering and outlier detection. From now onwards, we use 4S-S for the remaining experiments and write only 4S.

4.9.2. Experiments on real data

We apply all methods to two applications: outlier detection and classification. Clustering is skipped here since it conveys similar trends among the methods as with synthetic data.

Quality on outlier detection. As a standard procedure in outlier mining [LK05, KMB12, KKSZ12], the data sets used are converted to two-class ones, i.e., each contains only a class of normal objects and a class of outliers. This is done by either picking the smallest class or down-sampling one class to create the outlier class. The rest forms the normal class. From Table 4.6, 4S achieves the best results. Its superior performance compared to other methods, including APR-style methods (ENCLUS, CMI, and HICS), stems from the fact that 4S better discovers correlated subspaces where outliers are visible. For example, on the KIT data set, 4S finds subspaces where several consumption indicators of different buildings of the same type (e.g., office buildings, laboratories) cluster very well with a few exceptions, possibly caused by errors in smart-meter readings, or rare events (e.g., university holidays when energy consumption is low or large-scale physics experiments when electricity consumption is extremely high). These subspaces however are not discovered by *all* other methods.

On the PAMAP1 and PAMAP2 data sets, we can only compare 4S against FS, FB, DensEst, FEM, Krimp, and CompreX. This is because other methods take excessively long time without completing. These data sets contain data collected by sensors attached to human bodies when they perform different activities, e.g., walking, running, ascending stairs. The best AUC of 4S on both data sets once again implies that 4S successfully discovers high quality subspaces, which in turn assist in the detection of outliers. For example, the subspaces found by 4S on PAMAP1 exhibit correlations among the hand, chest, and ankle of human subjects. There are of course different grouping patterns representing different types of activity. In any case, such correlations let records representing transient activities become outliers. This is intuitive because those activities are very random and do not feature any specific correlation among different parts of human bodies [RS11].

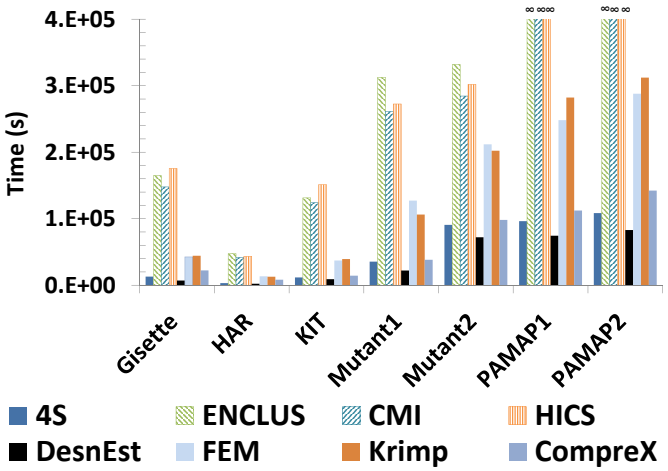


Figure 4.7.: Runtime (in seconds) of subspace search methods on real-world data sets. ENCLUS, CMI, and HICS did not finish within 5 days on the PAMAP data sets.

Data set	4S	Random Forest	FEM	CFS
Gisette	0.76	0.75	0.72	**0.84**
HAR	0.83	0.81	0.74	**0.85**
KIT	**0.97**	0.96	0.85	0.92
Mutant1	**0.99**	0.88	0.85	0.97
Mutant2	**0.99**	0.87	0.89	0.98
PAMAP1	**0.91**	0.71	0.69	0.87
PAMAP2	**0.93**	0.71	0.66	0.86

Table 4.7.: Classification accuracy for real-world data sets. Highest values are in **bold**.

In Figure 4.7, we show the wall-clock runtime (in seconds) for each subspace search method. We note that Apriori search techniques ENCLUS, CMI, and HICS did not finish within 5 days on the PAMAP data sets. The results show that except for DensEst, 4S is much faster than all other competitors. Combining with the results on quality, we can see that 4S yields the best balance between quality and efficiency.

Quality on classification. We here test 4S against the well-known Random Forest classifier [Ho98], FEM for unsupervised feature selection, and CFS [Hal00] for supervised feature selection. We skip other methods since previous experiments already show that 4S outperforms them. The classification accuracy (obtained by 10-fold cross validation) is in Table 4.7. Overall, 4S consistently yields better accuracy than Random Forest and FEM. It is comparable to CFS which has access to the class label. The results obtained show that the correlated subspaces found by 4S are also useful for data classification.

4.9.3. Discovering novel correlations on climate data

In this experiment, we evaluate whether 4S can be used to discover novel correlated subspaces in non-benchmark data. To this end, we apply 4S on a large real-world data set of climate and energy consumption measurements for an office building in Frankfurt, Germany [WLV+14]. After data pre-processing to handle missing values, our final data set contains 35601 records and 251 dimensions. Example dimensions include room CO_2 concentration, indoor and outdoor temperature, temperature produced by heating systems, drinking water consumption, and electricity consumption by different devices, etc. Since this data set is unlabeled, we cannot calculate clustering/outlier detection/classification quality as above. Instead, we focus on detecting correlated subspaces, and investigate the discovered correlations. Our objective is thus to study how climate and energy consumption indicators interact with each other.

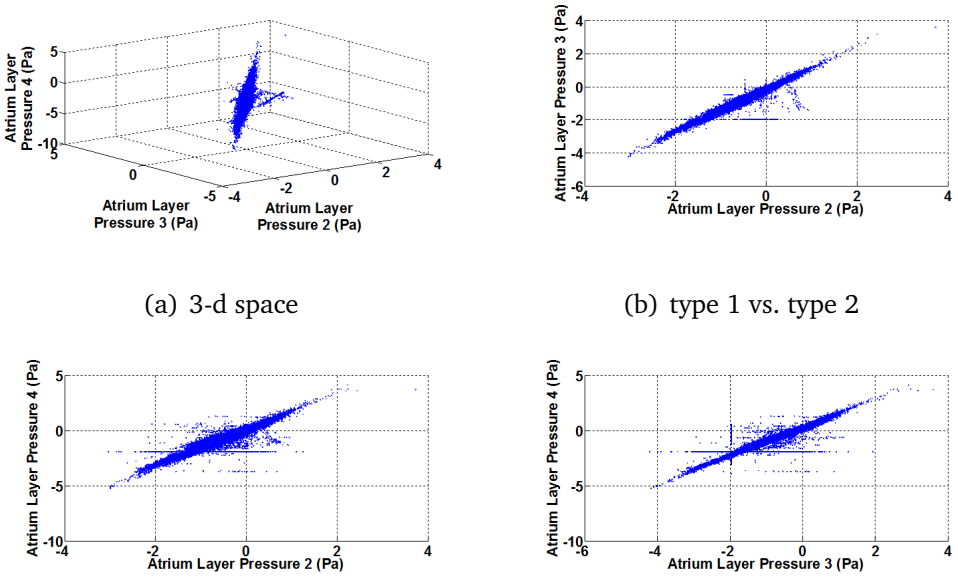
Overall, the results show that 4S detects many interesting high dimensional correlated subspaces. All reported correlations are significant at $\alpha = 0.05$. We verified all findings with a domain expert, resulting in some already known correlations, and others that are novel.

An example of a known multivariate correlation discovered using 4S is the one among atrium layer pressures (see Figure 4.8). This correlation is linear. Another known but more complex correlation is the one among between the air temperature supplied to the heating system, the temperature of the heating boiler, and the amount of heating it produces (see Figure 4.9). This relation is expected. However, the most interesting point is the interaction between the temperature of the heating boiler and the amount of heating produced. Intuitively, the higher the former, the larger the latter. However, the correlation is not linear. Instead, it seems to be a combination of two quadratic functions (see Figure 4.9(b)).

4S also finds an interesting correlation among the drinking water consumption, the outgoing temperature of the air conditioning (cooling) system, and the room CO_2 concentration (see Figure 4.10). There is a clear tendency: the more water consumed, the higher the CO_2 concentration (see Figure 4.10(c)). Besides, there is a sinusoidal-like correlation between the drinking water consumption and the outgoing temperature of the cooling system (see Figure 4.10(b)). These correlations, novel to our domain expert, offer a view on how human behavior interacts with indoor climate and energy consumption.

Other significant correlations detected by 4S whose intuitions can be straightforwardly perceived are given below:

- wind speed, wind direction, outdoor temperature, outdoor CO_2 concentration, outdoor humidity
- amount of heating by wood, amount of heating by gas, outdoor temperature, temperature of room 401, temperature of room 402
- occupation of building, outdoor temperature, amount of drinking water consumption, electricity used for ventilator



(a) 3-d space

(b) type 1 vs. type 2

(c) type 1 vs. type 3

(d) type 2 vs. type 3

Figure 4.8.: Atrium Layer Pressures.

- outdoor temperature, light intensity north (of the building), south, east, west, amount of solar heating produced

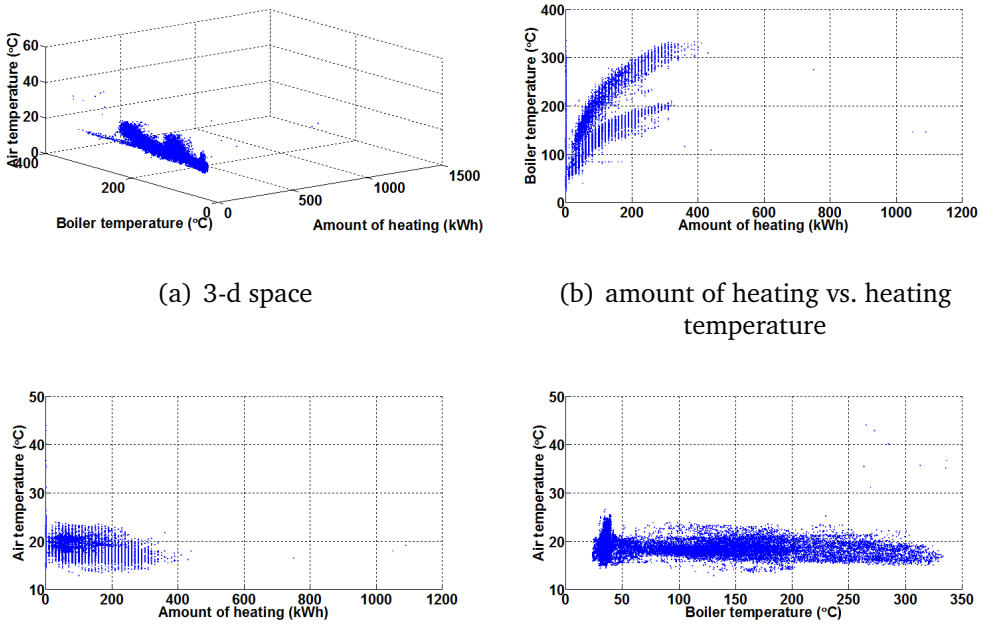
Overall, the correlations 4S detected range from linear ones to non-linear functional, and non-linear non-functional. In terms of runtime, 4S only needs about 1.5 hours to explore the huge search space of this data set. In conclusion, the results suggest that 4S is a practical tool for scalable correlation analysis—an important feature which is crucial for understanding and mining large and high dimensional real-world data.

4.9.4. Succinctness of output

We study the benefits of the MDL-based subspace merge on 4S. Our performance metric is the reduction ratio, i.e., $\frac{m}{m'}$ where m and m' are the number of subspaces before and after merging. The results are in Figure 4.11. We see that the merging phase achieves up to an order of magnitude reduction ratio. This verifies our claim that 4S produces a succinct set of overlapping correlated subspaces while still guaranteeing their quality (see for instance Sections 4.9.2). By providing end users with a succinct set of correlated subspaces, in practice 4S facilitates manual inspection and post-analysis which benefit advanced applications based on the knowledge derived from the subspaces.

4.10. Conclusions

Mining high dimensional correlated subspaces is a very challenging but important task for knowledge discovery in multivariate data. We have introduced 4S, a new



(a) 3-d space

(b) amount of heating vs. heating temperature

(c) amount of heating vs. air temperature

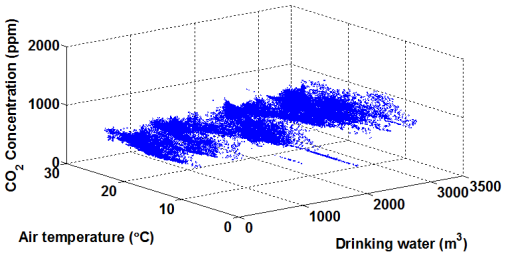
(d) heating temperature vs. air temperature

Figure 4.9.: Correlation among air temperature, heating temperature, amount of heating.

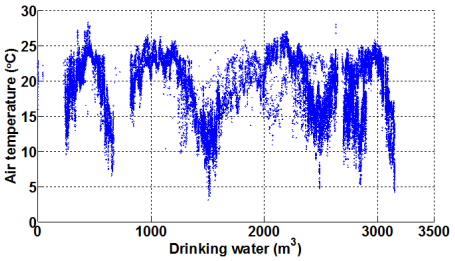
scalable subspace search scheme that addresses the issue. 4S works in three steps: scalable computation of $\mathcal{L}_2$, scalable mining of $\mathcal{L}_k$ ($k > 2$), and subspace merge to reconstruct fragmented subspaces and to reduce redundancy. Our experiments show that 4S scales to data sets of more than 1.5 million records and 5000 dimensions (i.e., more than 1 trillion subspaces). Not only being efficient, compared to existing methods 4S better detects high quality correlated subspaces that are useful for outlier mining, clustering, and classification.

The superior performance of 4S compared to existing methods comes from (a) our new notion of correlated subspace that has proved to be more general than existing notions, and hence, allows to discover subspaces missed by such methods, (b) our scalable subspace search scheme that can discover high dimensional correlated subspaces, and (c) the subspace merge that can recover fragmented subspaces and remove redundancy.

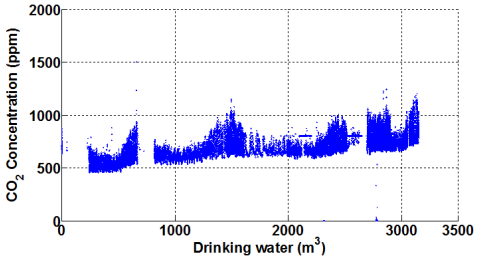
Nevertheless, we design 4S specifically for numerical data. Since real-world data often contains data of different types, e.g., numerical and categorical, it is interesting to address the problem of mining correlated subspaces in mixed typed data. The next chapter is devoted to such a study. However, we do not only target mixed data types: We take into account the fact that data may be stored in multiple relations as well. Thus, the upcoming chapter, in a more precise way to put, is about detecting groups of correlated columns (dimensions) in relational databases.



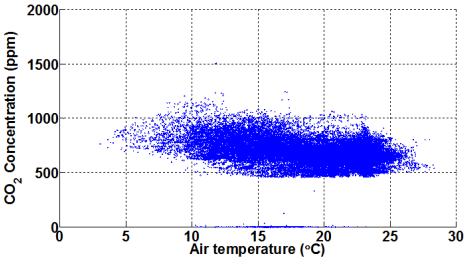
(a) 3-d space



(b) drinking water vs. cooling air temperature



(c) drinking water vs. CO_2 concentration



(d) cooling air temperature vs. CO_2 concentration

Figure 4.10.: Correlation among cooling air temperature, drinking water consumption, and CO_2 concentration.

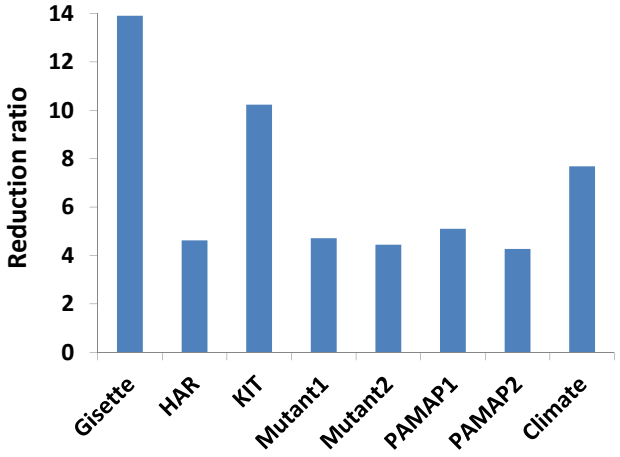


Figure 4.11.: Reduction ratio of the MDL merging phase. 4S achieves up to an order of magnitude reduction ratio.

Part III.

Mining Correlated Subspaces in Mixed Typed Data

5. Mining Correlated Columns in Databases with Mixed Data Types

This chapter is based on our work originally published as [NMAB14]:

H. V. Nguyen, E. Müller, P. Andritsos, and K. Böhm, *Detecting correlated columns in relational databases*, in SSDBM, 2014, p. 30.

In Chapters 3 and 4, we focus on the scalable mining of correlated subspaces in individual real-valued data sets. In this chapter, we still target at subspace search. However, we broaden our study to relational databases containing multiple relations and columns (dimensions) with heterogeneous data types. As we focus on databases in this chapter, we call dimensions as columns and correlated subspaces as groups of correlated columns. This is to ensure uniform terminologies with other work on databases.

In general, a relational database usually contains multiple relations (tables); each can be perceived as a data set. Each relation in turn can contain multiple columns (dimensions). Each column's data type can either be real-valued numerical, discrete numerical, or categorical. For instance, the original KIT Energy database contains different relations (Building, Institute, Consumption) and columns of different data types (total area of buildings: real-valued numerical, the number of employees in buildings: discrete numerical, and the location of buildings: categorical). The question of our interest remains intact: How to mine groups of correlated columns, where the columns can belong to different relations, in a scalable manner? At a first glance, this question can be resolved by simply joining all relations together to create a universal relation [TDJ11], and then applying existing subspace search methods (e.g., CMI or 4S) to extract groups of correlated columns out of the universal relation. However, this practice suffers from two major drawbacks. First, it potentially incurs a high storage overhead to store the universal relation. Second, CMI and 4S are not readily applicable to mixed data types due to their correlation measures. Thus, a new solution is required.

Therefore, we propose DECOREL, our novel method for *Detecting Column Correlations*. In short, DECOREL employs the jump search (with modifications as explained later) proposed in Chapter 4. That is, DECOREL starts by mining pairs of correlated columns (possibly located in different relations). Then, it jumps directly to groups of multiple correlated columns. By using the jump search, DECOREL only needs to focus first on pairwise correlations, and hence, it only needs to join maximum two relations at a time. Further, as we aim to handle mixed data types in this chapter, we modify some details of the original jump search in Chapter 4 to better fit the new scenario. In particular, previously we choose top pairs of dimensions/columns with largest correlation scores to construct the correlation graph. In DECOREL, we accomplish this by an adaptive thresholding scheme, which is inspired from eigenvalue spectrum analysis [JMK09]. Second, while we go for cliques in Chapter 4, here we mine quasi-cliques that are more fault-tolerant to noise [JP09]. Regarding the correlation measure, we propose a pairwise information-theoretic correlation measure, which is a blend between the pairwise version of the CMI++ measure and mutual information. Recalling that both are information-theoretic measures, we hence achieve a handling of mixed data types with no major change in the interpretation of correlation scores. Further, we show that this close relation between the CMI++ measure and mutual information enables a cross-data-type correlation computation. Besides, we note that our correlation measure here is related to the one employed in [PLP+10]; yet, as we will point out their measure uses Shannon entropy and does not consider real-valued data. Our measure in turn covers heterogeneous data types.

This chapter is organized as follows. Section 5.1 covers related work in the database area. Section 5.2 provides preliminaries and general notions. Section 5.3 gives the details of our correlation measure. Section 5.4 explains how to find mutual correlations through pairwise correlations. Section 5.5 describes our algorithm for grouping correlated columns. Section 5.6 presents our analysis on the generality of DECOREL compared to a recent approach [ZHO+11]. Section 5.7 reports our empirical study. Section 5.8 presents an important proof and Section 5.9 concludes the chapter.

5.1. Related Work in Database Research

Apart from related work in subspace search mentioned in Section 1.3 of Chapter 1, there are also other methods in the database area which are capable of mining columns with certain relationships. We review some representatives in the following.

ECLUS [ZHO+11] groups database columns that have similar distributions. It differs from DECOREL in many aspects. First, it uses the EMD test [LB01] which is bound to discover groups of columns with the same data type and overlapping value domains. A detailed analysis on this issue is given in Section 5.6. Second, it only detects non-overlapping groups, and hence, misses other semantics that a column may have.

CORDS [IMH+04] and ENTROPY [PLP+10] mine overlapping pairs of correlated columns using χ^2 test and Shannon entropy, respectively. They focus on discrete and categorical data; real-valued data is not included in their studies. Further, they lack a mechanism to combine those pairs into larger meaningful groups. Lastly, both χ^2 test and Shannon entropy are better suited to discrete and categorical data, limiting the generality of both CORDS and ENTROPY. DECOREL in turn detects overlapping groups of two or more columns. In addition, it is applicable to real-valued, discrete, and categorical data.

GORDIAN [SBHR06] and DUCC [HQRA+13] aim at finding overlapping composite primary keys. That is, unlike DECOREL, they do not detect groups of columns that are correlated but do not form any key.

Finding overlapping groups of correlated columns is also addressed by selectivity estimation methods, such as [TDJ11]. Nevertheless, for efficiency reasons, they keep the group sizes small (typically 2). Thus, their goal is not to optimize the quality of groups. In contrast, DECOREL discovers groups that are potentially useful for multiple purposes, e.g., selectivity estimation. Applying DECOREL to each specific task is a subject of our future work.

Clustering relational columns has also been explored in [AHS+09]. It groups columns based on generic data types. The type information of columns is captured using q-grams. Thus, this method can also benefit DECOREL when there is insufficient information to correctly identify data types of columns.

5.2. Preliminaries

Here we re-introduce the main terminologies related to correlation analysis. Most of them have been introduced in Chapter 2. However, as we are switching to the database domain, adapting these terminologies accordingly will ease the subsequent presentation.

Let $\mathcal{R}$ be the set of relations in the database. We write $\mathcal{C}$ as the set of all columns in $\mathcal{R}$. We regard each column $C \in \mathcal{C}$ as a random variable with a distribution $p(C)$. For notational convenience, we use C to represent both the column C and its associated random variable. If C is discrete or categorical, $p(C)$ is named the *probability mass function* (pmf for short). Otherwise, i.e., C is real-valued, $p(C)$ is called the *probability density function* (pdf for short). Let $g = \{C_i\}_{i=1}^{d} \subseteq \mathcal{C}$ be a group of columns. We write $p(C_1, \ldots, C_d)$ as the joint probability function of all columns in g, or of g for short. If g contains all discrete, or all categorical, or a mix of discrete and categorical columns, $p(C_1, \ldots, C_d)$ is a pmf. If g contains all real-valued columns, $p(C_1, \ldots, C_d)$ is a pdf. For simplicity, we call it probability function in any case.

A correlation function $Corr$ assigns each group g with at least two columns a non-negative real-valued correlation score, denoted as both $Corr(g)$ and $Corr(C_1, \ldots, C_d)$. In principle, $Corr(g)$ quantifies the extent to which its joint probability function differs from the product of its marginal probability functions

(see Chapter 2). The larger the difference, the higher $Corr(g)$ is, i.e., the more correlated the columns of g are. When g has a high correlation score, we regard g to be a group of correlated columns. In that case, the columns of g are (a) mutually (multi-way) correlated if g has more than two columns, and (b) pairwise correlated otherwise. (We discuss later how to decide if a correlation score is high.) When the context is clear, we omit 'pairwise' and 'mutually'.

In the next section, we describe our correlation measure $Corr$. Then, we explain the intuition of reasoning about mutual correlations by means of pairwise correlations in Section 5.4. Finally, we present an efficient group discovery algorithm in Section 5.5.

5.3. Our Correlation Measure

As a basic building block of our method, we present our correlation measure in this section. Our general goal is to mine groups with arbitrary numbers of correlated columns. Yet we will show later that one can find such groups by analyzing pairs of correlated columns. As a result, our $Corr$ measure of correlation is *pairwise* (we describe how to form the joint distribution of two columns in Section 5.3.1). Further, $Corr$ is able to handle heterogeneous data types. In order to help readers to understand this feature of $Corr$, we first extend the notion of conditional CE presented in Chapter 3 to incorporate discrete/categorical data.

Definition 13. ***Conditional CE:***
Consider a column Y.
If Y is discrete or categorical, then

$$h(X \mid Y) = \sum h(X \mid y)p(y)$$

where $p(Y)$ is the pmf of Y. Otherwise,

$$h(X \mid Y) = \int h(X \mid y)p(y)dy$$

where $p(Y)$ is the pdf of Y.

The extended conditional CE still possesses two important properties of the original CE, which are:

- $h(X \mid Y) \geq 0$ with equality iff X is a function of Y, and
- $h(X \mid Y) \leq h(X)$ with equality iff X is independent of Y, i.e., $p(X, Y) = p(X) \cdot p(Y)$.

These properties guarantee the suitability of the extended conditional CE for correlation analysis. Assume that we want to measure the correlation between two columns X and Y. As in CMI++, we aim at a correlation measure that produces scores in the range $[0, 1]$ (with 0 being no correlation at all) for an unbiased correlation analysis. This is an important feature for processing heterogeneous data types. Thus, we define $Corr(X, Y)$ as follows.

Definition 14. ***Correlation Measure Corr:***
$Corr(X, Y)$ is equal to

- $\frac{H(X) - H(X \mid Y)}{H(X)}$ *if X is categorical,*
- $\frac{h(X) - h(X \mid Y)}{h(X)}$ *if X is numerical (either real-valued or discrete).*

Note that if Y is discrete or categorical, then

$$H(X \mid Y) = \sum H(X \mid y)p(y) \quad .$$

Otherwise,

$$H(X \mid Y) = \int H(X \mid y)p(y)dy \quad .$$

In particular, when X is categorical we use $H(X) - H(X \mid Y)$ (i.e., the mutual information of X and Y) to quantify the correlation of both columns. For normalization, we use $H(X)$, which is a tight upper bound of $H(X) - H(X \mid Y)$ [CT06]. When X is numerical, we use $h(X) - h(X \mid Y)$ which also quantifies the correlation of X and Y; in addition, we obtain normalization with $h(X)$, which is a tight upper bound of $h(X) - h(X \mid Y)$ (see Chapter 3). Hence, our correlation measure $Corr$ essentially combines the nice characteristics of both mutual information and CMI++. It is straightforward to verify that it satisfies Properties 1 and 2, and Challenge 1. Further, its values are in $[0, 1]$. As we will show later, this facilitates the handling of heterogeneous data types. Since $Corr(X, Y)$ may not be equal to $Corr(Y, X)$, $Corr$ is asymmetric. This is beneficial for analyzing asymmetric dependencies among columns; one of which is the functional dependency [IMH+04]. For instance, zip code functionally determines the city name but the reverse may not be true.

Example 4. *Considering our Energy database, let E be the column capturing the electricity consumption (real-valued) of each building, N be the column for its number of staffs (discrete), and L be the column for its location (categorical). Following Definition 14, we have:*

$$Corr(E, N) = \frac{h(E) - h(E \mid N)}{h(E)}$$

$$Corr(E, L) = \frac{h(E) - h(E \mid L)}{h(E)}$$

$$Corr(N, E) = \frac{h(N) - h(N \mid E)}{h(N)}$$

$$Corr(N, L) = \frac{h(N) - h(N \mid L)}{h(N)}$$

$$Corr(L, E) = \frac{H(L) - H(L \mid E)}{H(L)}$$

$$Corr(L, N) = \frac{H(L) - H(L \mid N)}{H(L)}$$

$Corr$ is asymmetric, e.g., in general $Corr(E, L) \neq Corr(L, E)$.

To this end, we note that our $Corr$ measure is related to that of ENTROPY [PLP+10]. In particular, ENTROPY represents the correlation between two columns X and Y as $\frac{H(X)-H(X|Y)}{H(X)}$ and $\frac{H(Y)-H(Y|X)}{H(Y)}$, regardless of their data types. Because of this, ENTROPY does not take into account the fact that directly applying Shannon entropy on real-valued data is unreliable (see Chapter 2). Further, as mentioned in Section 5.1 ENTROPY does not have a mechanism to combine correlated pairs into larger meaningful groups. DECOREL in turn is capable of using $Corr$ to discover groups of correlated columns with heterogeneous data types.

In Definition 14, when X is numerical, we compute $Corr(X, Y)$ using CE since CE is applicable to both real-valued and discrete data. This helps us to avoid further checks of data types. In reality, database programmers sometimes declare discrete columns as real-valued, and such a check may cost additional effort. To show that $Corr$ is indeed suitable for correlation analysis, we prove the following results:

Lemma 14. *It holds that:*

- $0 \leq Corr(X, Y) \leq 1$.
- $Corr(X, Y) = 0$ *iff X and Y are statistically independent.*
- $Corr(X, Y) = 1$ *iff X is a function of Y.*

Proof. The results follow from the properties of Shannon entropy and CE (see Chapter 3). □

Based on Lemma 14, we derive the following lemma:

Lemma 15. $Corr(X, Y) > 0$ *iff* $p(X, Y)$ *and* $p(X) \cdot p(Y)$ *are different.*

We will use Lemma 15 in Section 5.4 where we explain how to find mutual correlations by means of pairwise correlations.

5.3.1. Forming joint distributions

Assume that we want to compute $Corr(X, Y)$ where X and Y are two columns. If X and Y belong to the same relation R, their joint distribution is defined as the projection of R onto (X, Y) (duplicates are kept).

If X and Y belong to two different relations R_1 and R_2, respectively, without any join relationship, their joint distribution is undefined, and they are considered to be independent.

If R_1 and R_2 have some join relationship, then we form a joint distribution for X and Y by performing a *left outer join* between R_1 and R_2. In this scenario, the outer join is preferred to the inner join since we want to punish unmatched values. The left outer join is used to reflect the asymmetric nature of $Corr$.

		X		
		r	**d**	**c**
Y	**r**	Case 1	Case 1	Case 1
	d	Case 2	Case 2	Case 2
	c	Case 3	Case 3	Case 3

Table 5.1.: Matrix of computation. 'r' is for real-valued, 'd' for discrete, and 'c' for categorical.

5.3.2. Computing $Corr$ on empirical data

Given the definition of the $Corr$ measure, we now describe how we compute it based on the type of value stored in column Y. A detailed mapping of which case to apply for $Corr(X, Y)$ is given in Table 5.1.

Case 1: Y is real-valued. W.l.o.g., we assume that X is categorical. The case for when X is numerical follows similarly. To compute $Corr(X, Y)$, we need to compute $H(X)$ and $H(X \mid Y)$. Computing $H(X)$ is straightforward. Computing $H(X \mid Y)$ on the hand also suffers from the empty space issue as conditional CE (cf., Section 3.4.2, Chapter 3). This in turn leads to unreliable correlation scores. To illustrate this, let us consider the following example.

Example 5. *Suppose that the joint distribution of X (categorical) and Y (real-valued) is as follows:*

X	orange	red	orange	red	orange	red
Y	1.00	1.01	2.00	2.01	4.00	4.01

Sticking to the exact formula of $H(X \mid Y)$, we have $H(X \mid Y) = \frac{1}{6}H(X \mid Y = 1.00) + \frac{1}{6}H(X \mid Y = 1.01) + \frac{1}{6}H(X \mid Y = 2.00) + \frac{1}{6}H(X \mid Y = 2.01) + \frac{1}{6}H(X \mid Y = 4.00) + \frac{1}{6}H(X \mid Y = 4.01) = 0$. Thus, $Corr(X, Y) = H(X) - H(X \mid Y) = 1 - 0 = 1$. This result is not accurate since it is due to small mismatches in the values of Y—a scenario which is common for real-valued columns. A more correct computation of $H(X \mid Y)$ is given in Example 6.

To overcome the issue, we propose to search for the histogram (discretization) of Y that minimizes $H(X \mid Y)$, i.e., maximizes $Corr(X, Y)$, by dynamic programming. Note that our solution here is based on the computation of CMI++ and the computation of mutual information in [RRF+11]. For readability, we skip the detail of our histogram search for now and present it instead in Section 5.8. Overall, by means of histogram construction, we can overcome the empty space issue and obtain a reliable estimation of $H(X \mid Y)$.

Example 6. *Continuing Example 5. To simplify our illustration, we here assume that the equal-frequency technique is applied where the number of bins is 3. Note, however, that in reality our method does not require fixing the number of bins in advance.* DECOREL *produces following bins for Y: $b_1 = [1.00, 2.00)$, $b_2 = [2.00, 4.00)$, and*

$b_3 = [4.00, 4.01]$. *We have* $H(X \mid Y) = \frac{1}{3}H(X \mid b_1) + \frac{1}{3}H(X \mid b_2) + \frac{1}{3}H(X \mid b_3) = 1$. *Thus,* $Corr(X, Y) = H(X) - H(X \mid Y) = 1 - 1 = 0$. *We note that this result makes sense since* X *is randomly distributed in any bin of* Y*, i.e., knowing the value (bin) of* Y *tells us nothing about* X.

Note that when forming the histogram for Y, we place its NULL values (caused by the left outer join) into the same histogram bin. We expect the values of X in this bin to be very disparate. In other words, the NULL bin is expected to have high entropy/CE with respect to X. Hence, $Corr(X, Y)$ gets smaller. This helps us to achieve our goal of punishing unmatched values between the two columns.

Case 2: Y is discrete. The usual way would be to compute $Corr(X, Y)$ as when Y is categorical. However, this approach may be prohibitive if Y has a large number of distinct values, for example, Y is an auto-generated primary key. To boost efficiency, similarly to Case 1, we also construct a histogram for Y. The rests thus are similar to Case 1.

Case 3: Y is categorical. Since categorical columns have straightforward distributions and thus information theoretic calculations, we omit further details of how to compute $Corr(X, Y)$.

5.4. From pairwise to mutual correlation

Since our goal is to detect groups of any size, we need to compute mutual correlations efficiently. In this section, we explain how to find mutual correlations through pairwise correlations. Our results are based on the theory of independence graphs [Whi90].

5.4.1. Independence graph

Consider a group $g = \{C_i\}_{i=1}^d$. Using $Corr$, we compute its pairwise correlation scores. Then, following [Whi90, MAK+09, TDJ11], we construct an independence graph $\mathcal{G}$ for g. In short, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = g = \{C_i\}_{i=1}^d$ (i.e., each column is a node) is undirected, acyclic, connected, and $(C_i, C_j) \notin \mathcal{E} \Leftrightarrow C_i \perp C_j \mid \mathcal{V} \setminus \{C_i, C_j\}$. That is, two columns C_i and C_j not connected by an edge are regarded as conditionally independent given all other columns $\mathcal{V} \setminus \{C_i, C_j\}$.

Example 7. *In Figure 5.1, we depict a possible independence graph for a hypothetical group* $g = \{C_1, C_2, C_3, C_4, C_5\}$. *One can see that this graph is undirected, acyclic, and connected. Further, since there is no edge connecting* C_1 *and* C_5*, they are considered to be conditionally independent given columns* C_2*,* C_3*, and* C_4*. Therefore,* $C_1 \perp C_5 \mid \{C_2, C_3, C_4\}$.

To show that it is feasible to reason about multivariate correlation through pairwise correlations, we prove the following result on the total correlation:

Theorem 15. $T(C_1, \ldots, C_d)$ *is equal to* $\sum_{(C_i, C_j) \in \mathcal{E}} I(C_i, C_j)$.

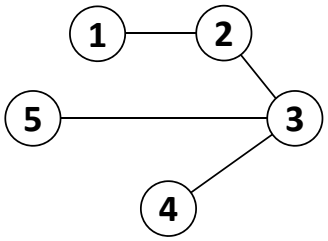


Figure 5.1.: Example of an independence graph for a group $g = \{C_1, C_2, C_3, C_4, C_5\}$.

Proof. From [MAK+09], $p(C_1, \ldots, C_d)$ is equal to:

$$\frac{\prod_{(C_i,C_j)\in\mathcal{E}} p(C_i, C_j)}{\prod_{C\in\mathcal{V}} p(C)^{deg(C)-1}}$$

where $deg(C)$ denotes the degree of C in $\mathcal{G}$.

W.l.o.g., we assume that $\{C_i\}_{i=1}^d$ are all real-valued. Then we have that $T(C_1, \ldots, C_d)$ is equal to:

$$\begin{aligned}
&\int p(\{c_i\}_{i=1}^d) \log \frac{p(\{c_i\}_{i=1}^d)}{\prod_{i=1}^d p(c_i)} dc_1 \cdots dc_d \\
&= \int p(\{c_i\}_{i=1}^d) \log \frac{\prod_{(C_i,C_j)\in\mathcal{E}} p(c_i, c_j)}{\prod_{C\in\mathcal{V}} p(c)^{deg(C)}} dc_1 \cdots dc_d \\
&= \sum_{(C_i,C_j)\in\mathcal{E}} \int p(\{c_i\}_{i=1}^d) \log \frac{p(c_i, c_j)}{p(c_i)p(c_j)} dc_1 \cdots dc_d \\
&= \sum_{(C_i,C_j)\in\mathcal{E}} \int p(c_i, c_j) \log \frac{p(c_i, c_j)}{p(c_i)p(c_j)} dc_i dc_j \\
&= \sum_{(C_i,C_j)\in\mathcal{E}} I(C_i, C_j)
\end{aligned}$$

□

Example 8. *Continuing Example 7. Following Theorem 15:* $T(C_1, C_2, C_3, C_4, C_5) = I(C_1, C_2) + I(C_2, C_3) + I(C_3, C_4) + I(C_3, C_5)$.

Theorem 15 shows a decomposition of total correlation into a sum of mutual information terms. In other words, it tells us that we can find mutual correlations by means of pairwise correlations.

We note that being able to estimate mutual correlations by pairwise correlations is insufficient. In particular, directly adopting the result of Theorem 15, a naïve solution would be as follows: For each group $g = \{C_i\}_{i=1}^d$, one measures the mutual information score of each column pair. Then, one constructs the maximum

spanning tree to obtain the independence graph of g [MAK+09, TDJ11]. One estimates the total correlation score of g using Theorem 15. Finally, one picks groups with largest scores. While this solution is straightforward, it suffers from two issues. First, as mentioned before, mutual information is not a reliable correlation measure for real-valued data. Second, the solution requires to examine each and every candidate group. However, the number of groups is still exponential in the number of columns. We address these issues next.

5.4.2. Our approach

We propose to mine groups g where each member column is correlated with *most* of the columns in g. We name such a group an approximate group. Our intuition behind this design choice is that by enforcing the requirement of almost perfect pairwise correlation among columns of g, the edge weights of its graph $\mathcal{G}$ (nodes are its columns and edge weights are pairwise correlation scores) are large. Hence, the sum of edge weights of its maximum spanning tree is likely large, which, according to our analysis in Section 5.4.1, signifies a large total correlation score, i.e., mutual correlation.

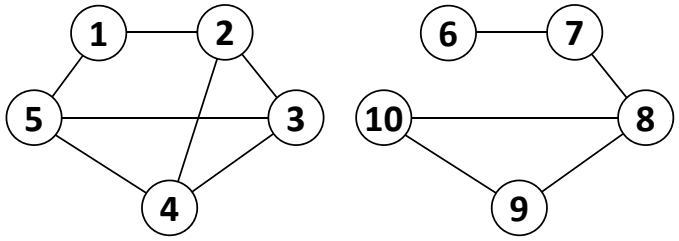
We support this observation by another result of ours as follows. W.l.o.g., consider a group $g = \{C_i\}_{i=1}^{d}$ where every C_i is correlated with every other C_j. We define C_i and C_j to be correlated iff $Corr(C_i, C_j)$ and $Corr(C_j, C_i)$ are large. We discuss how to decide if a correlation score produced by $Corr$ is large in Section 5.5. For now, to show that group g is highly suited for our purposes, we derive the following result.

Claim 1. *$\{C_i\}_{i=1}^{d}$ are likely mutually correlated under different correlation measures.*

This claim is derived from the discussion in Section 4.6, Chapter 4. It implies that for a given group, if every two of its columns are correlated, then its columns are likely mutually correlated. In other words, this group is likely a group of correlated columns. However, real-world data tends to contain noise making perfect pairwise correlation between columns of any group not always happen. In fact, our preliminary empirical analysis points out that sticking to the requirement of perfect pairwise correlation, we would likely end up only with small groups, e.g., those containing two to three columns. To address this issue, we go for a fault-tolerant solution. More specifically, we focus on groups g where each member column is correlated with *most* of the columns in g. We will show in Section 5.5 that such groups permit efficient mining. In addition, they yield very high quality in our experiments. Below we provide a formal definition of approximate groups.

Definition 15. ***Approximate Group**:*
$g = \{C_i\}_{i=1}^{d} \subset \mathcal{C}$ with $d \geq 2$ is an approximate group of correlated columns iff (a) C_i is correlated to at least $\lceil \delta \cdot (d-1) \rceil$ columns in g ($0 < \delta \leq 1$), and (b) no proper superset of g is an approximate group of correlated columns.

In Definition 15, we enforce the maximality requirement of approximate groups in order to eliminate redundancy. We note that DECOREL is not constrained to this



(a) g_1: An approximate group

(b) g_2: *Not* an approximate group

Figure 5.2.: Example of approximate groups ($\delta = 0.5$).

notion of approximate group. Depending on the application scenario, one could go for a tighter notion, e.g., perfect pairwise correlation by setting $\delta = 1$.

Example 9. *Figures 5.2(a) and 5.2(b) depict pairwise correlations of two toy groups* $g_1 = \{C_1, C_2, C_3, C_4, C_5\}$ *and* $g_2 = \{C_6, C_7, C_8, C_9, C_{10}\}$*, respectively. Note that they do* ***not*** *depict independence graphs. In both figures, the convention is that two nodes are connected iff the corresponding columns are correlated. For instance, in group* g_1*,* C_1 *and* C_2 *are correlated. Assume that* $\delta = 0.5$*. According to Definition 15,* g_1 *is an approximate group. On the other hand,* g_2 *does not meet the condition (a) of Definition 15 since* C_6 *is only correlated to one column* C_7 *(the minimum vertex degree is* $\lceil 0.5 \cdot (5-1) \rceil = 2$*). Hence,* g_2 *is not an approximate group. We note that* $\{C_1, C_2, C_3, C_5\}$*,* $\{C_2, C_3, C_4\}$*, and* $\{C_3, C_4, C_5\}$ *are also approximate groups. However, with the maximality requirement, these groups will not be output since they are redundant with respect to* g_1*.*

To solve our problem of detecting groups of correlated columns, for efficiency reasons we mine approximate groups instead. From now on, with groups we mean approximate groups.

5.5. Group Discovery

To mine groups, we have to address three questions. First, since $Corr$ produces real-valued correlation scores, how can we decide if a score is large enough to signify that two columns are correlated? Second, the search space is still potentially exponential to the total number of columns. So how can we efficiently mine groups? Third, the number of groups output may be too large for subsequent processing steps, e.g., when users want to manually inspect the groups for adjusting the query optimizer. Hence, how can we produce a succinct set of groups that facilitates post-analysis? We answer all these questions in this section.

5.5.1. Thresholding correlation scores

$Corr(X, Y)$ is upper-bounded by the entropy/cumulative entropy of X, which in turn is dependent on its value domain. Further, $Corr$ is asymmetric. Thus, we

propose to threshold the correlation scores in each individual column. Our goal is to obtain an adaptive thresholding scheme to facilitate parameterization. We achieve this as follows.

Let $\{(C_i, Corr(C, C_i))\}_{i=1}^{M}$ be the set of correlation scores between C and each other column C_i. W.l.o.g., we assume that the set is sorted in descending order w.r.t. $Corr(C, C_i)$. We define $ind(C) = \arg\max_{i \in [1, M-1]} \frac{Corr(C,C_i)+1}{Corr(C,C_{i+1})+1}$. That is, if we plot the correlation score spectrum against the index $i \in [1, M]$, $ind(C)$ is where there is the biggest jump in the correlation score ratio. We add 1 to both the numerator and the denominator to remove the impact of small correlation scores, which may cause the respective ratios of scores to be unusually large. Let the cutoff threshold be $th(C) = Corr(C, C_{ind(C)})$. We use it as a cutoff to determine which correlation scores $Corr(C, C_i)$ are high. This consequently helps us to identify the columns C_i that are most related to C in terms of $Corr(C, C_i)$, i.e., $Corr(C, C_i)$ is larger than or equal to $th(C)$. In particular, let the set of such columns be $\mathcal{N}(C)$. We identify $\mathcal{N}(C)$ as

$$\mathcal{N}(C) = \{C_i : i \in [1, M] \wedge Corr(C, C_i) \geq th(C)\}$$

This set can loosely be perceived as the set of neighboring columns of C where the proximity is quantified by $Corr$. It is useful for constructing a correlation graph, which leads to the discovery of candidate groups later.

Our thresholding scheme has its intuition from eigenvalue spectrum analysis [JMK09], which shows that using score ratios effectively separates small and zero scores from large ones in a score spectrum, without introducing any additional parameter.

Example 10. *In Figure 5.3, we plot the correlation score spectrum of NATIONKEY in* TPC-H. *We can see that the scores form three distinct clusters. Intuitively, a cutoff should be placed at rank 12. Using our thresholding scheme,* DECOREL *correctly sets* $ind(NATIONKEY)$ *to 12.*

5.5.2. Group mining

We form an undirected column graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $C \in \mathcal{V}$ is a database column. An edge $e \in \mathcal{E}$ exists between two nodes C_i and C_j ($i \neq j$) iff $C_i \in \mathcal{N}(C_j)$ and $C_j \in \mathcal{N}(C_i)$. The resulting $\mathcal{G}$ captures pairwise correlations of columns. In fact, $\mathcal{G}$ is equivalent to the correlation graph presented in Chapter 4.

Given a subset of vertices $S \subseteq \mathcal{V}$, we define the subgraph $\mathcal{G}(S)$ induced by S as the one with vertex-set being S, and edge-set being edges of $\mathcal{E}$ whose both endpoints are in S. In fact, the groups in Definition 15 correspond to quasi-cliques in $\mathcal{G}$ [JP09].

Definition 16. ***Quasi-clique****:*
A subset $S \subseteq \mathcal{V}$ with at least two vertices is a δ-quasi-clique (or quasi-clique for short) of $\mathcal{G}$ iff: (a) every vertex $v \in S$ has a degree in $\mathcal{G}(S)$ of at least $\lceil \delta \cdot (|S| - 1) \rceil$, and (c) no proper superset of S is a δ-quasi-clique.

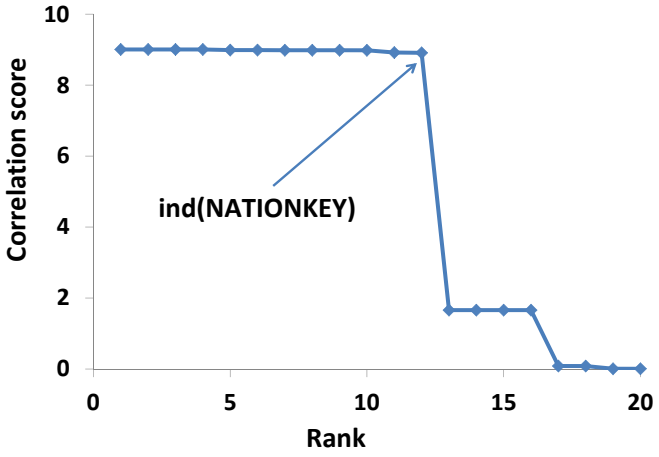


Figure 5.3.: Correlation score spectrum of NATIONKEY in *TPC-H*. According to our method, $ind(NATIONKEY) = 12$.

Considering Definition 15 of approximate groups, the requirement of minimum vertex degree ensures that each column of the group is correlated to most of the other columns. Also, the maximality requirement of S addresses the maximality of the group.

Hence, we mine groups forming quasi-cliques in $\mathcal{G}$. If $\delta = 1$, we end up with cliques, which would yield groups of columns with stronger correlations. However, we expect $\mathcal{G}$ to be sparse. Searching for cliques as in Chapter 4, we might end up only with groups corresponding to two end-points of the same edges. On the other hand, we still need to set δ high enough to ensure the compactness of the groups. Following the proposal in [LW08], we set $\delta = 0.5$. By this, we ensure that the quasi-cliques, and hence groups, are connected tightly. In fact, the problem of mining all quasi-cliques is NP-hard [JP09].

Since $\mathcal{G}$ tends to be sparse, we can address the NP-hardness complexity of the problem by practical algorithms that have good performance on real-world data. Here, we apply the algorithm in [LW08] having several pruning rules to efficiently explore the search space. For instance, when $\delta = 0.5$, the shortest path between any two vertices of a quasi-clique contains at most two edges. The algorithm exploits this observation to reduce the vertices which can be used to extend a candidate quasi-clique. Another pruning is based on the upper and lower bounds of the number of vertices that can be added to a candidate quasi-clique concurrently to form a larger quasi-clique. By these pruning rules, the algorithm is able to eliminate candidate extensions of existing quasi-cliques. Our experiments show that using this algorithm, DECOREL achieves both higher quality and better scalability than existing methods.

After mining groups, we may achieve very many groups, which hinder post-analysis. Our goal is to obtain a succinct set of groups that can be inspected manually. Therefore, we propose to merge similar groups in the subsequent step.

Example 11. *Assume that after the group mining phase, two of the groups that we obtain are* $g_3 = \{C_1, C_2, C_3, C_4\}$ *and* $g_4 = \{C_1, C_3, C_4, C_5\}$*. Since they have many dimensions in common, i.e., they bring about similar information, it would make sense to merge them to create the larger group* $\{C_1, C_2, C_3, C_4, C_5\}$*.*

5.5.3. Group merging

The group merging phase of DECOREL is done similarly to Section 4.7, Chapter 4. That is, we again apply the technique in [MV13]. We highlight that this merge step only merges groups with high overlap. This means that groups with low overlap remain separate. Thus, DECOREL still detects overlapping groups representing different semantics of columns. We ensure this by detecting overlapping groups of correlated columns by first mining quasi-cliques in the column graph $\mathcal{G}$, and then merging groups.

Alternative solutions to DECOREL could first detect connected components in $\mathcal{G}$ and then further decompose the connected components into groups of related columns. This has indeed been used in [ZHO+11]; however, both steps of that solution result in disjoint grouping, which is undesirable for real-world databases.

5.6. Theoretical comparison

In this section, we theoretically compare DECOREL against the state of the art method for grouping relational columns [ZHO+11]. This method, named ECLUS, uses a distribution test based on Earth Mover's Distance (EMD) [LB01] to assess relationships of columns. Our purpose here is to show that DECOREL is more general than ECLUS. Hence, we are able to discover column relationships that ECLUS misses. We now review the background of ECLUS before presenting our analysis.

5.6.1. Review: Distribution test in ECLUS

ECLUS uses EMD to quantify the (dis-)similarity between the marginal distributions of any two columns, e.g., C_1 and C_2. The lower the distance, the more similar C_1 and C_2 are.

Let $|C|$ be the number of distinct values of column C. To compute EMD, ECLUS first forms pmfs on the sets of values of C_1 and C_2 by assigning a probability mass of $1/|C_1|$ and $1/|C_2|$ to each value of C_1 and C_2, respectively. Then it sorts the union of the values of C_1 and C_2 (in lexicographical order for strings, and in numerical order for numerical values) and computes the EMD of two distributions of the *ranks*. Suppose that the distribution of C_1 is $p = \{(x_1, p_1), \ldots, (x_{|C_1|}, p_{|C_1|})\}$ and the distribution of C_2 is $q = \{(y_1, q_1), \ldots, (y_{|C_2|}, q_{|C_2|})\}$. Here, x_i and y_j are ranks, and p_i and p_j are their masses, respectively. ECLUS then instantiates $EMD(C_1, C_2)$

to $EMD(p, q)$. To compute $EMD(p, q)$, ECLUS solves the below optimization problem [ZHO+11]:

$$\text{minimize } W(p,q,F) = \sum_{i=1}^{|C_1|}\sum_{j=1}^{|C_2|} F_{ij}|x_i - y_j| \text{ subject to:}$$

$$F_{ij} \geq 0 \text{ and } \sum_{j=1}^{|C_2|} F_{ij} = p_i \text{ and } \sum_{i=1}^{|C_1|} F_{ij} = q_j$$

$$\sum_{i=1}^{|C_1|}\sum_{j=1}^{|C_2|} F_{ij} = \sum_{i=1}^{|C_1|} p_i = \sum_{j=1}^{|C_2|} q_j = 1 \quad .$$

where F is a joint distribution of the ranks of C_1 and C_2. That is, ECLUS computes $EMD(p, q)$ as the cheapest cost to transform p to q over all joint distributions F whose marginal distributions are p and q.

For efficiency purposes, the ranks may be further discretized into quantile histograms, and the EMD is applied on two such histograms accordingly. Zhang et al. [ZHO+11] show that the EMD test can discover various column relationships: (a) foreign key and primary key constraints, (b) two foreign keys referring to the same primary key, (c) a column in a view and its correspondence in the base table, (d) two columns in two different views but originating from the same column in the base table, and (e) two columns without any explicit association but semantically related through a third column.

However, their method only discovers groups of columns of the same data type and of overlapping value domains. This is too restrictive for real-world databases.

5.6.2. Correlation test is more general

Our claim is that two columns passing the EMD test, i.e., having a low EMD value, are very likely correlated. On the other hand, having have a high EMD value does not imply that they are uncorrelated. In other words, we have:

Claim 2. *Our correlation test is more general than EMD test.*

To show this, we utilize the statistical intuition of EMD. In fact, EMD can be regarded as the Wasserstein metric [Rac84]. Applying the Wasserstein metric to columns C_1 and C_2, their EMD becomes:

$$\min_F \{E_F(|X - Y|) : X \sim p, Y \sim q, (X, Y) \sim F\} \quad .$$

That is, $EMD(C_1, C_2)$ equals to the minimum of the expected difference between their ranks (X and Y, respectively), taken over all possible joint probability distributions F of their ranks such that the marginal distributions of F are p and q. Assume that $|C_1| = |C_2| = n$ and $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$ are sorted in ascending order. It holds that [LB01]: $EMD(C_1, C_2) = \frac{1}{n}\sum_{i=1}^{n} |x_i - y_i|$. Thus, $EMD(C_1, C_2) = 0$ iff

$x_i = y_i$ for every $i \in [n]$. This implies a perfect 1-to-1 mapping between X and Y, and hence, between the values of C_1 and C_2. Thus, both $Corr(C_1, C_2)$ and $Corr(C_2, C_1)$ are likely large, i.e., C_1 and C_2 are correlated. This observation also holds for general cases when $|C_1| \neq |C_2|$. That is, the lower the EMD score, the less cost of transforming p to q, the easier it is to define a 1-to-1 mapping between X and Y. Thus, the more likely it is that C_1 and C_2 are correlated.

All in all, a low EMD score tends to correspond to a high correlation score. Thus, if two columns have a low correlation score, they tend to have a high EMD score, i.e., they are unrelated under the EMD test. However, a high EMD does not say anything about the correlation. This is because EMD does not assess joint distributions while correlation analysis is involved in both marginal and joint distributions (see Section 5.2). As a result, even if two columns have similar marginal distributions, they may be uncorrelated. Thus, by performing correlation analysis, we can detect not only the relationships that an EMD test can find but also the relationships that such a test cannot cover. □

Hence, when the basic schema information is available, ECLUS is restrictive and misses correlations among columns with not only different value domains, but also different data types.

Example 12. *Our experiments reveal that on* TPC-H*,* ECLUS *cannot discover the correlation between the order date and the corresponding order status. This is because the two columns have very different value domains. On* Energy*,* ECLUS *also misses the correlation between the energy consumption indicators of each building (real-valued) and its location (categorical).* DECOREL *nevertheless handles these scenarios.*

In sum, DECOREL assesses the relationships of columns based on their joint distributions and marginal distributions. Thus, it does not require columns to have similar values or data types. This is an important contribution of ours to the area of schema extraction.

5.7. Experiments

In this section, we report our empirical study on the performance of DECOREL. Our objectives are to assess (a) the quality of groups produced by DECOREL, (b) the scalability of DECOREL with the database size as well as the number of columns, and (c) the succinctness of its output. To compute $Corr(X, Y)$ when Y is numerical, we search for the histogram of Y that maximizes the score $Corr(X, Y)$ (see Section 5.3.2).

We use both synthetic and real-world databases. In particular, we generate a synthetic database *SYNTH* containing several relations and columns of different data types (real-valued, discrete, and categorical). We model it according to the *TPC-H* benchmark. However, we additionally embed several correlations, ranging from simple linear to complex non-linear correlations. We use *SYNTH* to quantitatively assess the quality of the correlations detected by DECOREL. Further, we use the

Data	Tables	Rows	Columns	Data Types
SYNTH	8	50,000	40	r, d, c
SYNTH2	8	50,000	40	r, d
TPC-H	8	8,661,245	61	r, d, c
Energy	6	1,486,718	63	r, d, c
Climate	1	35,601	251	r
Census	1	95,130	41	d, c

Table 5.2.: Database characteristics. With data types: 'r' real-valued, 'd' discrete, and 'c' categorical

synthetic *TPC-H* benchmark database itself with scale factor 1 (i.e., about 1GB data).

For real-world data, we use the *Energy* database, which consists of several relations, e.g., Institution, Building, Consumption. It records the energy consumption patterns of the KIT university campus. As aforementioned, *Energy* contains categorical, discrete, and real-valued columns. Thus, it is a good real-world testbed for our evaluation. Another real-world database is *Climate*, which contains indoor climate and energy consumption indicators of a building in Frankfurt, Germany (cf., Chapter 4). *Climate* contains real-valued columns only. In addition, we include *Census*, a publicly available real-world data set from the UCI Machine Learning Repository. Table 5.2 summarizes the characteristics of all databases used.

We compare DECOREL to four state of the art techniques that also group columns in the database area: First, ECLUS groups columns using EMD [ZHO+11]. Second, CORDS groups columns into *pairs* using the χ^2 test [IMH+04]. Since CORDS is not designed for combining those pairs into larger groups of columns, we apply our group mining algorithm on its output pairs. Third, ENTROPY [PLP+10], similarly to CORDS, also finds pairs of correlated columns using a measure similar to ours, but this measure relies merely on Shannon entropy. Like CORDS, we extend ENTROPY to find groups of multiple correlated columns. We also include 4S (cf., Chapter 4) as another competitor.

5.7.1. Quantitative assessment of groups

Assessment based on Precision-Recall. We only use databases where we have prior knowledge on its correlations. In particular, we use *SYNTH* with categorical and numerical (discrete and real-valued) columns. We also generate its variant, named *SYNTH2*, with only numerical (discrete and real-valued) columns. Note that in both databases, the known correlations are involved in columns of different data types and/or different value domains. Further, the ground truth of each database contains overlapping groups of correlated columns.

The results are in Table 5.3. We can see that DECOREL performs very well, outperforming all of its competitors. In contrast, ECLUS has low accuracy since it clusters columns of different data types and non-overlapping value domains into separate

		DECOREL	ECLUS	CORDS	ENTROPY	4S
SYNTH	Prec.	1.00	0.56	0.75	0.78	-
	Rec.	1.00	0.66	0.72	0.78	-
SYNTH2	Prec.	1.00	0.54	0.73	0.70	1.0
	Rec.	1.00	0.67	0.74	0.74	0.99

Table 5.3.: Precision and Recall on synthetic data.

groups although these columns are correlated. Moreover, ECLUS produces disjoint groups, and hence, breaks overlapping groups of correlated columns. The performance of CORDS and ENTROPY is not high probably due to their use of sampling to compute correlation scores. 4S is inapplicable to *SYNTH* since 4S is not designed to handle categorical columns.

Overall, compared to all of its competitors, we see that DECOREL best detects overlapping groups of correlated columns with heterogeneous data types and value domains.

Next, we assess the parameterization of DECOREL w.r.t. ϵ, c, and δ using the SYNTH data set (as this data set contains columns of different data types). For ϵ, we fix $c = 2$ and $\delta = 0.5$. For c, we fix $\epsilon = 1/3$ and $\delta = 0.5$. For δ, we fix $\epsilon = 1/3$ and $c = 2$. The results are in Figure 5.4. We can see that DECOREL achieves the best performance (in terms of both precision and recall) at $\epsilon = 1/3$ and $\epsilon = 0.5$. However, as the former value results in faster runtime (see Section 5.8), we use it in other experiments. Regarding c, we see that DECOREL has fairly stable performance w.r.t. this parameter. We choose to use $c = 2$ as it results in the fastest runtime. Regarding δ, we can see that $\delta = 0.5$ yields the best performance. Thus, we set $\delta = 0.5$ in the rest of this chapter. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable parameterization might be required.

Assessment based on adjustment factor. Here, we use the adjustment factor to assess the results of DECOREL on *all* databases. Intuitively, the adjustment factor should be able to capture correlation. Thus, we define the adjustment factor of a group $g = \{C_i\}_{i=1}^{d}$ as

$$af(g) = \frac{\prod_{i=1}^{d} |C_i|}{|C_1, \ldots, C_d|}$$

where $|C_i|$ is the number of distinct values in C_i and similarly for $|C_1, \ldots, C_d|$. Note that if the columns belong to multiple relations, $|C_1, \ldots, C_d|$ is computed by applying full outer join of the involved relations, as done in [YPS11].

Intuitively, the adjustment factor of a group quantifies how much the joint selectivity $|C_1, \ldots, C_d|$ deviates from the selectivity under the independence assumption. Thus, the more correlated the columns of the group are, the higher its adjustment factor. We define the adjustment factor of a method producing groups $\{g_j\}_{j=1}^{n}$ as $\frac{1}{n}\sum_{j=1}^{n} af(g_j)$, i.e., the average of adjustment factors of its groups. Again, the larger

the adjustment factor of a method, the better the method is in finding groups of correlated columns. In fact, a similar notion of adjustment factor was used in [IMH+04] to rank pairs of correlated columns. Further, it has been suggested that the adjustment factor has an impact on the selectivity estimates of optimizers [IMH+04, TDJ11]. Thus, one can use the adjustment factor as an implicit measure for query optimizers.

The relative adjustment factors of all methods in comparison to DECOREL are in Figure 5.5. Recall that 4S is only applicable to the numerical *Climate* database. The results show DECOREL to achieve the best results, outperforming its competitors up to an order of magnitude. This suggests that DECOREL better discovers groups of correlated columns where the joint distributions deviate more profoundly from the product of the marginal distributions.

As mentioned above, the adjustment factor has an impact on selectivity estimates of optimizers. Thus, by being able to discover dependable groups of correlated columns, one could use DECOREL to reliably identify important column correlations for improving query optimizers. This is one of several possible applications of the groups detected by DECOREL.

5.7.2. Qualitative assessment of groups

We now explore in detail the groups discovered by DECOREL to gain more insight into its performance. We limit our discussion to *TPC-H* and real-world databases where we do not have full knowledge of the hidden correlations.

TPC-H. DECOREL correctly identifies all correlations which are involved in declared foreign key and primary key constraints, regardless of column data types, and place the respective columns into the same group. More than that, DECOREL is able to group columns with no explicit relationships but semantically correlated. For instance, DECOREL puts ORDERDATE and ORDERSTATUS into the same group. In fact, [ZHO+11] has pointed out that the order date and the corresponding order status are correlated. Since these two columns have totally different value domains, ECLUS cannot recognize their relationship. CORDS and ENTROPY in turn tend to place correlated columns where at least one column is real-valued into separate groups. Intuitively, there is a correlation between the total price of an order and the respective customer name. These two techniques however fail to detect it. 4S in turn is inapplicable to *TPC-H* since it only handles numerical columns. In contrast, DECOREL successfully groups correlated columns without being constrained to their data types as well as value domains.

Climate. One of the groups identified by DECOREL reflects a correlation among the air temperature supplied to the heating system, the temperature of the heating boiler, and the amount of heating it produces. While this relation is rather intuitive and expected, ECLUS does not detect it. This could be because the three measures have very different ranges of values: from 10 to 45 for the supplied air temperature, from 25 to 350 for the temperature of the heating boiler, and from 0 to 1100 for the amount of heating produced. ECLUS, by its design, places the three

columns into different groups, and hence, leaves their relationship undetected. CORDS and ENTROPY also do not find this correlation, partly because they both use correlation measures that are more suited to categorical and discrete columns. In addition, they both compute correlation scores using only samples randomly drawn from the data. We observe that the pairwise joint distributions of the three measures are rather skewed. Thus, a simple random sampling as employed by CORDS and ENTROPY likely misses the bigger picture [KGKB03]. Another correlation reflected in the groups discovered by DECOREL but not by any other method, is among the electricity consumed, the amount of drinking water, and the amount of heating. ECLUS misses this correlation, again, because the columns involved have different value ranges. CORDS and ENTROPY also cannot identify this correlation due to their correlation measures. 4S in turn does not detect it because of inaccuracy caused by sketching.

Energy. DECOREL groups columns of energy consumption of buildings with columns related to time (e.g., date-time, semester season, holiday/working day). This is expected as, e.g., more energy is consumed on working days than on holidays. DECOREL also groups consumption columns with the characteristics of buildings, e.g., total area, total number of staff members, and location. Again, this grouping is intuitively accurate: the larger area and the more staff members a building has, the more energy it consumes; buildings located in the campus section where, say, large-scale physics experiments are usually carried out consume more energy than buildings in other locations. However, since these columns have different value domains and even data types, ECLUS puts them into different groups. The quality of groups detected by CORDS and ENTROPY degrades greatly. This could stem from the fact that the consumption indicators are all real-valued. Hence, the statistical power of χ^2 test as well as measures based on Shannon entropy, applied to these columns, becomes weak.

Census. We present some groups DECOREL produces, which are undetected by other methods. We believe them to be interesting from a data analysis point of view. Their details are as follows:

- weeks worked in year, education, reason for unemployment, hispanic origin, member of a labor union, wage per hour
- weeks worked in year, education, citizenship, country of birth father, country of birth mother, capital gains, income level
- marital status, age, education, dividends from stocks, detailed occupation recode

We note that some of the correlations in the groups above intuitively make sense, e.g., education and wage per hour. In addition, some of the remaining correlations, e.g., marital status, age, and education, agree with a previous study [MPY05].

Overall, we see DECOREL to be capable of detecting column relationships regardless of their underlying data types as well as value domains. The relationships

covered by DECOREL range from known ones (e.g., declared foreign key and primary key constraints) to novel ones. The latter can be exploited for, e.g., improving query optimizers and advanced data analysis.

5.7.3. Scalability

We assess scalability of DECOREL to the database size using the *TPC-H* benchmark and the number of columns using *Census*. For the former, we vary the scale factor, and hence, effectively scale the number of tuples. For the latter, we scale the number of columns by appending the identical *Census* database for a number of times. The results are in Figures 5.6 and 5.7. For readability, we use a logarithmic scale on *both* axes for Figure 5.6.

We see that DECOREL scales linearly to the database size, i.e., the number of tuples. It has quadratic scalability to the number of columns. Moreover, DECOREL achieves better scalability than ECLUS in both tests. DECOREL also scales better than CORDS and ENTROPY even though they only work on samples of the data. This highlights the benefits of our efficient computation of pairwise correlations and our efficient mining of correlated groups. We are unable to display the runtimes of 4S since 4S is inapplicable to both *TPC-H* and *Census* due to their categorical columns. However, for numerical data such as *Climate*, we have observed that DECOREL shows runtimes similar to 4S. Thus, we can perform both categorical and numerical data assessment in a very scalable manner.

Overall, DECOREL is able to tackle the exponential search space of groups in polynomial time, and hence, enables correlation analysis for large databases.

5.7.4. Succinctness of output

We study the benefits of the MDL group merging on DECOREL. Our performance metric is the reduction ratio, i.e., $\frac{m}{m'}$ where m and m' are the number of groups before and after merging. The results are in Figure 5.8. We see that the group merging phase achieves up to an order of magnitude reduction ratio. This shows that by merging groups, DECOREL produces a succinct set of overlapping groups while still guaranteeing their quality (according to the above experiments).

5.8. Searching for the optimal histogram

We show the derivation for when X is categorical. The derivation for when X is numerical can be found in Chapter 3. The materials presented in the following are based on Chapter 3 and [RRF+11].

Let g be a histogram of Y. We denote the number of bins of g as $|g|$. We write Y^g as Y discretized by g. Following [RRF+11], we restrict that $|g| < N^\epsilon$ where N is the number of tuples in the joint distribution of X and Y, and $\epsilon \in (0, 1)$. We formulate the following problem: *Find the histogram g of Y with $|g| < N^\epsilon$ that minimizes $H(X \mid Y^g)$.*

We prove that our optimization problem can be solved by dynamic programming. In particular, w.l.o.g., let $Y(1) \leq \ldots \leq Y(N)$ be realizations of Y. Further, let

$$Y(j,m) = \{Y(j), Y(j+1), \ldots, Y(m)\}$$

where $j \leq m$. Slightly abusing notation, we write $Y(1,N)$ as Y. We use $H(X \mid \langle Y(j,m)\rangle)$ to denote $H(X)$ computed using the $(m-j+1)$ tuples of the joint distribution corresponding to $Y(j)$ to $Y(m)$, projected onto X. To show that the optimal discretization of Y minimizing $H(X \mid Y)$ can be searched by dynamic programming, we introduce the following formulation which will subsequently lead to the solution of our problem. In particular, for $1 \leq l \leq m \leq N$, we write

$$f(m,l) = \min_{g:|g|=l} H(X \mid Y^g(1,m))$$

where g is a histogram of $Y(1,m)$ with l bins, and $Y^g(1,m)$ is the discretized version of $Y(1,m)$ by g. That is, $f(m,l)$ is the minimum $H(X \mid Y(1,m))$ over all discretization g of $Y(1,m)$ into l bins. For $1 < l \leq m \leq N$, we prove the following recursive formulation of $f(m,l)$, which gives way to efficiently computing it using dynamic programming, and hence, to efficiently solving our problem of minimizing $H(X \mid Y)$.

Theorem 16. $f(m,l) = \min_{j \in [l-1,m)} A_j$ *where*
$A_j = \frac{j}{m} f(j,l-1) + \frac{m-j}{m} H(X \mid \langle Y(j+1,m)\rangle)$.

Proof. Let $g^* = \arg\min_{g:|g|=l} H(X \mid Y^g(1,m))$. We denote l bins that g^* generates on Y as $b_1, \ldots, b_l$. We write $|b_t|$ as the number of values of Y in b_t. Further, let $c_z = \sum_{i=1}^{z} |b_t|$. Note that each bin of Y is non-empty, i.e., $c_z \geq z$. We use $H(X \mid b_t)$ to denote $H(X)$ computed using the tuples of the joint distribution corresponding to the realizations of Y in b_t, projected onto X. We have: $f(m,l)$

$$\begin{aligned}
&= \sum_{t=1}^{l} \frac{|b_t|}{m} H(X \mid b_t) \\
&= \sum_{t=1}^{l-1} \frac{|b_t|}{m} H(X \mid b_t) + \frac{|b_l|}{m} H(X \mid b_l) \\
&= \frac{c_{l-1}}{m} \sum_{t=1}^{l-1} \frac{|b_t|}{c_{l-1}} H(X \mid b_t) + \frac{|b_l|}{m} H(X \mid b_l) \\
&= \frac{c_{l-1}}{m} f(c_{l-1}, l-1) \\
&\quad + \frac{m - c_{l-1}}{m} H(X \mid \langle Y(c_{l-1}+1, m)\rangle) \quad .
\end{aligned}$$

In the last line, $\sum_{t=1}^{l-1} \frac{|b_t|}{c_{l-1}} H(X \mid b_t)$ is equal to $f(c_{l-1}, l-1)$ because otherwise, we could decrease $f(m,l)$ by choosing a different histogram of $Y(1, c_{l-1})$ into $l-1$ bins. This in turn contradicts our definition of $f(m,l)$. Since $c_{l-1} \in [l-1, m)$ and $f(m,l)$ is minimal over all $j \in [l-1, m)$, we arrive at the final result. □

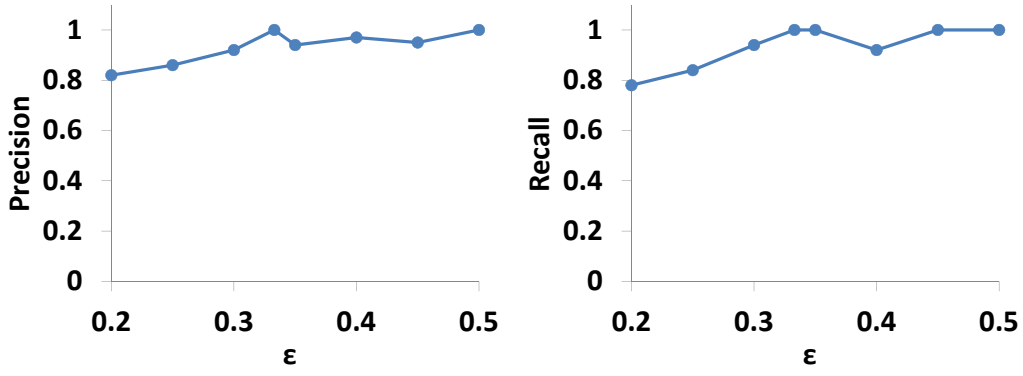
Theorem 16 shows that the optimal histogram of $Y(1,m)$ can be derived from that of $Y(1,j)$ with $j < m$. This allows us to design a dynamic programming algorithm to find the optimal histogram g of Y with $|g| < N^\epsilon$. To further boost efficiency, similarly to Chapter 3, we limit the number of cut points of Y to $c \times N^\epsilon$ with $c > 1$. We do this using equal-frequency discretization on Y with the number of bins equal to $(c \times N^\epsilon + 1)$. Regarding parameter setting, we fix $\epsilon = 0.333$ and $c = 2$ according to our preliminary analysis. Hence, the total time complexity of our histogram search is $O(N^{3\epsilon}) = O(N)$.

5.9. Conclusions

In this chapter, we have proposed DECOREL to mine groups of correlated columns in databases with mixed data types. In short, DECOREL employs the search scheme of 4S (with some modifications) to achieve high scalability. To handle mixed data types, we propose a new correlation measure which can be perceived as a combination of CMI++ and mutual information. Experiments on both synthetic and real-world databases show DECOREL to discover groups of higher quality than existing methods. Moreover, DECOREL scales better than its competitors with both the database size and the number of columns. This suggests that DECOREL is a very promising tool for large-scale correlation analysis on real-world databases.

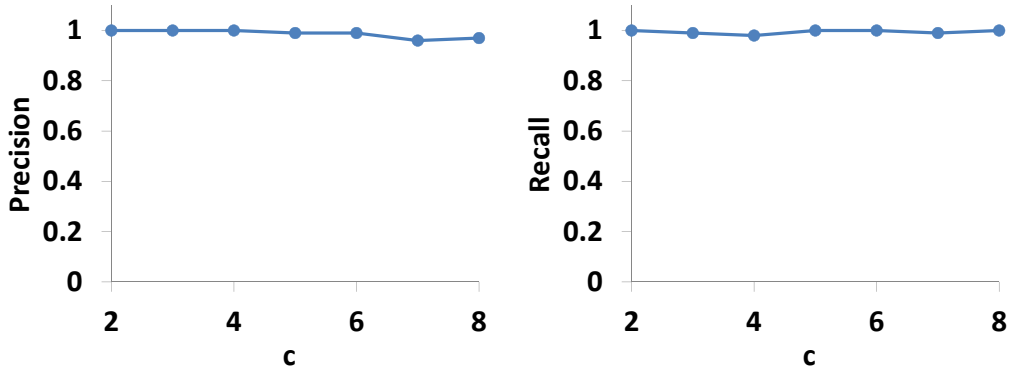
So far, we studied the discovery of correlated subspaces in multivariate data, which constitutes the first part of this thesis. In particular, we introduced novel correlation measures and new subspace search schemes which are able to handle large and high dimensional data sets. The techniques presented so far work mainly with cumulative distribution functions. While this facilitates computation on empirical data, it is also interesting to study correlation measures that favor probability mass functions, e.g., measures based on Shannon entropy. This is because such measures already have wide applications [RRF+11, CGA+01, SCH10, SCRR11, KN03, TDJ11]. Any of their developments will thus bring about high impact across various areas.

As mentioned in Chapter 1, one of the popular ways in data mining to compute measures based on Shannon entropy is to discretize (real-valued) data. In Chapters 3 and 5, we have demonstrated that correlation-aware discretization can successfully do the job. In the next part of the thesis, we further this idea and propose a correlation-aware discretization method to compute total correlation. Please note that this new part does not serve as a replacement for what has been discussed so far. It instead offers another facet of the multi-faceted problem of correlation analysis.



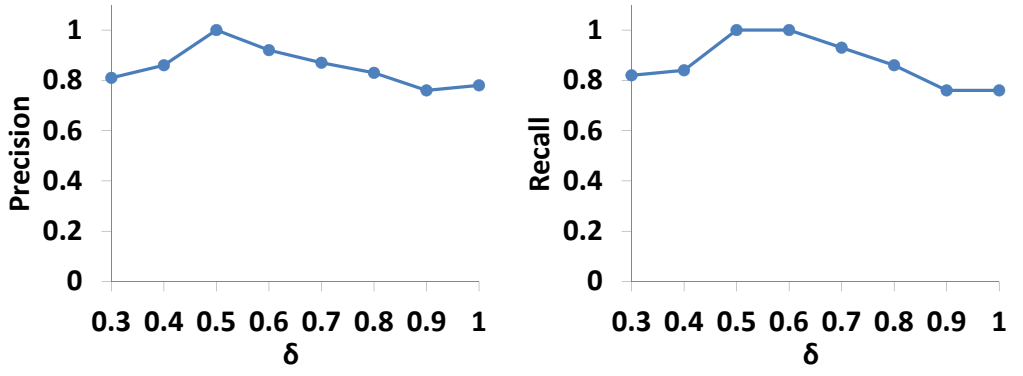
(a) Precision vs. ϵ

(b) Recall vs. ϵ



(c) Precision vs. c

(d) Recall vs. c



(e) Precision vs. δ

(f) Recall vs. δ

Figure 5.4.: [Higher is better] Sensitivity of DECOREL to ϵ, c, and δ on the SYNTH data set.

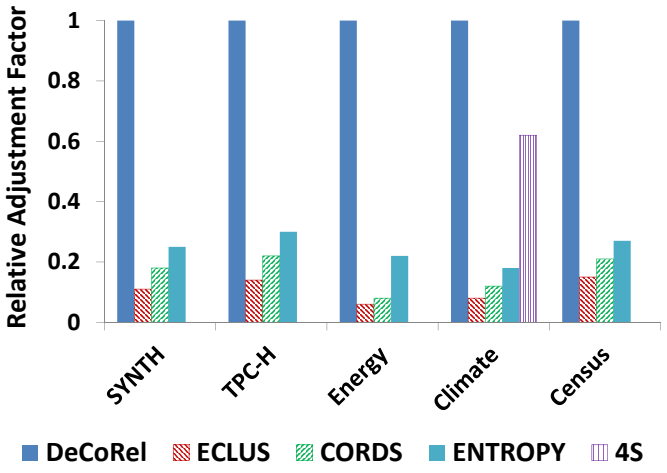


Figure 5.5.: Relative adjustment factor compared to DeCoRel.

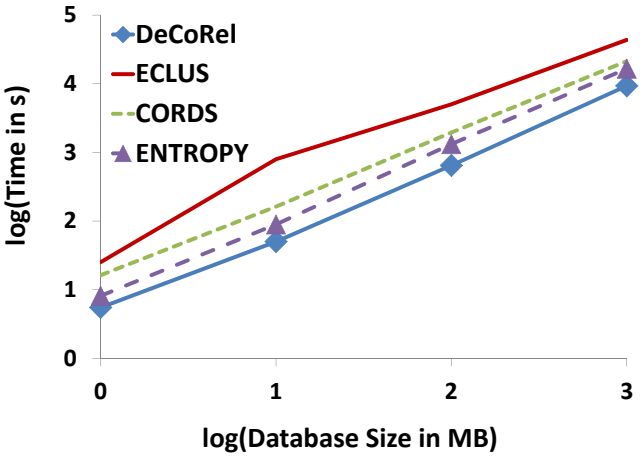


Figure 5.6.: Scalability to database size using *TPC-H*.

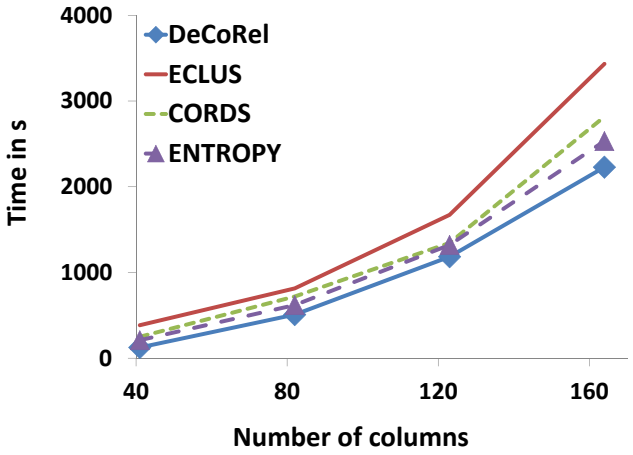


Figure 5.7.: Scalability to the number of columns using *Census*.

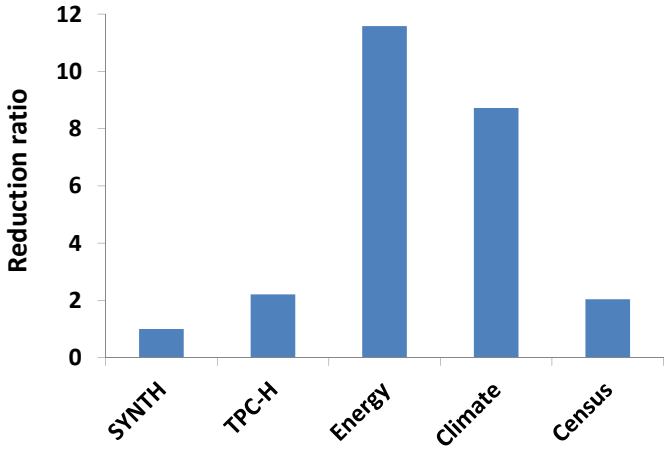


Figure 5.8.: Reduction ratio of the group merging.

Part IV.

Computing Total Correlation

6. Multivariate Maximal Correlation Analysis

This chapter is based on our work originally published as [NMV+14]:

H. V. Nguyen, E. Müller, J. Vreeken, P. Efros, and K. Böhm, *Multivariate maximal correlation analysis*, in ICML, 2014, pp. 775-783.

In Chapters 3 to 5, we propose new correlation measures and use them to mine correlated subspaces in multivariate data. In this chapter, we focus on the computation of total correlation, an existing well-known correlation measure based on Shannon entropy, using discretization. As opposed to naïve discretization, our goal here is to develop a correlation-aware discretization technique to compute total correlation. Recalling from Chapters 3 and 5, by correlation-aware, we mean that the technique should preserve correlations in the data. In Chapter 3, the preservation is w.r.t. CMI++. In Chapter 5, the preservation is w.r.t. pairwise CMI++ and mutual information. In the context of this chapter, the preservation is w.r.t. total correlation. Correlation-aware discretization has been proposed to compute mutual information [RRF+11]. Yet, we will show that directly adopting this technique is not sufficient for computing total correlation. Thus, our problem to this end still is open. But please note that the contributions of this chapter are beyond computing total correlation. In particular, we propose multivariate maximal correlation analysis, which we generalize from various existing methods for correlation analysis. Our handling of total correlation is an instantiation of this general notion.

More in particular, we go for maximal correlation analysis because it does not require assumptions on data distributions, can detect non-linear correlations, is very efficient, and robust to noise. Maximal correlation analysis is our generalization of a number of powerful correlation measures that, in a nutshell, discover correlations hidden in data by (1) looking at various admissible transformations of the data (e.g., discretizations [RRF+11], measurable mean-zero functions

[BF85]), and (2) identifying the maximal correlation score (e.g., mutual information [RRF+11], Pearson's correlation coefficient [BF85]) correspondingly. The key reason why these measures first transform the data is that otherwise only simple correlations can be detected: kernel transformations allow non-linear structures to be found that would go undetected in the original data space [BF85, HSST04]. In contrast, more complex measures such as mutual information can detect complex interactions without transformation, at the expense of having to assume and estimate the data distribution [Yin04]. However, Reshef et al. [RRF+11] showed that instead of making assumptions, we should use the discretizations that yield the largest mutual information. Our computation of total correlation in this chapter in fact is inspired from the work of Reshef et al. Before providing the details of our solution, we first formalize maximal correlation analysis.

6.1. Maximal Correlation Analysis

Above we sketch out important characteristics of maximal correlation analysis. On a formal level, we define this approach as follows.

Definition 17. ***Maximal Correlation – General Form:***
The maximal correlation of real-valued random variables $\{X_i\}_{i=1}^{D}$ is defined as:

$$Corr^*(X_1, \ldots, X_D) = \max_{f_1,\ldots,f_D} Corr(f_1(X_1), \ldots, f_D(X_D))$$

with $Corr$ being a correlation measure, $f_i : dom(X_i) \to \mathcal{A}_i$ being from a pre-specified class of functions, $\mathcal{A}_i \subseteq \mathbb{R}$.

That is, maximal correlation analysis discovers correlations in the data by searching for the permissible transformations f_i's of X_i's that maximize their correlation (measured by $Corr$). Following Definition 17, to search for maximal correlation, we need to solve an optimization problem over a search space whose size is potentially exponential to the number of dimensions. The search space in general does not exhibit structure that we can exploit for an efficient search. Thus, it is infeasible to examine it exhaustively, which makes maximal correlation analysis on multivariate data very challenging. Avoiding this issue, most existing work focuses on *pairwise* maximal correlations. More details are given below.

6.1.1. Instantiations of maximal correlation

Breiman and Friedman [BF85] defined the maximal correlation between two real-valued random variables X and Y as $\rho^*(X, Y) = \max_{f_1,f_2} \rho(f_1(X), f_2(Y))$ with $Corr = \rho$ being the Pearson's correlation coefficient, and $f_1 : \mathbb{R} \to \mathbb{R}$ and $f_2 : \mathbb{R} \to \mathbb{R}$ being two measurable mean-zero functions of X and Y, respectively. If f_1 and f_2 are non-linear functions, their method can find non-linear correlations.

Likewise, Rao et al. [RSX+11] searched for $a, b \in \mathbb{R}$ that maximize $Corr(X, Y) = U(aX + b, Y)$, which equals to $\left|\int \kappa(ax + b - y)(p(x, y) - p(x)p(y))dxdy\right|$ where κ

is a positive definite kernel function ($p(X)$, $p(Y)$, and $p(X,Y)$ are the pdfs of X, Y, and (X,Y), respectively). f_1 is a linear transformation function, and f_2 is the identity function. If κ is non-linear, they can find non-linear correlations.

Canonical correlation analysis (CCA) [Hot36, HSST04, AABL13, CKKZ13], instead of analyzing two random variables, considers two data sets of the same size. That is, $X \in \mathbb{R}^A$ and $Y \in \mathbb{R}^B$ represent two groups of random variables. CCA looks for (non-)linear transformations of this data such that their correlation, measured by $Corr$, is maximized. In [Yin04], $Corr$ is the mutual information, $f_1 : \mathbb{R}^A \to \mathbb{R}$ and $f_2 : \mathbb{R}^B \to \mathbb{R}$ are linear transformations. $Corr(f_1(X), f_2(Y))$ is computed by density estimation. Along this line, Generalized CCA [Car68, Ket71] is an extension of CCA to multiple data sets. Its focus so far, however, is on linear correlations [vdVT12].

Maximal Information Coefficient (MIC) [RRF+11] analyzes the correlation of X and Y by identifying the discretizations of X and Y that maximize their mutual information, normalized according to their numbers of bins. Here, $Corr$ is the normalized mutual information. f_1 and f_2 are functions mapping values of $dom(X)$ and $dom(Y)$ to $\mathcal{A}_1 = \mathbb{N}$ and $\mathcal{A}_2 = \mathbb{N}$ (with counting measures), respectively. Note that, MIC is applicable to CCA computation where mutual information is used [Yin04]. However, we note that MIC cannot be adapted to computing total correlation. To illustrate this, consider a toy data set with three dimensions $\{A, B, C\}$. MIC can find two *separate* ways to discretize B to maximize its correlation with A and C, but it cannot find a discretization of B such that the correlation with regard to *both* A and C is maximized. Thus, MIC is not suited for calculating correlations over more than two dimensions. Further, directly adapting the solution of MIC to the multivariate setting suffers from high computational cost. We will get back to this issue in Section 6.3.

In addition to the above measures, we note that CMI++ and the correlation measure of DECOREL are indeed related to MIC, and hence, to maximal correlation analysis. That is, they both also search for the discretization of the data to maximize their respective correlation score.

Having introduced our general notion of maximal correlation analysis, in the following, we present our instantiation of this notion to compute total correlation. We highlight that our solution in this chapter is related to CMI++. We name it as MAC for **M**ultivariate M**a**ximal **C**orrelation Analysis. Nonetheless, despite the close relationship between MAC and CMI++, they are not entirely exclusive, and in fact, are not meant to be like that. Instead, they just study different aspects of information-theoretic correlation analysis. More in particular, MAC targets at Shannon entropy while CMI++ targets at cumulative entropy. Before going to solve MAC, we first present its theory in the next section.

6.2. Theory of MAC

In this section, we discuss the theoretical model of MAC. Consider a d-dimensional data set $\mathbf{DB}$ with real-valued dimensions $\{X_i\}_{i=1}^{D}$ and N data points. We regard

each dimension X_i as a random variable, distributed according to pdf $p(X_i)$. Mapping MAC to Definition 17, we have that $Corr$ is the normalized total correlation (see below), and $f_i : dom(X_i) \to \mathbb{N}$ (with counting measure) is a discretization of X_i. By definition, the total correlation of $\{X_i\}_{i=1}^D$, i.e., of the data set $\mathbf{DB}$, written as $T(\mathbf{DB})$, is $T(\mathbf{DB}) = \sum_{i=1}^{D} H(X_i) - H(X_1, \ldots, X_D)$ where $H(.)$ is the Shannon entropy. As mentioned in Chapter 2:

- $T(\mathbf{DB}) \geq 0$ with equality iff $\{X_i\}_{i=1}^D$ are statistically independent.
- $T(\mathbf{DB}) > 0$ when the dimensions of $\mathbf{DB}$ exhibit any mutual correlation, regardless of the particular correlation types.

As aforementioned, to compute total correlation on real-valued data, a common practice in data mining is to use discretization. Following this line, MAC aims at correlation-aware discretization whose advantages have been discussed in Chapters 3 and 5.

In particular, let g_i be a discretization of X_i into $n_i = |g_i|$ bins. We will refer to n_i as the *grid size* of X_i. We write $X_i^{g_i}$ as X_i discretized by g_i. We call $G = \{g_1, \ldots, g_D\}$ a D-dimensional grid of $\mathbf{DB}$. For mathematical convenience, we focus only on grids G with $n_i \geq 2$. This has been shown to be effective in capturing complex patterns in the data, as well as detecting independence [RRF+11]. The product of grid sizes of G is $|G| = n_1 \times \ldots \times n_D$. We write $\mathbf{DB}^G$ as $\mathbf{DB}$ discretized by G. The grid G induces a probability mass function on $\mathbf{DB}$, i.e., for each cell of G, its mass is the fraction of $\mathbf{DB}$ falling into it. The total correlation of $\mathbf{DB}$ given G becomes $T(\mathbf{DB}^G) = \sum_{i=1}^{D} H(X_i^{g_i}) - H(X_1^{g_1}, \ldots, X_D^{g_D})$. For maximal correlation analysis, one could find an optimal grid G for $\mathbf{DB}$ such that $T(\mathbf{DB}^G)$ is maximized. However, the value of $T(\mathbf{DB}^G)$ is dependent on $\{n_i\}_{i=1}^D$:

Theorem 17. $T(\mathbf{DB}^G) \leq \sum_{i=1}^{D} \log n_i - \max(\{\log n_i\}_{i=1}^D)$.

Proof. We have:

$$T(\mathbf{DB}^G) = \sum_{i=2}^{D} H(X_i^{g_i}) - H(X_i^{g_i} \mid X_1^{g_1}, \ldots, X_{i-1}^{g_{i-1}}) \quad .$$

By not considering the subtracted terms $H(X_i^{g_i} \mid X_1^{g_1}, \ldots)$, we have $T(\mathbf{DB}^G) \leq \sum_{i=2}^{D} H(X_i^{g_i})$. Since $H(X_i^G) \leq \log(n_i)$, we arrive at $T(\mathbf{DB}^G) \leq \sum_{i=2}^{D} \log(n_i)$. Considering all permutations of $X_1, \ldots, X_D$, it holds that

$$T(\mathbf{DB}^G) \leq \sum_{i=1}^{D} \log(n_i) - \max_{1 \leq i \leq D} \log(n_i) \quad .$$

□

As our goal is to achieve an unbiased optimization for unbiased correlation analysis, we need to normalize $T(\mathbf{DB}^G)$ according to the grid sizes of G. Hence, we propose to maximize

$$T_n(\mathbf{DB}^G) = \frac{T(\mathbf{DB}^G)}{\sum_{i=1}^{D} \log n_i - \max(\{\log n_i\}_{i=1}^{D})} \tag{6.1}$$

which we name *the normalized total correlation*. From the fact that $T(\mathbf{DB}^G)$ is non-negative and Theorem 17, we arrive at $T_n(\mathbf{DB}^G) \in [0, 1]$. However, maximizing $T_n(\mathbf{DB}^G)$ is not enough. Consider the case where each dimension has N distinct values. If we discretize each dimension into N bins, then $T_n(\mathbf{DB}^G)$ becomes 1, i.e., maximal. To avoid this trivial binning, we need to impose the maximum product of grid sizes B of the grids G considered. For pairwise correlation ($d = 2$), Reshef et al. [RRF+11] prove that $B = N^{1-\epsilon}$ with $\epsilon \in (0, 1)$. As generalizing this result to the multivariate case is beyond the scope of our work, we adopt it, and hence, restrict $n_i \times n_j < N^{1-\epsilon}$ for $i \neq j$. As a result, we define MAC($\mathbf{DB}$) as follows:

Definition 18. ***Maximal Correlation**:*
MAC($\mathbf{DB}$) *is given as:*

$$\text{MAC}(\mathbf{DB}) = \max_{\substack{G=\{g_1,\ldots,g_D\} \\ \forall i \neq j : n_i \times n_j < N^{1-\epsilon}}} T_n(\mathbf{DB}^G) \quad . \tag{6.2}$$

That is, MAC($\mathbf{DB}$) is equal to the maximum normalized total correlation over all grid $G = \{g_1, \ldots, g_D\}$ of $\mathbf{DB}$ where $n_i \times n_j < N^{1-\epsilon}$ (to avoid inflated scores). We will write MAC($\mathbf{DB}$) and MAC($X_1, \ldots, X_D$) interchangeably.

We have MAC($\mathbf{DB}$) $\in [0, 1]$. When MAC($\mathbf{DB}$) $= 0$, we consider $\{X_i\}_{i=1}^{D}$ to be statistically independent. Due to insufficient sample sizes, the theoretical zero score might not happen in practice.

Theorem 18. *If $2^D > N$ then for any $\{n_i\}_{i=1}^{D}$ with $n_i \geq 2$, we have* MAC($\mathbf{DB}$) > 0.

Proof. We start by observing that $n_1 \times \ldots \times n_D \geq 2^D > N$. For every set of grid sizes $\{n_i\}_{i=1}^{D}$, there is a discretization $G = \{g_1, \ldots, g_D\}$ such that each dimension X_i is divided into $n_i = |g_i|$ equal-frequency bins. Such a strategy yields

$$T(\mathbf{DB}^G) = \sum_{i=1}^{D} \log(n_i) - H(X_1^{g_1}, \ldots, X_D^{g_D}) \quad .$$

The resulting D-dimensional space has

$$n_1 \times \ldots \times n_D > N$$

D-dimensional hypercubes. Allocating N points into these hypercubes, the maximum number of non-empty ones is N. Thus:

$$H(X_1^{g_1}, \ldots, X_D^{g_D}) \leq \log(N) \quad .$$

Hence: $T(\mathbf{DB}^G) \geq \log \frac{n_1 \times \ldots \times n_D}{N} > 0$. Therefore: $T_n(\mathbf{DB}^G) > 0$. Since $\text{MAC}(\mathbf{DB}) \geq T_n(\mathbf{DB}^G)$, the theorem is proved. □

Theorem 18 implies that, when $2^D > N$, there will always be some correlation in the data, i.e., MAC is always positive. This is also known as the curse of dimensionality. For real-world data sets, the issue is very common. Nevertheless, a low score always indicates a low mutual correlation of $\{X_i\}_{i=1}^D$, and vice versa. We will show in Section 6.5 that MAC performs very well in analyzing multivariate data.

6.3. Calculating MAC: Naïve Approach

To use MAC in practice, we need to compute it efficiently. Let us consider naïvely extending the strategy that MIC uses. To approximate the optimal discretization of two dimensions, MIC employs a heuristic: for every equal-frequency discretization of a dimension, it searches for the discretization over the other dimension that maximizes the normalized mutual information.

Naïvely extending this to the multivariate case, for every set of grid sizes $\{n_i\}_{i=1}^D$ we would partition each set of $(D-1)$ dimensions into equal-frequency bins. We would then try to find the optimal discretization of the remaining dimension. For every set $\{n_i\}_{i=1}^D$, we repeat this per dimension, and report the maximum over these D values.

However, by using $n_i \times n_j < N^{(1-\epsilon)}$ for any $i \neq j$, one can prove that $n_1 \times \ldots \times n_D < N^{(1-\epsilon)D/2}$. Hence, we know the size of the search space is $O(N^D)$—which implies this scheme is infeasible for high dimensional data. In fact, even for two dimensions MIC already faces efficiency issues [RRF+11].

6.4. Calculating MAC: Our Approach

We propose a simple and efficient greedy method for estimating MAC. To compute $\text{MAC}(\mathbf{DB})$, one typically has to find concurrently the discretizations of all dimensions that maximize their normalized total correlation $T_n(\mathbf{DB}^G)$ (see Equation (6.1) and (6.2)), which is the source of the computational intractability. Our intuition is to *serialize* this search. That is, step-by-step we find the dimension and its discretization that maximizes its normalized total correlation with all the dimensions already selected and discretized. In particular, we first identify two dimensions X_1' and X_2' such that $\text{MAC}(X_1', X_2')$ is maximal among all pairs of dimensions. Then, at each subsequent step $k \in [2, D-1]$, let $C_k = \{X_1', \ldots, X_k'\}$ be the set of dimensions already picked and discretized. We aim to

- identify the dimension X'_{k+1} that is most likely correlated with C_k *without* having to pre-discretize X'_{k+1}, and
- find the discretization of X'_{k+1} yielding the MAC score of X'_{k+1} and C_k.

Finally, we approximate MAC($\mathbf{DB}$) using the grid G obtained. From now on, when using Shannon entropy, we imply the involved dimensions have been discretized, e.g., we leave the superscript and write X_i for $X_i^{g_i}$. The details of our method are as follows.

6.4.1. Identifying and discretizing X'_1 and X'_2

We compute MAC(X, Y) for every pair of dimensions (X, Y) and pick (X'_1, X'_2) with the largest MAC score.

To compute MAC(X, Y), for each pair of grid sizes (n_X, n_Y) with $n_X n_Y < N^{1-\epsilon}$, we maximize $H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$. Note that, one could solve this through MIC. However, since MIC fixes one dimension to equal-frequency bins before discretizing the other dimension, we conjecture that the solution of MIC is suboptimal. Instead, we compute MAC(X, Y) by cumulative entropy (see Chapter 3). In particular, $h(X) - h(X \mid Y)$ also captures the correlation between X and Y. Therefore, by maximizing $h(X) - h(X \mid Y)$, we maximize the correlation between X and Y, and hence, intuitively maximizes $H(X) - H(X \mid Y)$.

Maximizing $h(X) - h(X \mid Y)$ in fact is equivalent to minimizing $h(X \mid Y)$. The latter can be solved by searching for the optimal discretization of Y (cf., Chapter 3). Hence, we achieve the optimal discretization of Y that intuitively maximizes $H(X) - H(X \mid Y)$. As we need grid sizes for normalization, we apply our solution in CMI++ to find the optimal discretizations of Y at different grid sizes n_Y. As $n_X \geq 2$, we only consider $n_Y < N^{1-\epsilon}/2$.

We apply the same optimization process for $h(Y) - h(Y \mid X)$. Then, we combine the optimal discretizations of both X and Y where $n_X n_Y < N^{1-\epsilon}$. We compute MAC(X, Y) accordingly. We identify (X'_1, X'_2) as the pair of dimensions with the largest MAC score.

6.4.2. Efficient heuristic to identify X'_{k+1}

In practice, one could identify X'_{k+1} and its optimal discretization concurrently by computing MAC(X, C_k) for every dimension X left, and select the dimension with the best MAC score as X'_{k+1}. Note that since all dimensions in C_k have already been discretized, we do not discretize them again. We prove in Section 6.4.3 that MAC(X, C_k) can be solved by dynamic programming. Yet, doing this for each and every dimension X left may become inefficient for high dimensional data. Thus, our intuition here is to first heuristically identify X'_{k+1} and then find its optimal discretization.

In particular, let n'_i be the grid size of $X'_i \in C_k$. From Equation (6.2), it follows that to compute MAC(X, C_k), we need to maximize

$$\frac{\left(\sum_{i=1}^{k} H(X'_i)\right) + H(X) - H(X \mid C_k) - H(C_k)}{\log n + \sum_{i=1}^{k} \log n'_i - \max(\{\log n\} \cup \{\log n'_i\}_{i=1}^{k})} \tag{6.3}$$

where $n < \frac{N^{(1-\epsilon)}}{\max(\{n'_i\}_{i=1}^{k})}$ and X is discretized into n bins. Our goal is to efficiently identify X'_{k+1} without having to discretize all dimensions left. In order to achieve this, we need an objective function that is free of the bin size of each candidate dimension X. Thus, we consider the following term

$$\frac{\left(\sum_{i=1}^{k} H(X'_i)\right) + h(X) - h(X \mid C_k) - H(C_k)}{h(X) + \sum_{i=1}^{k} \log n'_i - \max(\{\log n'_i\}_{i=1}^{k})} \quad . \tag{6.4}$$

Informally speaking, we can regard both Equation (6.3) and (6.4) to represent the normalized mutual correlation of X and all dimensions in C_k. To show that we can use one to replace the other for optimization purposes, we prove in the following theorem that they indeed have very similar properties.

Theorem 19. *Equation* (6.3) *and* (6.4) *have the following properties:*

(1) Their values are in $[0, 1]$.

(2) They are both equal to 0 iff (discretized) X *and all dimensions in* C_k *are statistically independent.*

(3) They are both maximal when there exists $X'_i \in C_k$ *such that (discretized)* X *and all dimensions in* $C_k \setminus \{X'_i\}$*, each is a function of* X'_i.

Proof. For the first property, Equation (6.3) is the normalized total correlation of discretized X and all dimensions in C_k, so its value is in $[0, 1]$. Considering Equation (6.4), from $h(X) \geq h(X \mid C_k)$, we have

$$\left(\sum_{i=1}^{k} H(X'_i)\right) + h(X) - h(X \mid C_k) - H(C_k) \geq 0 \quad .$$

Further, from $h(X \mid C_k) \geq 0$

$$\left(\sum_{i=1}^{k} H(X'_i)\right) + h(X) - h(X \mid C_k) - H(C_k)$$
$$\leq h(X) + \sum_{i=1}^{k} \log n'_i - \max(\{\log n'_i\}_{i=1}^{k}) \quad .$$

Therefore, the value of Equation (6.4) is in $[0, 1]$.

For the second property, from Theorem 1, it holds that Equation (6.3) is equal to zero iff discretized X and all dimensions in C_k are statistically independent. Also, Equation (6.4) is equal to zero iff all the dimensions in C_k are statistically independent, *and* X is independent of C_k; hence,

$$p(X, C_k) = p(X)p(C_k) = p(X)p(X'_1) \cdots p(X'_k)$$

Thus, X and all dimensions in C_k are statistically independent. We therefore have proven the second property.

For the third property, it holds that Equation (6.3) is maximal when there exists $X'_i \in C_k$ such that discretized X and all dimensions in $C_k \setminus \{X'_i\}$, each is a function of X'_i. Considering Equation (6.4), it is maximal when (a) there exists $X'_i \in C_k$ such that all dimensions in $C_k \setminus \{X'_i\}$, each is a function of X'_i, and (b) X is a function of C_k. These two conditions imply X is also a function of X'_i. Thus, we have the third property proven. □

Therefore, instead of solving Equation (6.3) for every n and every X to obtain X'_{k+1}, we propose to use Equation (6.4) as a surrogate indicator of how likely a dimension X is indeed X'_{k+1} (the larger the indicator, the better). This indicator has three advantages: (a) it does not require us to discretize X, (b) it is independent of grid size n, and (c) it can be computed much more efficiently (see Chapter 3). Note that we are not restricted to this heuristic: If there are enough computational resources, one can just skip this step and run the solution in Section 6.4.3 for every dimension X not yet processed.

6.4.3. Discretizing X'_{k+1}

For readability, we use X to denote X'_{k+1} in this section. To find the optimal discretization of X, for each grid size n, we find the respective discretization of X that maximizes $H(X) - H(X, C_k)$; we ignore $\left(\sum_{i=1}^{k} H(X'_i)\right)$ as it has a fixed value. We prove that this can be solved at multiple grid sizes simultaneously by dynamic programming. Our derivation in the following in fact extends the dynamic programming proof of MIC [RRF+11]. That is, the latter proved for the pairwise case, and we generalize to the multivariate case.

First, w.l.o.g., let $X(1) \leq \ldots \leq X(N)$ be realizations of X. Further, let

$$X(j, m) = \{X(j), X(j + 1), \ldots, X(m)\}$$

where $j \leq m$. As before, we write $X(1, N)$ as X. We use $H(C_k \mid \langle X(j, m)\rangle)$ to denote $H(C_k)$ computed using the $(m - j + 1)$ points of **DB** corresponding to $X(j)$ to $X(m)$, projected onto the dimensions of C_k. Note that the bins of each dimension in C_k are intact. To show that the optimal discretization of X maximizing $H(X) - H(X, C_k)$ can be searched by dynamic programming, we introduce the

following formulation which will subsequently lead to the solution of our problem. In particular, for $1 \leq l \leq m \leq N$, we write

$$F(m,l) = \max_{g:|g|=l} H(X^g(1,m)) - H(X^g(1,m), C_k)$$

where g is a discretization of $X(1,m)$ in to l bins, and $X^g(1,m)$ is the discretized version of $X(1,m)$ by g. That is, $F(m,l)$ is the maximum value of $H(X(1,m)) - H(X(1,m), C_k)$ over all discretizations g of $X(1,m)$ into l bins. For $1 < l \leq m \leq N$, we derive the following recursive formulation of $F(m,l)$ which gives way to efficiently computing it using dynamic programming, and hence, to efficiently solving our problem of maximizing $H(X) - H(X, C_k)$.

Theorem 20. *We have:*

$$F(m,l) = \max_{j \in [l-1,m)} \frac{j}{m} F(j, l-1) - \frac{m-j}{m} H(C_k \mid \langle X(j+1,m) \rangle) \quad .$$

Proof. Let $g^* = \arg\max_{g:|g|=l} H(X^g(1,m)) - H(X^g(1,m), C_k)$. We denote l bins that g^* generates on X as $b(X)_1, \ldots, b(X)_l$. We write $|b(X)_t|$ as the number of values of X in $b(X)_t$. For each $X'_i \in C_k$, we denote its bins as $b(X'_i)_1, \ldots, b(X'_i)_{n'_i}$.

Let $c_z = \sum_{t=1}^{z} |b(X)_t|$. Note that each bin of X is non-empty, i.e., $c_z \geq z$. We use $H(C_k \mid b_t)$ to denote $H(C_k)$ computed using the points of $\mathbf{DB}$ corresponding to the realizations of X in b_t, projected onto C_k.

We write $(t, t_1, \ldots, t_k)$ as the number of points in the cell made up by bins $b(X)_t$, $b(X'_1)_{t_1}, \ldots, b(X'_k)_{t_k}$. We use $(t, *, \ldots, *)$ to also denote $b(X)_t$. We note that $|(t, *, \ldots, *)| = \sum_{t_1=1}^{n'_1} \ldots \sum_{t_k=1}^{n'_k} |(t, t_1, \ldots, t_k)|$. We have: $F(m,l)$

$$
\begin{aligned}
&= \sum_{t=1}^{l} \frac{|(t,*,\ldots,*)|}{m} \log \frac{m}{|(t,*,\ldots,*)|} \\
&- \sum_{t=1}^{l} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(t,t_1,\ldots,t_k)|}{m} \log \frac{m}{|(t,t_1,\ldots,t_k)|} \\
&= \sum_{t=1}^{l} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(t,t_1,\ldots,t_k)|}{m} \log \frac{|(t,t_1,\ldots,t_k)|}{|(t,*,\ldots,*)|} \\
&= \sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(t,t_1,\ldots,t_k)|}{m} \log \frac{|(t,t_1,\ldots,t_k)|}{|(t,*,\ldots,*)|} \\
&\quad + \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(l,t_1,\ldots,t_k)|}{m} \log \frac{|(l,t_1,\ldots,t_k)|}{|(l,*,\ldots,*)|} \\
&= \frac{c_{l-1}}{m} \times \sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(t,t_1,\ldots,t_k)|}{c_{l-1}} \log \frac{|(t,t_1,\ldots,t_k)|}{|(t,*,\ldots,*)|} \\
&\quad + \frac{|(l,*,\ldots,*)|}{m} \times \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(l,t_1,\ldots,t_k)|}{|(l,*,\ldots,*)|} \log \frac{|(l,t_1,\ldots,t_k)|}{|(l,*,\ldots,*)|} \\
&= \frac{c_{l-1}}{m} F(c_{l-1}, l-1) - \frac{m-c_{l-1}}{m} H(C_k \mid b(X)_l) \\
&= \frac{c_{l-1}}{m} F(c_{l-1}, l-1) - \frac{m-c_{l-1}}{m} H(C_k \mid \langle X(c_{l-1}+1, m) \rangle) \quad .
\end{aligned}
$$

In the last line,

$$
\sum_{t=1}^{l-1} \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} \frac{|(t,t_1,\ldots,t_k)|}{c_{l-1}} \log \frac{|(t,t_1,\ldots,t_k)|}{|(t,*,\ldots,*)|}
$$

is equal to $F(c_{l-1}, l-1)$ because otherwise, we could increase $F(m,l)$ by choosing a different discretization of $X(1, c_{l-1})$ into $l-1$ bins. This in turn contradicts our definition of $F(m,l)$. Since $c_{l-1} \in [l-1, m)$ and $F(m,l)$ is maximal over all $j \in [l-1, m)$, we arrive at the final result. □

We design a dynamic programming search following Theorem 20, and identify the best discretizations of X at different grid sizes $n < \frac{N^{(1-\epsilon)}}{\max(\{n'_i\}_{i=1}^{k})}$. Then, we use Equation (6.3) to identify the optimal discretization of X.

We present the pseudo-code of our method in Algorithm 3. Similarly to CMI++, if we use the original set of cut points per dimension, the time complexity of using dynamic programming for each dimension is $O(N^3)$, which would be restrictive

Algorithm 3: COMPUTING MAC

$R = \{X_1, \ldots, X_D\}$
$C = \emptyset$
Pick X'_1 and X'_2 according to Section 6.4.1
$R = R \setminus \{X'_1, X'_2\}$
$C = C \cup \{X'_1, X'_2\}$
for $k = 2 \rightarrow D - 1$ **do**
    Pick $X'_{k+1} \in R$ according to Section 6.4.2
    $R = R \setminus \{X'_{k+1}\}$
    $C = C \cup \{X'_{k+1}\}$
    Discretize X'_{k+1} according to Section 6.4.3
end for
Compute MAC using the grid obtained

for large data. To address this, we also impose a maximum grid size max_grid and limit its number of cut points per dimension to $c \times max\_grid$ with $c > 1$. Again, we achieve this by equal-frequency binning on the dimension with the number of bins equal to $(c \times max\_grid + 1)$. In this chapter, we set $c = 2$ according to our preliminary analysis.

The time complexity of MAC includes (a) the cost of pre-sorting the values of all dimensions $O(DN \log N)$, (b) the cost of finding and discretizing X'_1 and X'_2 $O(D^2 N^{3(1-\epsilon)})$, and (c) the cost of finding and discretizing subsequent dimensions $O(D^2 N + DN^{3(1-\epsilon)})$. The overall complexity is $O(D^2 N^{3(1-\epsilon)})$. As we fix ϵ to 0.5 in our implementation, the complexity of MAC is $O(D^2 N^{1.5})$.

6.5. Experiments

For assessing the performance of MAC in detecting pairwise correlations, we compare against MIC [RRF+11] and DCOR [SR09], two state-of-the-art correlation measures. However, neither MIC nor DCOR are directly applicable in the multivariate setting. In order to make a meaningful comparison, we consider two approaches for extending these methods: (a) taking the sum of pairwise correlation scores and normalizing it by the total number of pairs, and (b) taking the maximum of these scores. Empirically, we found the second option to yield best performance, and hence, we use this as the multivariate extension for both MIC and DCOR. Note that we have also compared MAC and CMI++ and the results show that they perform relatively similar. So we skip CMI++ to better focus on demonstrating the benefits of MAC.

6.5.1. Synthetic data

To evaluate how MAC performs in different settings, we first use synthetic data. We aim to show MAC can successfully detect both pairwise and multivariate correlations.

Assessing functional correlations. As a first experiment, we investigate whether MAC can detect linear and non-linear functional correlations. To this end, we create data sets simulating four different functions.

As performance metric we use the power of the measures, as in [RRF+11]: For each function, the null hypothesis is that the data dimensions are statistically independent. For each correlation measure, we determine the cutoff for testing the independence hypothesis by (a) generating 100 data sets of a fixed size, (b) computing the correlation score of each data set, and (c) setting the cutoff according to the significance level $\alpha = 0.05$. We then generate 100 data sets *with* correlations, adding Gaussian noise. The power of the measure is the proportion of the new 100 data sets whose correlation scores exceed the cutoff.

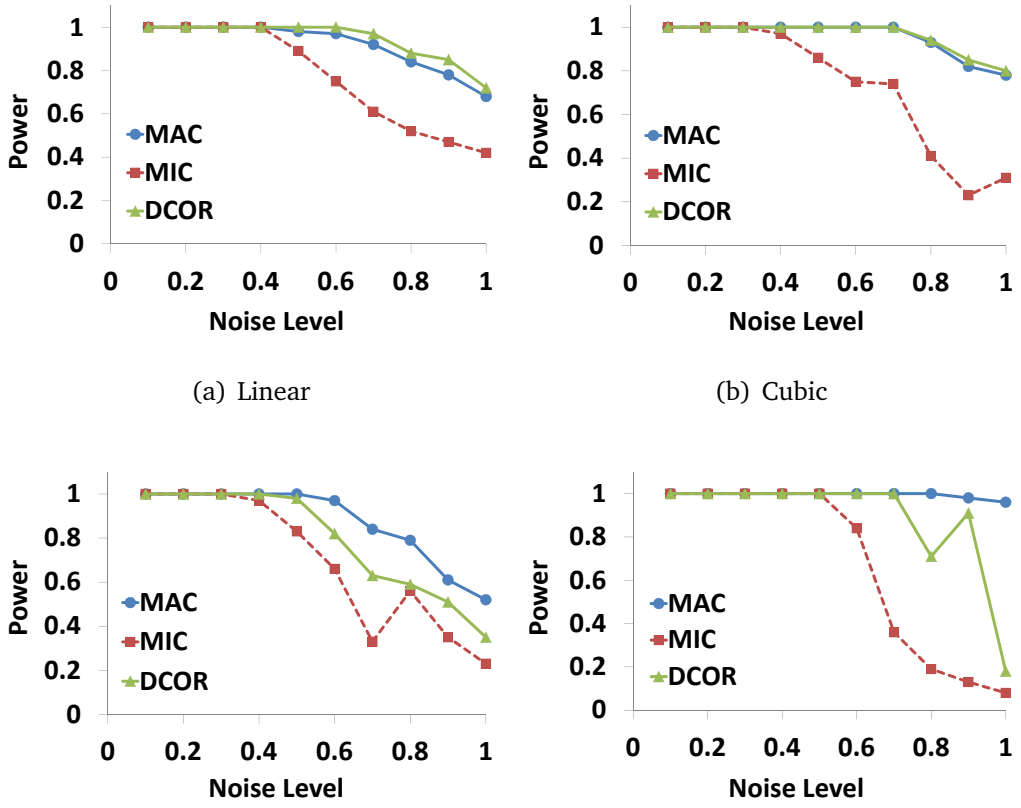
Results on pairwise functional correlations. We create data sets of 1000 data points, using respectively a linear, cubic, sine, and circle as generating functions. Recall that for pairwise cases, we search for the optimal discretization of one dimension at a time (Section 6.4.1). We claim this leads to better quality than MIC, which heuristically fixes a discretization on the remaining dimension.

We report the results in Figure 6.1. Overall, we find that MAC outperforms MIC on all four functions. Further, we see that MAC and DCOR have about the same power in detecting linear and cubic correlations. For the more complex correlations, the performance of DCOR starts to drop. This suggests MAC is better suited than DCOR for measuring and detecting strongly non-linear correlations.

Results on multivariate functional correlations. Next we consider multivariate correlations. To this end we again create data sets with 1000 data points, but of differing dimensionality. Among the functions is a multi-dimensional spiral. Figure 6.2 shows an example 3-d spiral function.

We show the results for 4-variate, 32-variate, and 128-variate functions in Figures 6.3, 6.4, and 6.5, respectively. We see that MAC outperforms both MIC and DCOR in all cases. We also see that MAC is well suited for detecting multivariate correlations.

We also use synthetic data sets with multivariate functional correlations to study the parameterization of MAC w.r.t. to ϵ and c. In particular, to examine the sensitivity of MAC to ϵ, we fix $c = 2$. To examine the sensitivity of MAC to c, we fix $\epsilon = 0.5$. For illustration purposes, we show the results for 32-variate functions (at noise = 20%) in Figures 6.6 and 6.7. Results on other dimensionality exhibit a similar trend, and hence, are skipped. Going over the two figures, we can see that with $0.4 \leq \epsilon \leq 0.5$, the performance of MAC (in terms of statistical power) is quite stable. However, $\epsilon = 0.5$ takes the least runtime (as the larger ϵ is, the faster MAC is). Thus, we choose to set $\epsilon = 0.5$ in the rest of this chapter. On the other hand, we notice that the performance of MAC is fairly robust to c; in particular, $c = 2$ yields an overall good quality. Thus, we recommend to use $c = 2$ in the remaining experiments. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable parameterization might be required.



(a) Linear

(b) Cubic

(c) Sine

(d) Circle

Figure 6.1.: [Higher is better] Baseline results for 2-dimensional functions, statistical power vs. noise.

Assessing non-functional correlations. Finally, we consider multivariate non-functional correlations. To this end we generate data with density-based subspace clusters as in [MGAS09]. For each dimensionality $r < D$, we select w subspaces $\mathcal{S}_c$ having r dimensions (called correlated subspaces), and embed two density-based clusters representing correlation patterns. Since density-based clusters can have arbitrarily complex shapes and forms, we can simulate non-functional correlations of arbitrary complexity. For each correlated subspace, we create another subspace by substituting one of its dimensions by a randomly sampled noisy dimension. Thus, in total, we have $2w$ subspaces. We compute the correlation score for each of these subspaces, and pick the top-w subspaces $\mathcal{S}_t$ with highest scores. The power of the correlation measure is identified as its precision and recall, i.e., $\frac{|\mathcal{S}_c \cap \mathcal{S}_t|}{w}$ since $|\mathcal{S}_c| = |\mathcal{S}_t| = w$. We add noise as above.

Results on non-functional correlations. We create data sets with 1000 data points, of varying dimensionality. For each value of r and w, we repeat the above process 10 times and consider the average results, noting that the standard deviations are very small. As a representative, we present the results with $w = 14$ in

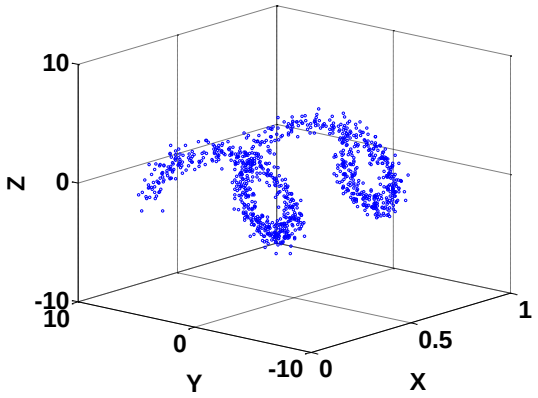
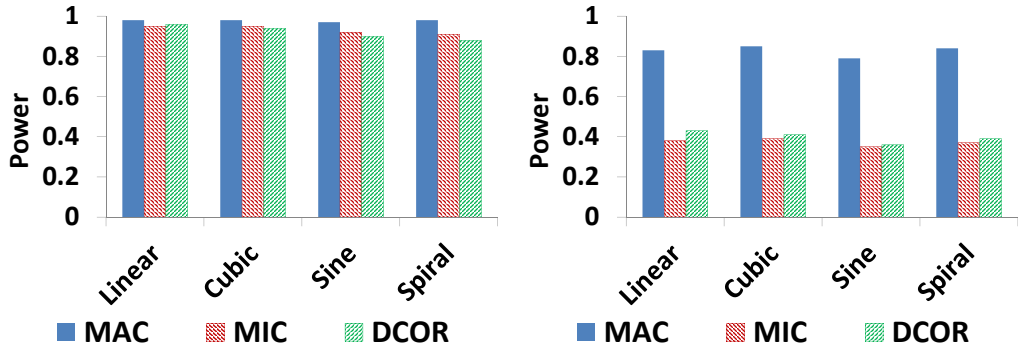


Figure 6.2.: 3-d spiral function.



(a) Noise = 20%

(b) Noise = 80%

Figure 6.3.: [Higher is better] Statistical power vs. noise for 4-dimensional functions.

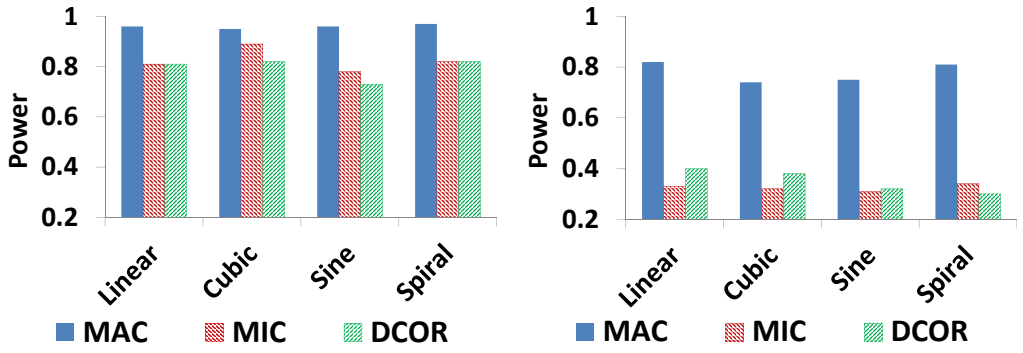
Figure 6.8. We see that compared to both MIC and DCOR, MAC identifies these correlations best. Notably, its performance is consistent across different dimensionalities. In addition, MAC is robust against noise.

Scalability. Finally, we examine scalability of measures with regard to dimensionality and data size. For the former, we generate data sets with 1000 data points and dimensionality varied. For the latter, we generate data sets with dimensionality 4 and data size varied. We show the results in Figure 6.9. Each result is the average of 10 runs. Overall, in both dimensionality and data size, we find that MAC scales much better than MIC and is close to DCOR.

The experiments so far show that MAC is a very efficient and highly accurate multivariate correlation measure.

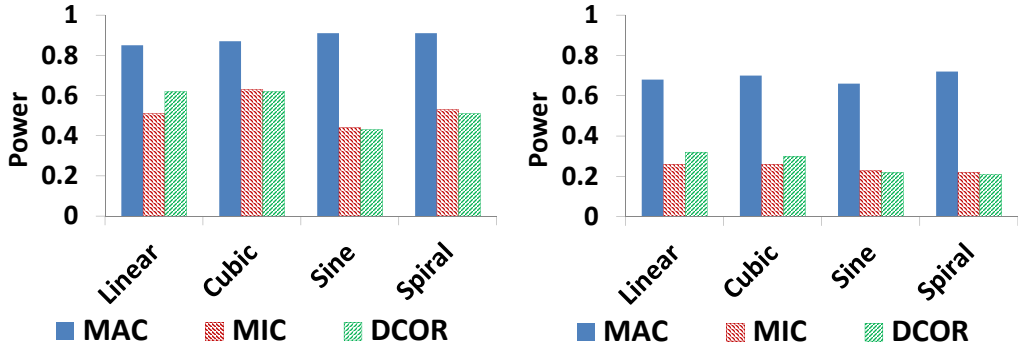
6.5.2. Real-world data

Next, we consider real-world data. We apply MAC in two typical applications of correlations measures in data analysis: cluster analysis and data exploration.



(a) Noise = 20% (b) Noise = 80%

Figure 6.4.: [Higher is better] Statistical power vs. noise for 32-dimensional functions.

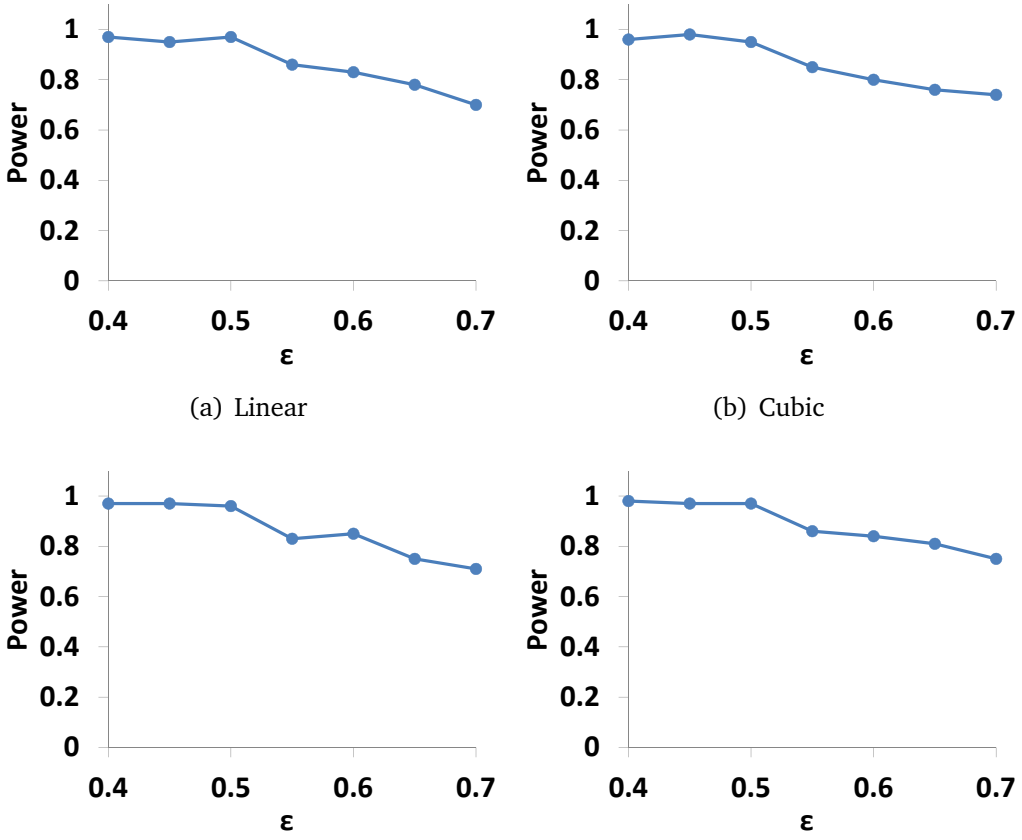


(a) Noise = 20% (b) Noise = 80%

Figure 6.5.: [Higher is better] Statistical power vs. noise for 128-dimensional functions.

Cluster analysis. For cluster analysis, it has been shown that mining subspace clusters is particularly useful when the subspaces show high correlation, i.e., include few or no irrelevant dimensions [MGAS09]. Thus, in this experiment, we plug MAC and MIC into the Apriori subspace search framework to assess their performance. Here, we omit DCOR as we saw above that MIC and DCOR perform similarly on multivariate data. Instead, we consider ENCLUS [CFZ99] as a baseline. Note that ENCLUS computes total correlation using equal-width discretization rather than correlation-aware discretization. We show an enhanced performance by using MAC instead.

Our setup follows that in Chapter 3: We use each measure for subspace search, and apply DBSCAN [EKSX96] to the top 100 subspaces with highest correlation scores. Using these we calculate Accuracy and F1 scores. However, different from Chapter 3, here for each method tested, we try different parameter settings and pick the best result. Regarding the data, we use 7 labeled data sets from different



(a) Linear (b) Cubic

(c) Sine (d) Circle

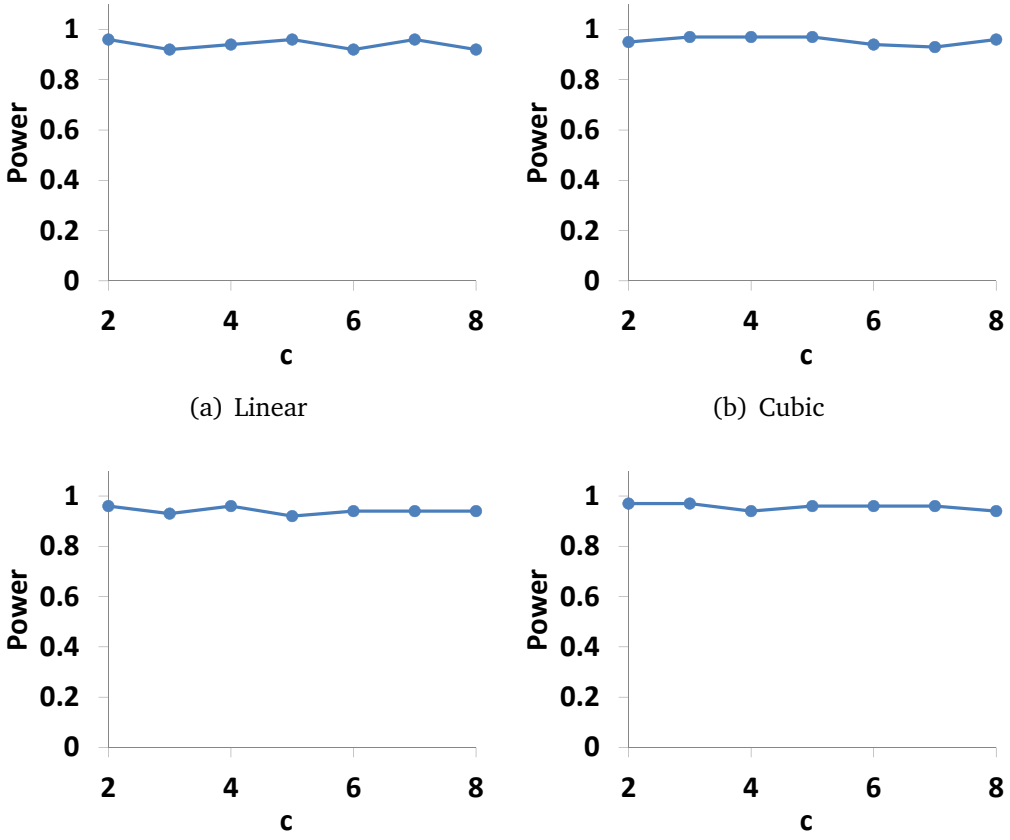
Figure 6.6.: [Higher is better] Sensitivity to ϵ.

domains ($N \times D$): Musk (6598×166), Letter Recognition (20000×16), PenDigits (7494×16), Waveform (5000×40), WBCD (569×30), Diabetes (768×8), and Glass (214×9), taken from the UCI ML repository. For each data set, we regard the class labels as the ground truth clusters.

The results are in Table 6.1. Overall, MAC achieves the highest clustering quality. It consistently outperforms MIC and ENCLUS. Notably, it discovers higher dimensional subspaces. Recall that Apriori imposes the requirement that each subspace is only considered if all of its child subspaces show high correlation. Whereas MAC correctly identifies correlations in these lower-order projections, the other methods assign inaccurate correlation scores more often, which prevents them from finding the larger correlated subspaces. As a result, MAC detects correlations in multivariate real-world data sets better than its competitors.

By applying a Friedman test [Dem06] at significance level $\alpha = 0.05$, we find that the observed differences in Accuracy and F1 are significant. By performing a post-hoc Nemenyi test we learn that MAC significantly outperforms MIC and ENCLUS.

Discovering novel correlations. To evaluate the efficacy of MAC in data exploration, we also apply MAC with 4S search scheme on the climate data set (cf.,



(a) Linear (b) Cubic

(c) Sine (d) Circle

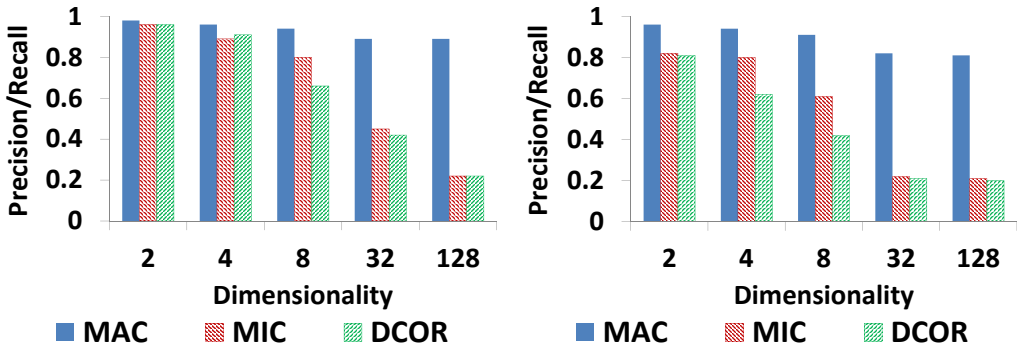
Figure 6.7.: [Higher is better] Sensitivity to c.

Chapter 4). The results also show that MAC is able to discover novel and significant correlations.

6.6. Conclusions

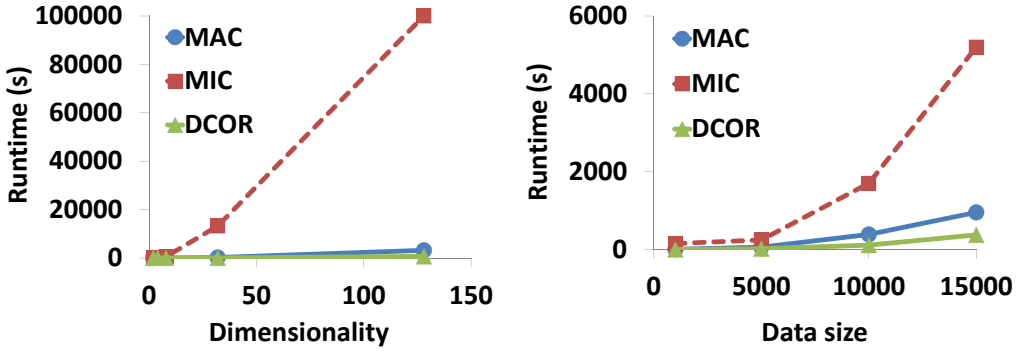
We introduced MAC, a maximal correlation measure for multivariate data as well as another correlation-aware discretization technique. In short, MAC discovers correlation patterns by identifying the discretizations of all dimensions that maximize their normalized total correlation. We proposed an efficient estimation of MAC that also ensures high quality. Experiments showed that MAC successfully discovered interesting complex correlations in real-world data sets.

The research proposed here gives way to computing the total correlation on empirical data, which has wide applications in various fields. In addition, it demonstrates the potential of multivariate maximal correlation analysis to data analytics. Through MAC, we have shown that searching for the optimal transformations of all dimensions concurrently is impractical. Instead, we conjecture that: *To efficiently solve the optimization problem in Definition 17, one needs to find an order of* $\{X_i\}_{i=1}^{D}$ *to process as well*. Solving this conjecture for other general cases is beyond



(a) Noise = 20% (b) Noise = 80%

Figure 6.8.: [Higher is better] Precision/Recall vs. noise for non-functional correlations (i.e., clusters).



(a) Runtime vs. dimensionality (b) Runtime vs. data size

Figure 6.9.: [Lower is better] Scalability of correlation measures with regard to dimensionality and data size.

the scope of this thesis, and reserved for our future work on maximal correlation analysis.

	MAC	**MIC**	**ENCLUS**
Musk (6598×166)			
F1	**0.89**	0.66	0.62
Accuracy	**0.88**	0.70	0.63
Letter (20000×16)			
F1	**0.82**	0.64	0.64
Accuracy	**0.84**	0.69	0.71
PenDigits (7494×16)			
F1	**0.85**	0.48	0.50
Accuracy	**0.88**	0.75	0.66
Waveform (5000×40)			
F1	**0.50**	0.28	0.31
Accuracy	**0.65**	0.37	0.40
WBCD (569×30)			
F1	**0.74**	0.55	0.57
Accuracy	**0.92**	0.80	0.75
Diabetes (768×8)			
F1	**0.72**	0.54	0.25
Accuracy	**0.78**	0.53	0.67
Glass (214×9)			
F1	**0.70**	0.32	0.26
Accuracy	**0.70**	0.47	0.52

Table 6.1.: Clustering results on real-world data sets.

7. Supervised MAC and MAC for Mixed Typed data

In Chapter 6, we propose MAC which searches for the optimal discretization to compute total correlation on real-valued data sets. In this chapter, we introduce two extensions of MAC to broaden its applicability.

In the first extension, we aim at incorporating external information to the discretization process of MAC. An example of external information is class labels. Through this, we show that MAC can be adapted to the supervised setting where expert knowledge is available. Thus, we name this variant of MAC as SUPMAC for supervised MAC. Taking into account class labels, the discretization of SUPMAC is potentially suited for supervised tasks, such as classification. In addition, SUPMAC can be combined with subspace search schemes (e.g., the ones proposed in Chapters 3 and 4) for supervised feature selection. We will shortly discuss the relation between SUPMAC and each type of related work in the following.

In the second extension, we aim at adapting MAC to mixed typed data, i.e., we increase its applicability in real-world applications. We name this variant of MAC as MIXEDMAC. In principle, MIXEDMAC is able to search for the optimal discretization of real-valued dimensions taking into account the available discrete/categorical dimensions, instead of ignoring them. To this end, MIXEDMAC is applicable to heterogeneous data types and can be used as an alternative to DECOREL (see Chapter 5) for analyzing single data sets. Before going into the details of both extensions, we discuss their respective related work.

7.1. Related Work

Here, we discuss related work of both SUPMAC and MIXEDMAC.

Related work to SUPMAC. As aforementioned, the discretization of SUPMAC can be used for supervised tasks, such as classification. To this end, we note that

SUPMAC is different from existing supervised discretization methods [FI93, Ker92, KC04, Bou06] in two aspects:

- Supervised discretization aims at finding the discretization optimizing the correlation between each individual dimension and the class label. In contrast, SUPMAC searches for the discretization optimizing the correlation of all dimensions *given* the class label. That is, the class label serves as a "guideline" for SUPMAC in finding good discretization.
- Since the class label is taken into account during the discretization process, the result of SUPMAC is potentially suited for supervised tasks, such as classification. Further, its discretization can also be used for correlation analysis. Supervised discretization, on the other hand, does not optimize the correlation of all dimensions. Thus, its result may not be relevant for correlation analysis.

The second important property of SUPMAC is that it can be combined with subspace search schemes for supervised feature selection. We note that in Chapter 6, we did combine MAC with subspace search methods for the unsupervised setting, namely clustering analysis and knowledge discovery. The combination of SUPMAC and subspace search in turn potentially yields subspaces that are beneficial supervised tasks. To this end, SUPMAC is related to supervised feature selection [Hal00, PLD05, SSG+07, GN09, RLHEC10, SM11] which has been discussed in Chapter 1. Nevertheless, the connection is not that obvious: Supervised feature selection methods consider correlation among selected dimensions to represent redundancy, which should be minimized. SUPMAC in turn optimizes correlation of all dimensions. However, we note that this correlation is conditioned on the class label. On the other hand, recent work in supervised feature selection [BPZL12, HZW+13, NCRB14] suggests that maximizing correlation of dimensions given the class label can also lead to a relevant set of dimensions. As a result, we find the relevancy of SUPMAC to feature selection. Yet, SUPMAC here provides a more global picture than [BPZL12, HZW+13, NCRB14]: It considers total correlation while the latter considers mutual information only. We will show in our experiments that this brings better performance gain w.r.t. classification.

Related work to MIXEDMAC. DECOREL is capable of performing correlation analysis for mixed data types. Its correlation measure is pairwise. MIXEDMAC on the other hand is a multivariate correlation measure. In addition, the correlation measure of DECOREL uses both cumulative entropy and Shannon entropy. MIXEDMAC in turn is based entirely on Shannon entropy. This could make the scores produced by MIXEDMAC easier to understand. This is because till now, Shannon entropy has been studied much more comprehensively than cumulative entropy. One could also apply total correlation to mixed data types by performing naïve discretization (e.g., equal-width and equal-frequency) on real-valued dimensions. As we have explained many times in this thesis, this practice however is suboptimal. There is also work in the database area for analyzing related columns with mixed data types [ZHO+11]. Its solution tries to separate columns with different

types. MIXEDMAC on the other hand brings them together for a better processing. When combining MIXEDMAC with subspace search schemes, we also see related work on mining clusters and outliers in the setting of heterogeneous data types [OGP06, KG10, MAS09]. However, similarly to methods for subspace clustering and outlier detection, such techniques are closely tied to the specific notion of cluster or outlier used.

7.2. SUPMAC: MAC with External Information

Consider external knowledge expressed in the form of a discrete/categorical multivariate/univariate random variable Z. We further assume that Z has N realizations—each associated with a record of $\mathbf{DB}$. In practice, Z can represent class labels. We consider a setting where we need to condition on Z in our computation, i.e., how the knowledge about Z changes our results. An example of such a setting is supervised feature selection [NCRB14]. W.l.o.g., we assume that Z is univariate and discrete/categorical. The following materials extend straightforwardly to the case when Z is multivariate.

Our goal now is to compute MAC$(\mathbf{DB} \mid Z)$ which is given by:

$$\text{MAC}(\mathbf{DB} \mid Z) = \max_{\substack{G=\{g_1,\ldots,g_D\} \\ \forall i \neq j : n_i \times n_j < N^{1-\epsilon}}} T_n(\mathbf{DB}^G \mid Z)$$

where

$$T_n(\mathbf{DB}^G \mid Z) = \frac{T(\mathbf{DB}^G \mid Z)}{\sum_{i=1}^{D} \log n_i - \max(\{\log n_i\}_{i=1}^{D})} \quad .$$

Intuitively, $T_n(\mathbf{DB}^G \mid Z)$ is the normalized total correlation of $\mathbf{DB}^G$ conditioned on Z. Below we show that each step of the original MAC algorithm can be adapted to incorporate Z. As a result, solving MAC$(\mathbf{DB} \mid Z)$ is analogous to solving MAC$(\mathbf{DB})$. This completes our extension of MAC to create SUPMAC.

7.2.1. Extending Section 6.4.1

We aim to compute MAC$(X, Y \mid Z)$ for every pair of dimensions (X, Y) and pick (X'_1, X'_2) with the largest score. We achieve this by cumulative entropy. We formulate a similar problem: *Given a grid size of Y, find the respective discretization g of Y that minimizes $h(X \mid Z, Y^g)$*. Solving this problem, we essentially find the optimal discretization of Y at the given grid size that intuitively maximizes $H(X \mid Z) - H(X \mid Z, Y)$ without having to discretize X at the same time.

To this end, we can solve our new optimization problem using our calculation of CMI++ (cf., Chapter 3). For readability, we re-present the proof for this particular problem, which is simpler than the one in CMI++. In particular, in the following we prove that our new optimization problem can be solved at multiple grid sizes simultaneously by dynamic programming.

First, we use $h(X \mid Z, \langle Y(j,m)\rangle)$ to denote $h(X \mid Z)$ computed using the $(m-j+1)$ points of **DB** corresponding to $Y(j)$ to $Y(m)$, projected onto X and Z. To show that the optimal discretization of X minimizing $h(X \mid Z,Y)$ can be searched by dynamic programming, we introduce the following formulation which will subsequently lead to the solution of our problem. In particular, for $1 \leq l \leq m \leq N$, we write

$$f(m,l) = \min_{g:|g|=l} h(X \mid Z, Y^g(1,m))$$

where g is a discretization of $Y(1,m)$ in to l bins, and $Y^g(1,m)$ is the discretized version of $Y(1,m)$ by g. That is, $f(m,l)$ is the minimum valued of $h(X \mid Z, Y(1,m))$ taken over all discretization g of $Y(1,m)$ into l bins. For $1 < l \leq m \leq N$, we derive the following recursive formulation of $f(m,l)$ which gives way to computing it using dynamic programming, and hence, to solving our problem of minimizing $h(X \mid Z,Y)$.

Theorem 21. *We have:*

$$f(m,l) = \min_{j\in[l-1,m)} \frac{j}{m} f(j,l-1) + \frac{m-j}{m} h(X \mid Z, \langle Y(j+1,m)\rangle) \quad .$$

Proof. Let $g^* = \arg\min_{g:|g|=l} h(X \mid Z, Y^g(1,m))$. We denote l bins that g^* generates on Y as $b_1, \ldots, b_l$. We write $|b_t|$ as the number of values of Y in b_t. We denote the bins of Z as $b(Z)_1, \ldots, b(Z)_{n_\alpha}$.

Further, let $c_\beta = \sum_{t=1}^{\beta} |b_t|$. Note that each bin of Y is non-empty, i.e., $c_\beta \geq \beta$. We use $h(X \mid Z, b_t)$ to denote $h(X \mid Z)$ computed using the points of **DB** corresponding to the realizations of Y in b_t, projected onto X and Z.

We write $|(t,t_\alpha)|$ as the number of points in the cell made up by bins $b_t, b(Z)_{t_\alpha}$.

We have: $f(m,l)$

$$\begin{aligned}
&= \sum_{t=1}^{l}\sum_{t_\alpha=1}^{n_\alpha} \frac{|(t,t_\alpha)|}{m} h(X \mid b(Z)_{t_\alpha}, b_t) \\
&= \sum_{t=1}^{l-1}\sum_{t_\alpha=1}^{n_\alpha} \frac{|(t,t_\alpha)|}{m} h(X \mid b(Z)_{t_\alpha}, b_t) + \frac{|b_l|}{m}\sum_{t_\alpha=1}^{n_\alpha} \frac{|(l,t_\alpha)|}{|b_l|} h(X \mid b(Z)_{t_\alpha}, b_l) \\
&= \sum_{t=1}^{l-1}\sum_{t_\alpha=1}^{n_\alpha} \frac{|(t,t_\alpha)|}{m} h(X \mid b(Z)_{t_\alpha}, b_t) + \frac{|b_l|}{m} h(X \mid Z, b_l) \\
&= \frac{c_{l-1}}{m}\sum_{t=1}^{l-1}\sum_{t_\alpha=1}^{n_\alpha} \frac{|(t,t_\alpha)|}{c_{l-1}} h(X \mid b(Z)_{t_\alpha}, b_t) + \frac{m-c_{l-1}}{m} h(X \mid Z, \langle Y(c_{l-1}+1,m)\rangle) \\
&= \frac{c_{l-1}}{m} f(c_{l-1}, l-1) + \frac{m-c_{l-1}}{m} h(X \mid Z, \langle Y(c_{l-1}+1,m)\rangle) \quad .
\end{aligned}$$

Note that $\sum_{t=1}^{l-1}\sum_{t_\alpha=1}^{n_\alpha}\frac{|(t,t_\alpha)|}{c_{l-1}}h(X \mid b(Z)_{t_\alpha}, b_t)$ is equal to $f(c_{l-1}, l-1)$ as otherwise, we could decrease $f(m,l)$ by choosing a different discretization of $Y(1, c_{l-1})$ into $l-1$ bins. This in turn contradicts our definition of $f(m,l)$. Since $c_{l-1} \in [l-1, m)$ and $f(m,l)$ is minimal over all $j \in [l-1, m)$, we arrive at the final result. □

7.2.2. Extending Section 6.4.2

To compute MAC$(X, C_k \mid Z)$, we need to maximize

$$\frac{\left(\sum_{i=1}^{k} H(X'_i \mid Z)\right) + H(X \mid Z) - H(X \mid Z, C_k) - H(C_k \mid Z)}{\log n + \sum_{i=1}^{k} \log n'_i - \max(\{\log n\} \cup \{\log n'_i\}_{i=1}^{k})} \tag{7.1}$$

where $n < \frac{N^{(1-\epsilon)}}{\max(\{n'_i\}_{i=1}^{k})}$ and X is discretized into n bins. To efficiently identify X'_{k+1} without having to discretize all dimensions left, we need an objective function that is free of the bin size of each candidate dimension X. Thus, we consider the following term

$$\frac{\left(\sum_{i=1}^{k} H(X'_i \mid Z)\right) + h(X \mid Z) - h(X \mid Z, C_k) - H(C_k \mid Z)}{h(X \mid Z) + \sum_{i=1}^{k} \log n'_i - \max(\{\log n'_i\}_{i=1}^{k})} \quad . \tag{7.2}$$

Informally speaking, we can regard both Equation (7.1) and (7.2) to represent the normalized mutual correlation of X and all dimensions in C_k given Z. They have very similar properties. First, their values are in $[0, 1]$. Second, they are both equal to 0 iff (discretized) X and all dimensions in C_k are statistically independent given Z (conditioning reduces both Shannon and cumulative entropy). Third, they are both maximal when there exists $X'_i \in C_k$ such that (discretized) X and all dimensions in $C_k \setminus \{X'_i\}$, each is a function of X'_i. The detailed explanation of all three properties is as before.

Therefore, instead of solving Equation (7.1) for every n and every X to obtain X'_{k+1}, we propose to use Equation (7.2) as a surrogate indicator of how likely a dimension X is indeed X'_{k+1}—the larger the indicator, the better. Note that Equation (7.2) can still be computed efficiently as Z is discrete/categorical.

7.2.3. Extending Section 6.4.3

To find the optimal discretization of X, for each grid size n, we find the respective discretization of X that maximizes $H(X \mid Z) - H(X, C_k \mid Z)$, which is equivalent to maximizing $H(X, Z) - H(X, Z, C_k)$; we ignore $\left(\sum_{i=1}^{k} H(X'_i \mid Z)\right)$ as it has a fixed

value. We prove that this can be solved at multiple grid sizes simultaneously by dynamic programming.

We use $H(C_k \mid \langle X(j,m)\rangle, Z)$ to denote $H(C_k \mid Z)$ computed using the $(m-j+1)$ points of $\mathbf{DB}$ corresponding to $X(j)$ to $X(m)$, projected onto the dimensions of C_k and Z. Note that the bins of each dimension in C_k are intact. Similarly, the bins of Z are intact. To show that the optimal discretization of X maximizing $H(X,Z) - H(X,Z,C_k)$ can be searched by dynamic programming, we introduce the following formulation which will subsequently lead to the solution of our problem. In particular, for $1 \leq l \leq m \leq N$, we write

$$F(m,l) = \max_{g:|g|=l} H(X^g(1,m),Z) - H(X^g(1,m),Z,C_k)$$

where g is a discretization of $X(1,m)$ in to l bins, and $X^g(1,m)$ is the discretized version of $X(1,m)$ by g. That is, $F(m,l)$ is the maximum value of $H(X(1,m),Z) - H(X(1,m),Z,C_k)$ taken over all discretization g of $X(1,m)$ into l bins. For $1 < l \leq m \leq N$, we derive the following recursive formulation of $F(m,l)$ which gives way to efficiently computing it using dynamic programming, and hence, to efficiently solving our problem of maximizing $H(X,Z) - H(X,Z,C_k)$.

Theorem 22. *We have:*

$$F(m,l) = \max_{j\in[l-1,m)} \frac{j}{m} F(j,l-1) - \frac{m-j}{m} H(C_k \mid \langle X(j+1,m)\rangle, Z) \quad .$$

Proof. Let $g^* = \arg\max_{g:|g|=l} H(X^g(1,m),Z) - H(X^g(1,m),Z,C_k)$.

We denote l bins that g^* generates on X as $b(X)_1,\ldots,b(X)_l$. We write $|b(X)_t|$ as the number of values of X in $b(X)_t$.

For each $X'_i \in C_k$, we denote its bins as $b(X'_i)_1,\ldots,b(X'_i)_{n'_i}$. For Z, we denote its bins as $b(Z)_1,\ldots,b(Z)_{n_z}$.

Let $c_\beta = \sum_{i=1}^{\beta} |b(X)_t|$. Note that each bin of X is non-empty, i.e., $c_\beta \geq \beta$. We use $H(C_k \mid b_t)$ to denote $H(C_k)$ computed using the points of $\mathbf{DB}$ corresponding to the realizations of X in b_t, projected onto C_k.

We write $(t,t_z,t_1,\ldots,t_k)$ as the number of points in the cell made up by bins $b(X)_t, b(Z)_{t_z}, b(X'_1)_{t_1},\ldots,b(X'_k)_{t_k}$. We use $(t,*,\ldots,*)$ to also denote $b(X)_t$. We note that

$$|(t,*,\ldots,*)| = \sum_{t_1=1}^{n'_1} \cdots \sum_{t_k=1}^{n'_k} |(t,t_1,\ldots,t_k)| \quad .$$

We have: $F(m,l)$

$$
\begin{aligned}
&= \sum_{t=1}^{l}\sum_{t_z=1}^{n_z} \frac{|(t,t_z,*,\ldots,*)|}{m} \log \frac{m}{|(t,t_z,*,\ldots,*)|} \\
&- \sum_{t=1}^{l}\sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(t,t_z,t_1,\ldots,t_k)|}{m} \log \frac{m}{|(t,t_z,t_1,\ldots,t_k)|} \\
&= \sum_{t=1}^{l}\sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(t,t_z,t_1,\ldots,t_k)|}{m} \log \frac{|(t,t_z,t_1,\ldots,t_k)|}{|(t,t_z,*,\ldots,*)|} \\
&= \sum_{t=1}^{l-1}\sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(t,t_z,t_1,\ldots,t_k)|}{m} \log \frac{|(t,t_z,t_1,\ldots,t_k)|}{|(t,t_z,*,\ldots,*)|} \\
&\quad + \sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(l,t_z,t_1,\ldots,t_k)|}{m} \log \frac{|(l,t_z,t_1,\ldots,t_k)|}{|(l,t_z,*,\ldots,*)|} \\
&= \frac{c_{l-1}}{m} \times \sum_{t=1}^{l-1}\sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(t,t_z,t_1,\ldots,t_k)|}{c_{l-1}} \log \frac{|(t,t_z,t_1,\ldots,t_k)|}{|(t,t_z,*,\ldots,*)|} \\
&\quad + \frac{|(l,*,\ldots,*)|}{m} \times \sum_{t_z=1}^{n_z} \frac{|(l,t_z,*,\ldots,*)|}{|(l,*,\ldots,*)|} \times \\
&\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(l,t_z,t_1,\ldots,t_k)|}{|(l,t_z,*,\ldots,*)|} \log \frac{|(l,t_z,t_1,\ldots,t_k)|}{|(l,t_z,*,\ldots,*)|} \\
&= \frac{c_{l-1}}{m} F(c_{l-1}, l-1) - \frac{m-c_{l-1}}{m} H(C_k \mid b(X)_l, Z) \\
&= \frac{c_{l-1}}{m} F(c_{l-1}, l-1) - \frac{m-c_{l-1}}{m} H(C_k \mid \langle X(c_{l-1}+1, m)\rangle, Z) \quad .
\end{aligned}
$$

In the second last line,

$$
\sum_{t=1}^{l-1}\sum_{t_z=1}^{n_z}\sum_{t_1=1}^{n'_1}\cdots\sum_{t_k=1}^{n'_k} \frac{|(t,t_z,t_1,\ldots,t_k)|}{c_{l-1}} \log \frac{|(t,t_z,t_1,\ldots,t_k)|}{|(t,t_z,*,\ldots,*)|}
$$

is equal to $F(c_{l-1}, l-1)$ because otherwise, we could increase $F(m,l)$ by choosing a different discretization of $X(1, c_{l-1})$ into $l-1$ bins. This in turn contradicts our definition of $F(m,l)$. Since $c_{l-1} \in [l-1, m)$ and $F(m,l)$ is maximal over all $j \in [l-1, m)$, we arrive at the final result. □

7.3. MIXEDMAC: MAC with Mixed Typed Data

To extend MAC to create MIXEDMAC, we first assume that among D dimensions, there are D^r real-valued dimensions and D^c discrete/categorical dimensions. W.l.o.g., we write the set of real-valued dimensions as $\mathbf{X} = \{X_1, \ldots, X_{D^r}\}$,

Algorithm 4: COMPUTING MAC

1: $R = \{X_1, \ldots, X_{D^r}\}$
2: $C = \mathbf{Y}$
3: **for** $k = 1 \rightarrow D^r$ **do**
4: Pick $X'_{k+1} \in R$ according to Section 6.4.2
5: $R = R \setminus \{X'_{k+1}\}$
6: $C = C \cup \{X'_{k+1}\}$
7: Discretize X'_{k+1} according to Section 6.4.3
8: **end for**
9: Compute MAC using the grid obtained

and the set of discrete/categorical dimensions as $\mathbf{Y} = \{Y_1, \ldots, Y_{D^c}\}$. Each dimension Y_j has n_j^c bins. Note that if $n_j^c = 1$, Y_j does not contribute to MAC($\mathbf{DB}$) at all. Thus, we can safely assume that $n_j^c \geq 2$. This implies that if there does not exist Y_j with $n_j^c \geq 2$, solving MAC($\mathbf{DB}$) essentially reduces to the case when all dimensions are real-valued. In addition, as a limitation of this thesis, we require that n_j^c is relatively small compared to N. This is to avoid inflated pairwise correlation scores [RRF+11]. More in particular, we assume that $n_{j_1}^c \times n_{j_2}^c < N^{1-\epsilon}$.

Consider a grid $G = \{g_1, \ldots, g_{D^r}\}$ that partitions each X_i into $n_i^r = |g_i|$ bins. The total correlation of $\mathbf{DB}$ given G is:

$$T(\mathbf{DB}^G) = \sum_{i=1}^{D^r} H(X_i^{g_i}) + \sum_{j=1}^{D^c} H(Y_j) - H(X_1^{g_1}, \ldots, X_{D^r}^{g_{D^r}}, Y_1, \ldots, Y_{D^c}) \quad .$$

Let $n^c = \max(\{\log n_i^c\}_{i=1}^{D^c})$. As in Chapter 6, for an unbiased optimization we use the normalized total correlation, which is given as

$$T_n(\mathbf{DB}^G) = \frac{T(\mathbf{DB}^G)}{\sum_{i=1}^{D^r} \log n_i^r + \sum_{j=1}^{D^c} \log n_i^c - \max(\{\log n_i^r\}_{i=1}^{D^r} \cup \{n^c\})} \quad .$$

MAC($\mathbf{DB}$) is given by:

$$\text{MAC}(\mathbf{DB}) = \max_{\substack{G=\{g_1,\ldots,g_D^r\} \\ \forall i_1 \neq i_2 : n_{i_1}^r \times n_{i_2}^r < N^{1-\epsilon}}} T_n(\mathbf{DB}^G) \quad .$$

As $\mathbf{Y} \neq \emptyset$, to compute MAC($\mathbf{DB}$), we in fact can skip the first step of identifying two initial dimensions X'_1 and X'_2 of $\mathbf{X}$. Instead, we only need to proceed by picking one dimension of $\mathbf{X}$ at a time, i.e., only the details of Sections 6.4.2 and 6.4.3 are applicable. The modified pseudo-code is in Algorithm 4. With these, we have completed our construction of MIXEDMAC.

7.4. Experiments

In this section, we report our empirical study on SUPMAC and MIXEDMAC. For each method, we first experiment it with synthetic data sets to assess its statistical power. Then, we use real-world data sets to better understand its performance in practical scenarios.

7.4.1. Results on SUPMAC

In the following, we study the performance of SUPMAC on both synthetic and real-world data sets.

7.4.1.1. Synthetic data

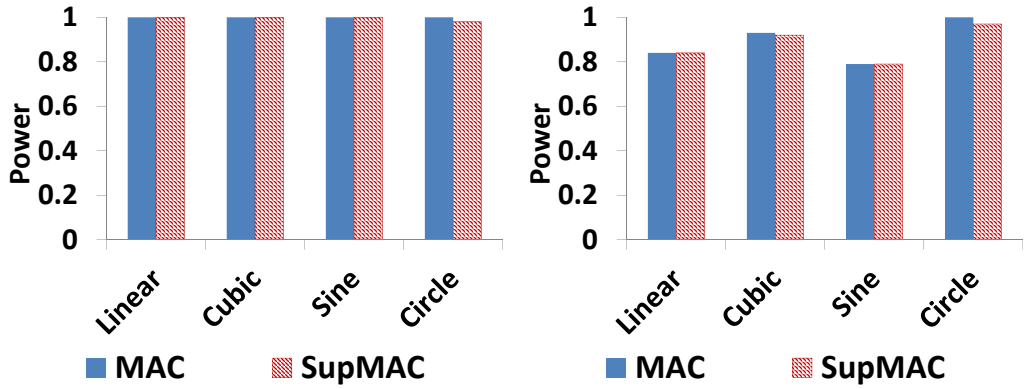
Statistical power. We follow the same procedure in Chapter 6 to test the statistical power of SUPMAC. However, here we need to generate the class label as well. Thus, the data generation process needs to be slightly modified. In the following, we only highlight these modifications. The details that are not mentioned remain intact.

First, we consider functional correlations (i.e., the correlation among dimensions is functional). For each function f, to create a data set with D correlated dimensions, we first generate $(D-1)$ dimensions $X_1, \ldots, X_{D-1}$. Then, we generate the $X_D = f(X_1, \ldots, X_{D-1})$. Next, we generate another dimension $Z = f(X_1, \ldots, X_D)$. Finally, we discretize Z using equal-frequency to obtain the discrete class label. In this way, the data set created contains a correlation among dimensions $X_1, \ldots, X_D$ as well as a correlation between all dimensions and the class label. A data set with no correlation among its dimensions is created by (a) generating a data set with correlation and (b) replacing the data in one dimension with uniformly distributed data.

We now consider non-functional correlations (i.e., the correlation among dimensions is non-functional). To create a data set with D correlated dimensions, we generate a D-dimensional density-based cluster. Next, we generate another dimension $Z = f(X_1, \ldots, X_D)$ where f is a linear function (other functions are also considered but the results show a similar tendency). Finally, we discretize Z using equal-frequency to obtain the discrete class label. In this way, the data set created contains a non-functional correlation among dimensions $X_1, \ldots, X_D$ as well as a functional correlation between all dimensions and the class label. A data set with no correlation among its dimensions is again obtained by replacing a dimension with uniformly distributed data.

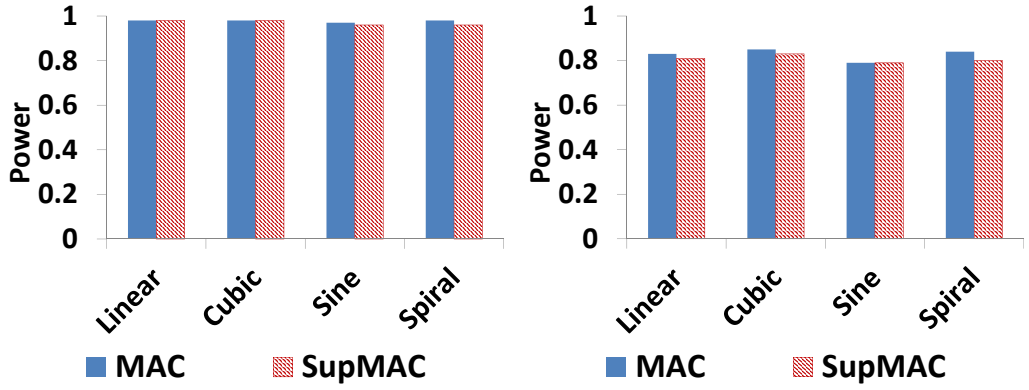
As competitor, in Chapter 6, we have shown that MAC outperforms both MIC [RRF+11] and DCOR [SR09] in terms of statistical power. So we test SUPMAC against MAC only. Note that when testing MAC, we ignore the class label.

The results are in Figures 7.1 to 7.5. We can see that SUPMAC is very close to MAC in terms of statistical power. This shows that the discretization of SUPMAC also successfully optimizes the correlation among dimensions.



(a) Noise = 20% (b) Noise = 80%

Figure 7.1.: [Higher is better] Results on MAC and SUPMAC: Statistical power vs. noise for 2-dimensional functions.

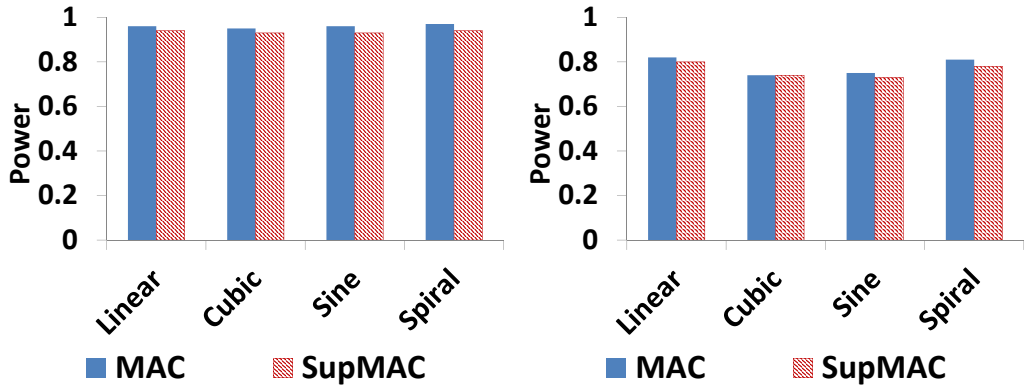


(a) Noise = 20% (b) Noise = 80%

Figure 7.2.: [Higher is better] Results on MAC and SUPMAC: Statistical power vs. noise for 4-dimensional functions.

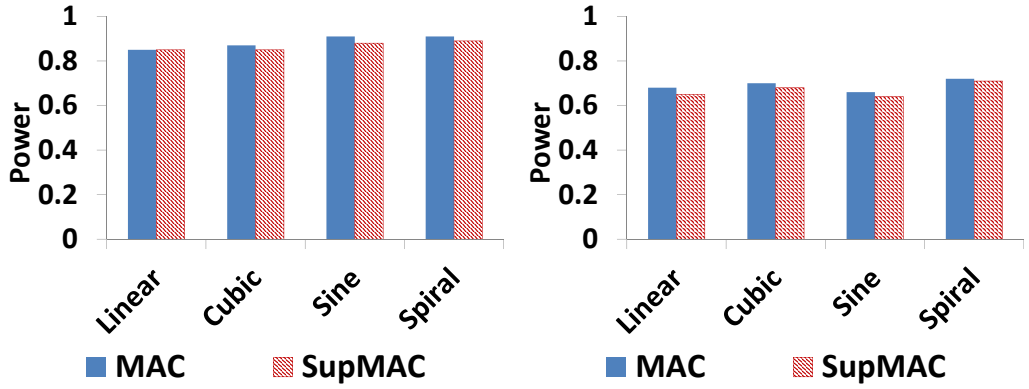
Similarly to MAC, we use synthetic data sets with functional correlations to study the parameterization of SUPMAC. In particular, to examine the sensitivity of SUPMAC to ϵ, we fix $c = 2$. To examine the sensitivity of SUPMAC to c, we fix $\epsilon = 0.5$. For illustration purposes, we show the results for 32-variate functions (at noise = 20%) in Figures 7.6 and 7.7. As in MAC, we make the observation that $\epsilon = 0.5$ and $c = 2$ yield good quality in terms of statistical power. Hence, we will use these values in the remaining experiments of SUPMAC.

Classification accuracy. In the previous experiment, MAC and SUPMAC have rather similar statistical power. It is interesting to see if their discretization results are also suitable for classification. To answer this question, we consider C4.5 as the classifier, keeping its default parameter setting in WEKA. We also again use synthetic data sets. However, we now only use those with correlation (generated as described above). Regarding functional correlations, at each combination *(function, dimensionality, noise level)*, we use multiple data sets and record the average



(a) Noise = 20% (b) Noise = 80%

Figure 7.3.: [Higher is better] Results on MAC and SUPMAC: Statistical power vs. noise for 32-dimensional functions.



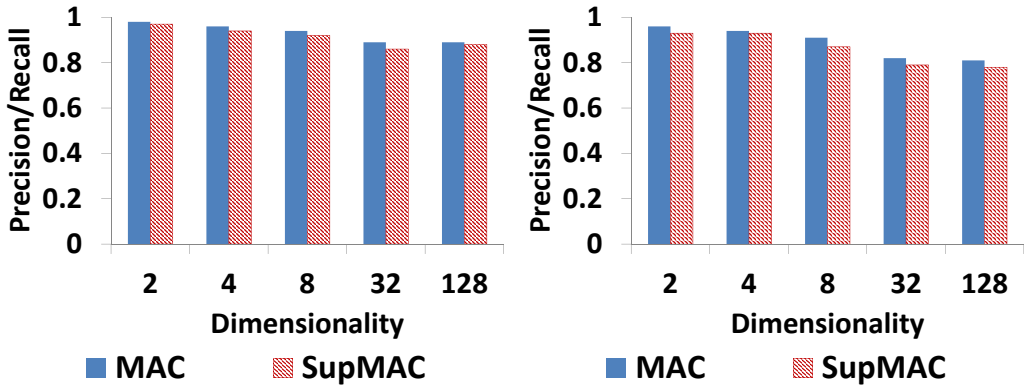
(a) Noise = 20% (b) Noise = 80%

Figure 7.4.: [Higher is better] Results on MAC and SUPMAC: Statistical power vs. noise for 128-dimensional functions.

classification accuracy for each method. Similarly, for non-functional correlations, we record the average accuracy at each combination *(dimensionality, noise level)*. The results are in Figures 7.8 to 7.12. We can see that the difference between SUPMAC and MAC is now clearer with SUPMAC being better. This shows that by incorporating the class label during the optimization process, the discretization of SUPMAC is more suitable for the supervised setting. In both experiments, we observe that SUPMAC takes slightly more time than MAC as it has to consider in addition the class label. However, the difference is negligible.

7.4.1.2. Real-world data

Supervised feature selection. We now combine SUPMAC with the Apriori search scheme to create a method for supervised feature selection. As competitors, we pick InfoGain, GainRatio, and WrapperSE [KJ97] whose implementations are available in WEKA. We also include MRMR [PLD05], QPFS [RLHEC10], and



(a) Noise = 20% (b) Noise = 80%

Figure 7.5.: [Higher is better] Results on MAC and SUPMAC: Precision/Recall vs. noise for non-functional correlations (i.e., clusters).

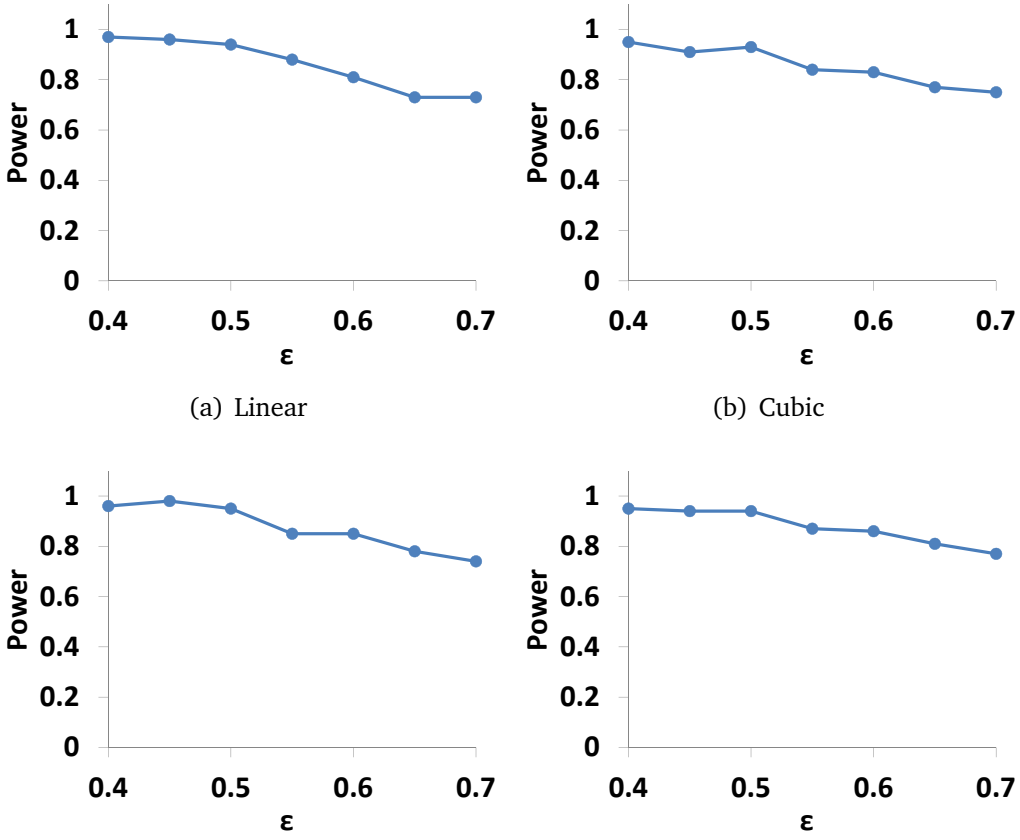
SPEC [NCRB14], three state of the art techniques for supervised feature selection. To test the quality of subspaces/feature subsets detected by each method, we again use C4.5 as the base classifier. Since SUPMAC produces multiple subspaces, we combine the classification results on these subspaces following [Ho98].

Considering the data, we use 12 labeled data sets from the UCI ML repository ($N \times D$): Diabetes (768×8), Glass (214×9), German Credit Data (1000×24), Letter (20000×16), Liver Disorders (345×6), PenDigits (7494×16), Red Wine Quality (1599×11), Segment (2310×16), Waveform (5000×40), WBC (198×33), WBCD (569×30), and White Wine Quality (4898×11).

The classification accuracy is in Table 7.1. Going over the results, we can see that SUPMAC achieves the best overall outcome. In particular, it yields the best classification accuracy in 8 out of 12 data sets. The good performance of SUPMAC could be attributed to the fact that it takes into account the correlation of all dimensions, as well as their association with the class label. In other words, it provides a more global picture than, e.g., MRMR, QPFS, and SPEC, which focus on pairwise correlations. All in all, we conclude that by incorporating class label—a specific type of external information, SUPMAC combined with the Apriori search yields a supervised feature selection technique which is comparable to state of the art methods.

Discovering novel correlations. We here apply SUPMAC with 4S search scheme on the climate data set (cf., Chapter 4). For the class label, we choose a dimension describing the proportion of time within each hour a window is open. We discretize this dimension to create a discrete class label with values 'low', 'medium', and 'high'. As baselines, we again consider MRMR, QPFS, SPEC, InfoGain, GainRatio, and WrapperSE. The classification results are in Table 7.2. We can see that SUPMAC yields the best accuracy. Besides, we also explore the subspaces found by SUPMAC and observe the following interesting correlations:

- occupation of building, outdoor temperature, and indoor temperature



(a) Linear (b) Cubic

(c) Sine (d) Circle

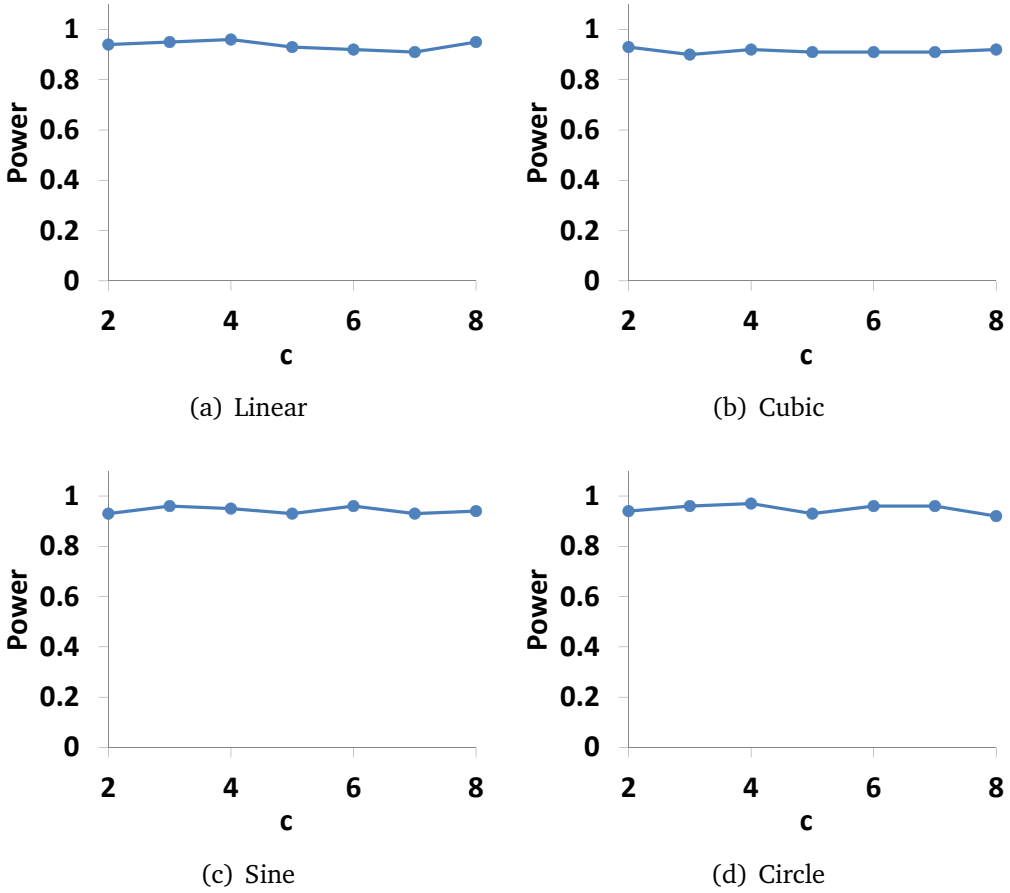
Figure 7.6.: [Higher is better] Sensitivity of SUPMAC to ϵ.

- room CO_2 concentration, indoor humidity, and outdoor humidity
- cooling air temperature, amount of drinking water consumption, and room CO_2 concentration

Those correlations are interesting because the dimensions involved all can impact the chance the window is open. For instance, if there are too many people in the building, it is likely that the window is open to get more fresh air, especially during summer. Likewise, the more CO_2 concentration in the room and the higher indoor humidity, the higher likelihood that the window is open. We note that except for SUPMAC, no other method tested is able discover all of these correlations. This highlights the benefits of SUPMAC for both correlation analysis and feature selection.

7.4.2. Results on MIXEDMAC

To study the performance of MIXEDMAC, we also use both synthetic and real-world data sets.



(a) Linear (b) Cubic

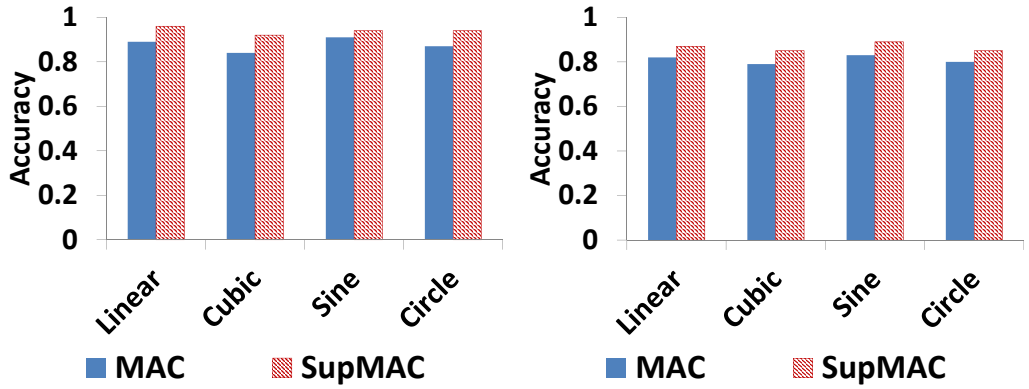
(c) Sine (d) Circle

Figure 7.7.: [Higher is better] Sensitivity of SUPMAC to c.

7.4.2.1. Synthetic data: Statistical power

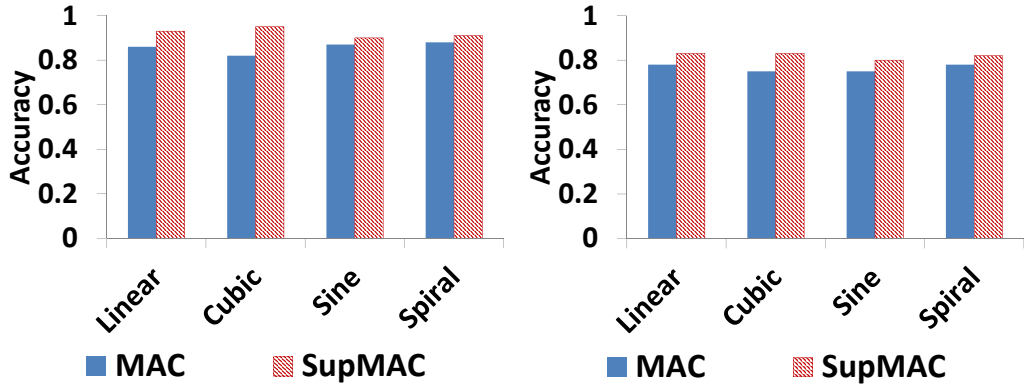
We first generate data sets as in Chapter 6. Then, to create discrete/categorical dimensions, we pick half of the dimensions and discretize them using equal-frequency discretization. As competitors, we consider MIC (adapted to mixed data types as in Chapter 5). DCOR on the other hand requires a suitable distance function for mixed data types, which could be obtained by considering the functions proposed in [BCK08]. Yet, as our goal is to test correlation measures, not distance functions, we prefer a measure not involved in such a choice to avoid unnecessary biases. Thus, we replace DCOR by the measure proposed in DECOREL (cf., Chapter 5). Similarly to MIC, this measure is pairwise. Thus, we adapt it to the multivariate setting using the same procedure we apply for MIC. For readability, we simply call this measure as DECOREL. However, please do not mistaken it as the original DECOREL method, which consists of the correlation measure and a mechanism to generalize from pairwise to multivariate correlations.

The results are in Figures 7.13 to 7.17. We can see that MIXEDMAC consistently outperforms both MAC and DECOREL across different dimensionality and noise levels. This shows that MIXEDMAC better captures multivariate correlations of dimensions with different data types.



(a) Noise = 20% (b) Noise = 80%

Figure 7.8.: [Higher is better] Results on MAC and SUPMAC: Classification accuracy vs. noise for 2-dimensional functions.



(a) Noise = 20% (b) Noise = 80%

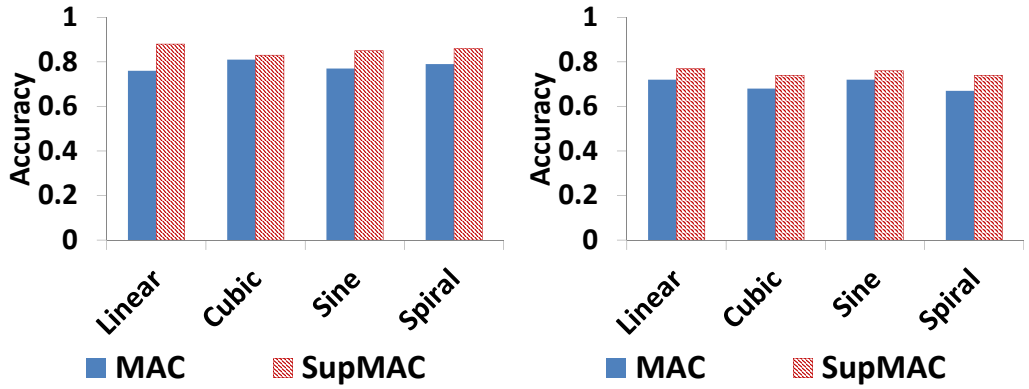
Figure 7.9.: [Higher is better] Results on MAC and SUPMAC: Classification accuracy vs. noise for 4-dimensional functions.

In terms of runtime, we observe that MIC and DECOREL scales rather similarly. The tendency between MIXEDMAC and MIC in turn is analogous to that between MAC and MIC.

To study the parameterization of MIXEDMAC, we again use synthetic data sets with functional correlations. For exposition purposes, we display the results for 32-variate functions (at noise = 20%) in Figures 7.18 and 7.19. We see that $\epsilon = 0.5$ and $c = 2$ yield good performance in terms of statistical power, and hence, we will use these settings in the remaining experiments of MIXEDMAC.

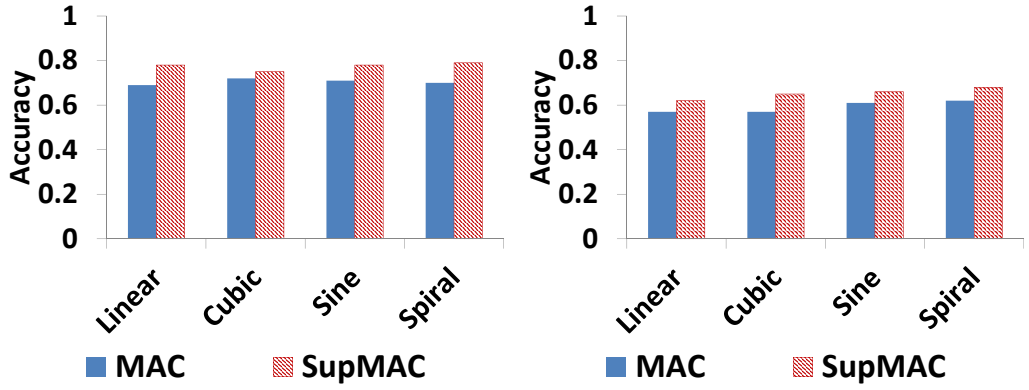
7.4.2.2. Real-world data

Cluster analysis. As in Chapter 6, we plug MIXEDMAC, MIC, and ENCLUS into the Apriori subspace search framework to assess their performance w.r.t. cluster analysis. Note that we exclude CMI++ as it does not handle mixed data types.



(a) Noise = 20% (b) Noise = 80%

Figure 7.10.: [Higher is better] Results on MAC and SUPMAC: Classification accuracy vs. noise for 32-dimensional functions.

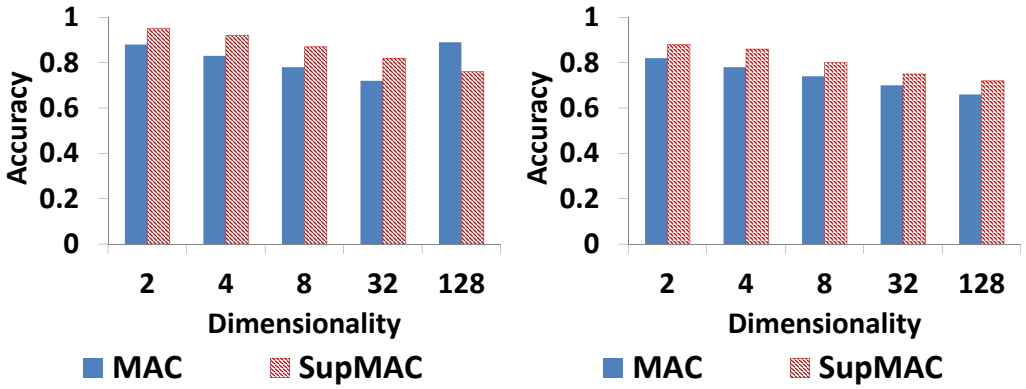


(a) Noise = 20% (b) Noise = 80%

Figure 7.11.: [Higher is better] Results on MAC and SUPMAC: Classification accuracy vs. noise for 128-dimensional functions.

For ENCLUS, we discretize real-valued dimensions using equal-width discretization [CFZ99]. We use each of the 3 measures for subspace search, and apply DBSCAN [EKSX96] to the top subspaces with highest correlation scores. Then, we calculate Accuracy and F1 scores. Since the data sets used contain mixed data types, we apply the GoodAll3 distance function in [BCK08] for categorical dimensions.

As further baselines, we include CLIQUE [AGGR98]—a subspace clustering method using equal-width discretization for real-valued dimensions, HSM [MAS09]—a subspace clustering method designed to handle mixed data types, and FEM [DB04]—an unsupervised feature selection method. FEM selects features by iteratively (a) testing the quality of the current feature set using K-means clustering, and (b) refining the feature set to improve the clustering quality (i.e., minimizing the mean-squared error). As the data sets we consider here have mixed typed dimensions, we combine K-means with the equivalently efficient K-



(a) Noise = 20% (b) Noise = 80%

Figure 7.12.: [Higher is better] Results on MAC and SUPMAC: Classification accuracy vs. noise for non-functional correlations (i.e., clusters).

Data	SUPMAC	MRMR	QPFS	SPEC	InfoGain	GainRatio	WrapperSE
Diabetes	**0.76**	0.75	0.74	0.74	0.74	0.74	0.73
German	**0.75**	0.74	0.73	0.70	**0.75**	0.74	**0.75**
Glass	**0.98**	0.97	0.96	0.73	**0.98**	**0.98**	0.93
Letter	**0.85**	0.81	0.83	0.81	0.68	0.68	0.83
Liver	0.65	0.63	0.60	**0.68**	0.62	0.62	0.64
PenDigits	**0.93**	0.89	0.87	0.89	0.82	0.84	0.86
Red Wine	0.58	0.60	0.59	0.59	0.59	0.61	**0.62**
Segment	0.93	**0.96**	**0.96**	**0.96**	0.88	0.88	0.93
Waveform	**0.80**	0.76	0.65	0.77	0.69	0.68	0.75
WBC	**0.76**	0.75	0.73	**0.76**	0.73	0.73	**0.76**
WBCD	**0.94**	**0.94**	0.92	0.92	0.92	0.92	0.92
White Wine	0.58	0.58	0.57	**0.59**	0.57	0.56	0.58

Table 7.1.: Classification accuracy of SUPMAC, MRMR, QPFS, SPEC, InfoGain, GainRatio, and WrapperSE (using C4.5) on real-world data sets.

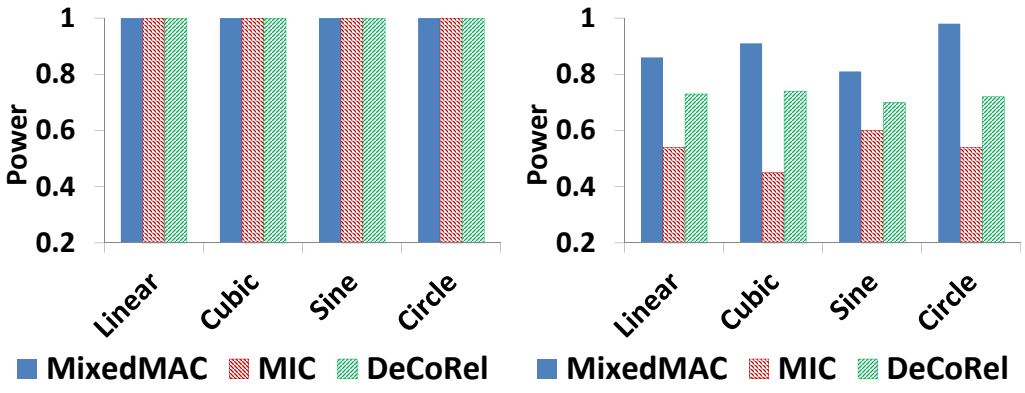
modes [Hua97]. We again plug the GoodAll3 distance function into K-modes. We also apply DBSCAN on the feature set mined by FEM.

The results are in Table 7.3. We see that MIXEDMAC overall outperforms all competitors. This implies that MIXEDMAC better detects correlated subspaces in real-world data sets containing heterogeneous data types. We also note that MIXEDMAC discovers higher dimensional subspaces than other methods tested—a sign signifying that MIXEDMAC better unveils correlations in the data.

Discovering novel correlations. We also apply MIXEDMAC to the subspace search of the original DECOREL method and test on TPC-H, Energy, and Census databases. We compare the groups detected by the modified DECOREL to those of the original DECOREL. We consider two groups to be similar if 80% of their columns are similar. The results show that MIXEDMAC and DECOREL have high similarity

SUPMAC	MRMR	QPFS	SPEC	InfoGain	GainRatio	WrapperSE
0.85	0.75	0.71	0.77	0.64	0.62	0.80

Table 7.2.: Classification accuracy of SUPMAC, MRMR, QPFS, SPEC, InfoGain, GainRatio, and WrapperSE (using C4.5) on climate data set.



(a) Noise = 20%

(b) Noise = 80%

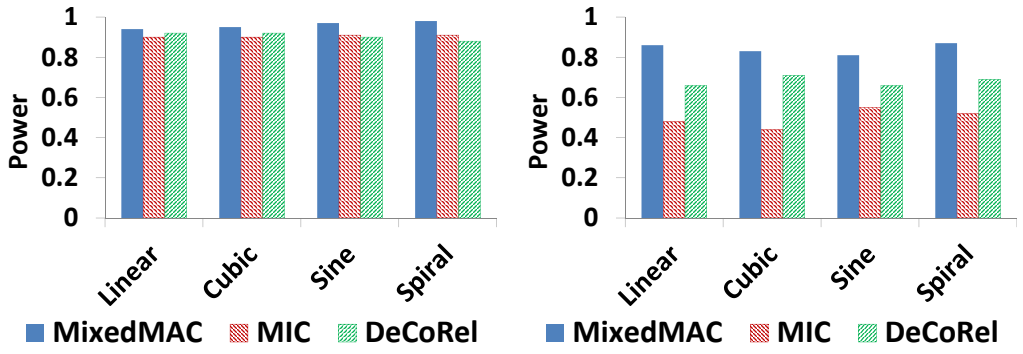
Figure 7.13.: [Higher is better] Results on MIXEDMAC, MIC, and DECOREL: Statistical power vs. noise for 2-dimensional functions.

in their results. This means that MIXEDMAC also is capable of detecting novel correlations in both benchmark and real-world relational databases with mixed data types.

7.5. Conclusions

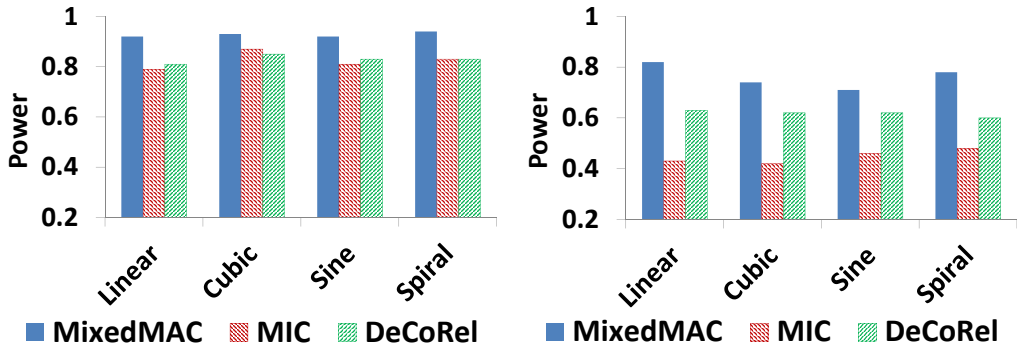
In this chapter, we propose two extensions of MAC: SUPMAC to incorporate external information, and MIXEDMAC to handle mixed typed data. Our experiments show that SUPMAC preserves not only the correlation among the dimensions but also the discriminative power w.r.t. class label. Further, we also demonstrate that MIXEDMAC well detects correlations in real-world data sets containing heterogeneous data types.

This chapter also marks the end of pure correlation analysis. In the next part, two extensions of our research to two other related venues are presented. First, we introduce a discretization scheme which preserves more general interactions in the data. Since this scheme is not bound to any specific correlation measure, it can assist any measure and any task that requires discrete data maintaining dimension interactions. Second, we propose an information-theoretic method for causal discovery. With this development, we go beyond telling if two (groups of) dimensions are correlated: We are able to tell, under the assumption of causal sufficiency (no hidden confounders), which of the two causes the other.



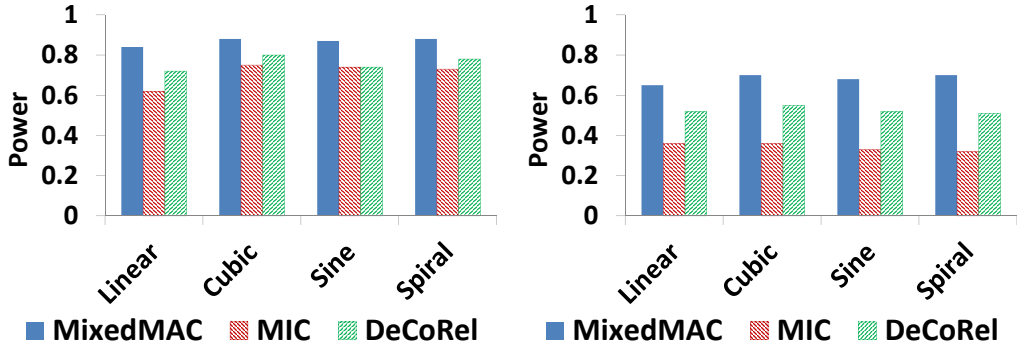
(a) Noise = 20% (b) Noise = 80%

Figure 7.14.: [Higher is better] Results on MIXEDMAC, MIC, and DECOREL: Statistical power vs. noise for 4-dimensional functions.



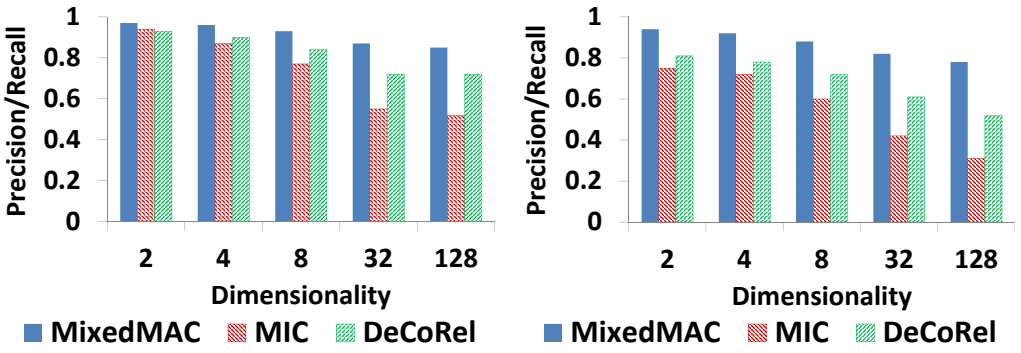
(a) Noise = 20% (b) Noise = 80%

Figure 7.15.: [Higher is better] Results on MIXEDMAC, MIC, and DECOREL: Statistical power vs. noise for 32-dimensional functions.



(a) Noise = 20% (b) Noise = 80%

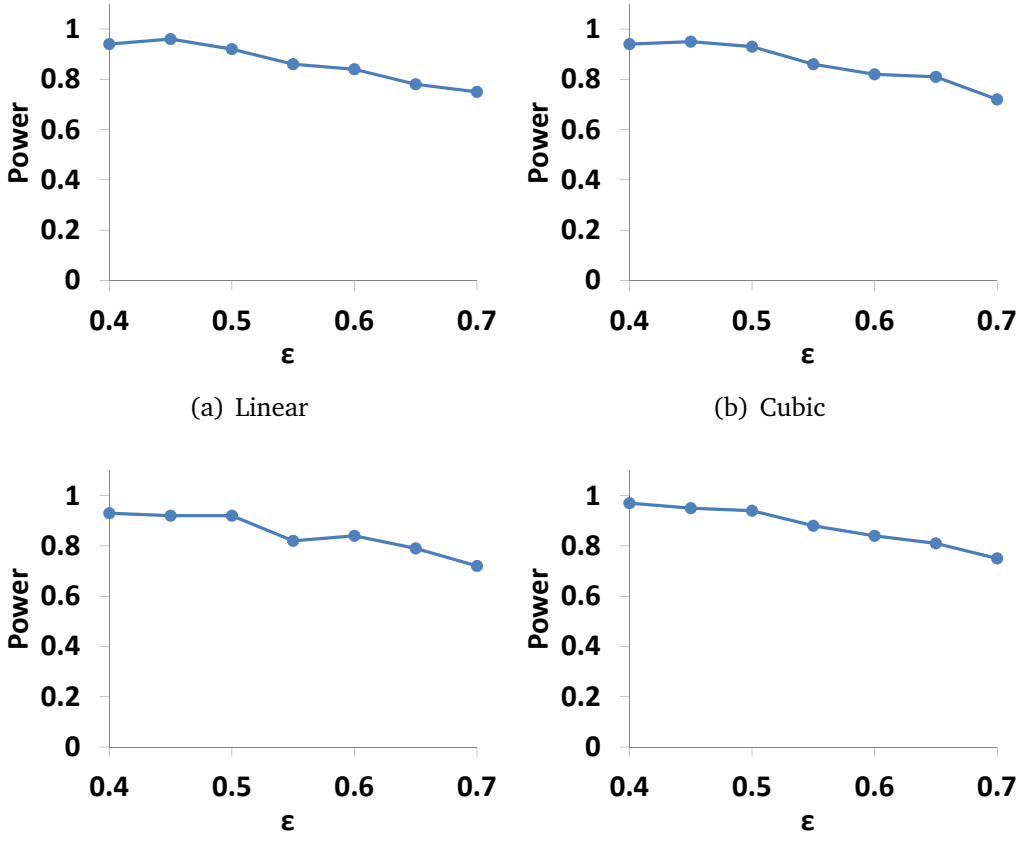
Figure 7.16.: [Higher is better] Results on MIXEDMAC, MIC, and DECOREL: Statistical power vs. noise for 128-dimensional functions.



(a) Noise = 20%

(b) Noise = 80%

Figure 7.17.: [Higher is better] Results on MIXEDMAC, MIC, and DECOREL: Precision/Recall vs. noise for non-functional correlations (i.e., clusters).



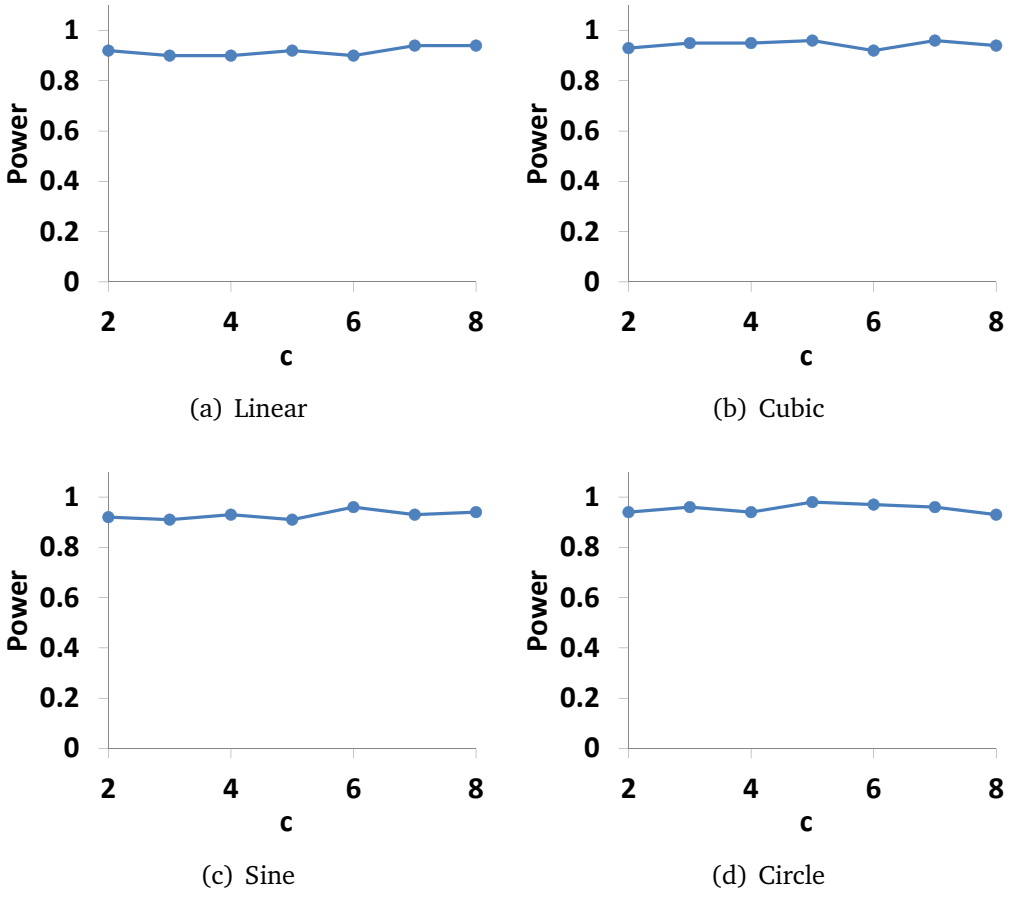
(a) Linear

(b) Cubic

(c) Sine

(d) Circle

Figure 7.18.: [Higher is better] Sensitivity of MIXEDMAC to ϵ.



(a) Linear (b) Cubic

(c) Sine (d) Circle

Figure 7.19.: [Higher is better] Sensitivity of MIXEDMAC to c.

	MIXEDMAC	MIC	ENCLUS	CLIQUE	HSM	FEM
Abalone (4177×8)						
F1	**0.51**	0.46	0.42	0.49	0.49	0.44
Accuracy	**0.55**	0.43	0.50	0.45	0.51	0.45
Annealing (798×38)						
F1	**0.64**	0.57	0.61	0.62	0.60	0.56
Accuracy	**0.62**	0.60	0.61	0.58	0.56	0.58
Arrhythmia (452×279)						
F1	**0.68**	0.63	0.59	0.64	0.66	0.61
Accuracy	**0.69**	0.62	0.62	**0.69**	**0.69**	0.58
Australian Credit Approval (690×14)						
F1	**0.76**	0.73	0.66	0.71	0.73	0.65
Accuracy	**0.78**	0.69	0.65	0.71	0.73	0.68
Indian Liver Patient (579×10)						
F1	0.75	0.66	0.72	**0.76**	0.74	0.71
Accuracy	**0.74**	0.63	0.70	0.72	0.70	0.68

Table 7.3.: Clustering results of MIXEDMAC, MIC, ENCLUS, CLIQUE, HSM, and FEM on real-world data sets.

Part V.

Going beyond Correlation Analysis

8. Interaction-Preserving Discretization of Multivariate Data

This chapter is based on our work originally published as [NMVB14]:

H. V. Nguyen, E. Müller, J. Vreeken, and K. Böhm, *Unsupervised interaction-preserving discretization of multivariate data*, Data Min. Knowl. Discov., vol. 28, no. 5-6, pp. 1366-1397, 2014.

Here, we continue our study on correlation-aware discretization as in Chapters 3, 5, 6 and 7. These chapters essentially focus on finding the optimal discretization w.r.t. a specific correlation measure. In this chapter, we aim at finding discretization preserving interactions of dimensions, or correlations, in general. That is, we do not constrain ourselves to any specific measure. This is beneficial in the sense that the discretization results potentially possess generic properties which in turn are required by some data mining methods, e.g., pattern mining. This new problem setting, however, raises two research questions: What is the characteristic of an interaction-preserving discretization? How to find such discretization? Solving these two questions will lead to a generic correlation-aware discretization method. In fact, there have been attempts to address these questions [Bay01, MPY05]. Our method in this chapter indeed is inspired from such works. However, as we point out later, they have not satisfactorily tackled generic correlation-aware discretization due to (a) the pitfalls of their interaction distances on real-valued data and (b) their lack of an objective function to guide the discretization process. Our technique in turn handles both issues and hence features a promising solution for interaction-preserving discretization. Note that the methods we propose in Chapters 3, 5, 6 and 7 are not applicable here as their solutions are closely tied to their own measure. Therefore, a new technique is required. We stress again that this chapter is about generic discretization techniques that are not tied to any specific correlation measure; yet, their goal is to preserve interactions of dimensions

during the discretization process.

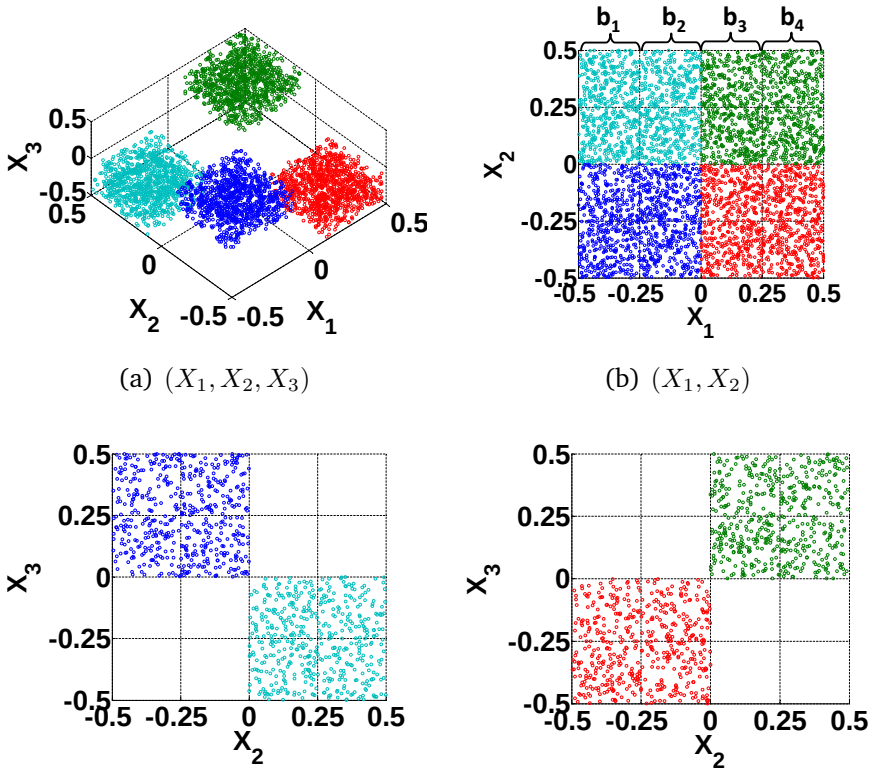
8.1. Introduction

Interaction-preserving discretization has been studied in [Bay01, MPY05]. To understand it, let us first consider unsupervised discretization. In general, this approach aims at transforming continuous data into discrete bins without prior knowledge about any patterns hidden in the data. Well-known (naïve) examples include equal-width and equal-frequency binning, as well as methods that optimize the binning w.r.t. univariate the data distribution [KM07]. Considering only a single dimension, all of these methods fail to preserve *interactions* among multiple dimensions, i.e., they may unwittingly cut a multivariate distribution into many parts and so destroy essential characteristics of that data.

To further illustrate our point, let us now consider an example where it is impossible to find correct cut points by univariate discretization. Consider the simple toy example in Figure 8.1 (this example is based on those of [Bay01, MPY05, KMB12]). It features a 3-dimensional data set with 4 clusters. There are interactions among dimensions X_1, X_2, and X_3. The clusters are only detectable when all dimensions are considered together. For illustration purposes, we assume that X_1 initially has 4 bins: b_1, b_2, b_3, and b_4. To discretize this data set while preserving all clusters, one should discretize X_1 into two bins ($X_1 < 0$ and $X_1 \geq 0$), and similarly for X_2 and X_3. Univariate methods, however, will place cut points randomly w.r.t. the multivariate distribution of the data, and will hence fail to preserve the interactions and clusters.

To alleviate the above issue, one clearly needs a discretization which can exploit dependencies/interactions among dimensions for better multivariate discretization. That is, one needs an interaction-preserving discretization. Existing methods [Bay01, MPY05] consider a discretization to be of this sort if it places two multivariate regions in the same bin iff the objects in those regions have similar multivariate joint distributions in the other dimensions. That is, it enforces each bin to only contain data of similar distributions. For instance, in Figure 8.1, bins b_1 and b_2 should be merged because the distributions of X_2 and X_3 in the two bins (see Fig. 8.1(c)) are similar. We adopt this view here and propose a solution that overcomes the drawbacks of [Bay01, MPY05].

To perform interaction-preserving discretization, one needs to measure the difference between multivariate distributions in different bins without assuming an underlying distribution. This is especially important when the focus of this chapter is on real-valued data. Well-known measures such as Kullback-Leibler divergence [CT06] and Earth Mover's Distance [PWR89] require assumptions on the data distribution. Second, one needs a bin merge strategy that balances how well interactions are maintained with the level of detail of the discretization. Third, the search space of all possible discretizations is potentially exponential to the number of data points. As a result, one needs efficient methods to find good discretizations. To this end, existing techniques [Bay01, MPY05] do not fully address the first two



(a) (X_1, X_2, X_3) (b) (X_1, X_2)

(c) $p(X_2, X_3 \mid b_1)$ and $p(X_2, X_3 \mid b_2)$ (d) $p(X_2, X_3 \mid b_3)$ and $p(X_2, X_3 \mid b_4)$

Figure 8.1.: Example of the parity problem. There are 4 clusters in (X_1, X_2, X_3) marked by different colors. The correct discretization: one cut point at 0.0 for each dimension.

requirements. In particular, the measure of [Bay01] relies on itemset mining and hence is not entirely suited to real-valued data. The measure of [MPY05] on the other hand captures linear interaction only. Further, both of these works simply perform bottom-up merge of bins, i.e., they lack a formal objective function to achieve the mentioned balance.

With our method, named IPD for Interaction-Preserving Discretization, we tackle each of these problems and overcome the drawbacks of [Bay01, MPY05]. In a nutshell, our method performs multivariate discretization in the sense that it discretizes one dimension while preserving its interactions with *all* other dimensions. We propose to assess the similarity between multivariate distributions in different bins through a new interaction distance, or ID for short. It works on *cumulative distributions*, and hence, does not require any prior assumption on the data distribution. Also because of this, it is not limited to linear interactions. In addition, its computation on empirical data is in closed form, and we prove that it is a metric distance. This ensures easy-to-interpret distance values and reliable assessment of multivariate distributions.

As the second main ingredient of IPD, we define the task of multivariate discretization in terms of the Minimum Description Length (MDL) principle [Ris78]. By opti-

mizing the resulting objective function, IPD is able to balance between preserving dimension interactions and information of the underlying dimension.

Third, we give two efficient methods for finding good discretizations. The first strategy finds the optimal bin merge by dynamic programming, while the second is a fast greedy heuristic. Though both algorithms have the same theoretical complexity, the latter is by far the faster in practice, while providing very high quality results—in fact, we formally prove that it is a $(2-\epsilon)$-approximation of our optimal strategy.

Empirical evaluation of IPD on a wide range of real and synthetic data, and unsupervised and supervised tasks including pattern-based compression, outlier mining, and classification clearly shows IPD to consistently outperform the state of the art in both quality and speed. In short, our main contributions include:

- General notions and a set of abstract desiderata for interaction-preserving multivariate discretization.
- The first interaction distance for quantifying the (dis-)similarity of multivariate distributions over bins, specifically designed for real-valued data and computed in closed form on empirical data.
- An MDL-based framework for finding a balance between complexity and interaction preservation of a discretization.
- Efficient algorithms for discretization, including an optimal solution based on dynamic programming, and a greedy $(2-\epsilon)$-approximation.

The road map of this chapter is as follows. We start by general notions and a set of abstract criteria for interaction-preserving discretization. In Section 8.3 we review related work. Afterward, we introduce IPD, which consists of ID, our new interaction distance (Section 8.4), our MDL-based framework for multivariate discretization (Section 8.5), and efficient algorithms (Section 8.6). We empirically evaluate IPD in Section 8.7. We round up with discussion in Section 8.8 and finally conclude in Section 8.9.

8.2. General Notions

We consider a database $\mathbf{DB}$ of n objects and m dimensions. Each dimension $X_i \in \mathbf{A}$, where $\mathbf{A} = \{X_1, \ldots, X_m\}$, is considered as a continuous-valued random variable. We denote the domain of X_i on $\mathbf{DB}$ as $[min_i, max_i]$. We write $p(X_i)$ for the probability density function (pdf) of the database projected on X_i. Further, we write $p(x_i)$ for $p(X_i = x_i)$. All logarithms are to base 2, and by convention $0 \log 0 = 0$.

A discretization of X_i into k_i bins induces a set of cut points $K_i = \{c_i^1, \ldots, c_i^{k_i-1}\}$, which partitions $[min_i, max_i]$ into k_i *bins*, $[min_i, c_i^1], (c_i^1, c_i^2], \ldots, (c_i^{k_i-1}, max_i]$.

To preserve interactions, two sets of objects should only be in the same bin if they have similar multivariate joint distributions. That is, one should only consider

merging two consecutive bins B^g and B^f of dimension X_i if the distributions of all other dimensions, i.e., $\mathbf{A} \setminus \{X_i\}$, over the objects identified by B^g and B^f are similar. In such a case, we say the bins exhibit similar interactions with regard to the data. To tell whether sets of objects should be in the same bin we need to quantify the interactions of bins. For this purpose, we propose a general notion of interaction distance in the following.

Definition 19. ***Interaction Distance**:*
Assume we want to measure the interaction distance between bins of dimension X_i over target variables $\{Y_1, \ldots, Y_z\}$. In our setting of unsupervised multivariate discretization, the target variables will typically be $\mathbf{A} \setminus \{X_i\}$, i.e., $\{Y_1, \ldots, Y_z\} = \mathbf{A} \setminus \{X_i\}$. Let $\mathcal{B}_i$ be the set of all possible bins on X_i. An interaction distance should be applicable to any two bins of $\mathcal{B}_i$. As such we have $G : \mathcal{B}_i \times \mathcal{B}_i \to \mathbb{R}_0^+$. In general, an interaction distance $G(B^g, B^f)$ with $B^g, B^f \in \mathcal{B}_i$ quantifies the difference of the distributions over variables $\{Y_1, \ldots, Y_z\}$ in two bins B^g and B^f. The more different they are, i.e., the less B^g and B^f interact, the higher their interaction distance $G(B^g, B^f)$ will be. Formally,

$$G(B^g, B^f) \sim \mathit{diff}(p(Y_1, \ldots, Y_z|B^g), p(Y_1, \ldots, Y_z|B^f)) \quad .$$

In order to facilitate assessment of existing techniques, we introduce four desired properties of meaningful interaction distances.

- **Property 1 (Unsupervised)**: G does not require labeled data.
- **Property 2 (Non-negativity)**: For any two bins B^g and B^f, $G(B^g, B^f) \geq 0$.
- **Property 3 (Zero interaction)**: $G(B^g, B^f) = 0$ if and only if the distributions of $\{Y_1, \ldots, Y_z\}$ in B^g and B^f are identical.
- **Property 4 (Non-parametric)**: G should not require prior assumptions on either distributions or correlations.

Properties 1 and 4 are mandatory to ensure the generality of the discretization scheme: to be applicable for exploratory data analysis, to be applicable on unlabeled data, as well as easily computable on empirical data. In particular, one should only use data distribution functions that can be computed directly from empirical data with no prior assumptions or given labels. Properties 2 and 3 guarantee that the distance properly quantifies the difference in data distributions of any two bins.

8.3. Related Work

Before introducing our approach, we review the literature. We divide related work into four categories: univariate discretization, multivariate discretization, assessment of dimension interactions, and other related work.

8.3.1. Univariate discretization

First, we consider univariate solutions, which include standard approaches such as equal-width and equal-frequency binning, as well as state of the art methods such as UD [KM07] and its close cousin Bayesian Blocks [SNJ+13]. All of these methods discretize dimensions individually, without considering other dimensions. By definition, they do not preserve interactions. As they neither instantiate G, Properties 1–4 are not applicable.

Supervised techniques [FI93, Ker92] aim to preserve interactions with a target class label. As such, they instantiate $G(B^g, B^f)$ by the difference between the class label distributions of two bins. To measure this difference, ChiM [Ker92] employs χ^2 test. By requiring prior information on class labels, supervised methods do not match Property 1. More importantly, as only interactions with class labels are preserved, the discretized data preserves only structure related to the given class labels. In contrast, IPD aims to preserve all major interactions.

8.3.2. Multivariate discretization

Ferrandiz and Boullé [FB05] proposed a supervised multivariate discretization technique. Though multivariate, its objective function is tightly coupled with the class label distribution.

Kang et al. [KWL+06] introduced an unsupervised multivariate technique based on ICA. However, due to approximation [SRPP11], ICA transformation is not guaranteed to preserve all important interactions. As a result, it is not guaranteed to fulfill Property 3. Further, it does not meet Property 4 as it implicitly assumes the dimensions to be non-Gaussian in the transformed space.

MVD [Bay01] instantiates $G(B^g, B^f)$ by means of STUCCO [BP99], a contrast set mining algorithm. Two bins B^g and B^f are considered similar if no itemset can be found that separates the two. However, by considering continuous values as items, the support of most itemsets is very low, which leads to high false alarm rates. Hence, in general, MVD does not meet Property 3. CPD [MPY05] transforms the data using PCA, mines itemsets on the eigenspaces of the bins, and instantiates $G(B^g, B^f)$ as the Jaccard coefficient between the resulting collections. By using PCA, it can only capture and preserve linear correlations [LV07]. As such, it may miss complex interactions, and is hence not guaranteed to satisfy Property 3. In addition, neither MVD nor CPD are designed to work directly with data distribution functions, and hence, neither meet Property 4.

Both MVD and CPD employ a heuristic bottom-up approach as their binning strategy. That is, two consecutive bins are merged if their interaction distance is low. As such, the binning strategy of both MVD and CPD can be viewed as a hierarchical clustering where no objective function is directly optimized. In contrast, univariate methods UD [KM07] and SD [FI93] search for bins based on the MDL principle [Grü07]. That is, they seek the bins that yield the best balance between goodness of fit and model complexity. Their encodings are designed for univariate discretization, and hence, only reward precision, not the preservation of the multivariate interactions of the data.

8.3.3. Assessing dimension interactions

An interaction distance quantifies differences between two multivariate data distributions. In principle, one could use Kullback-Leibler divergence or a variant, such as Jensen-Shannon divergence [CT06]. To apply these, one needs to assume a distribution, or estimate the multivariate pdf—which involves hard parametrization [LV07]. Advanced density estimation techniques such as kernel methods [Sil86] require selecting a kernel function and bandwidth. Other popular measures include Earth Mover's Distance [PWR89], which requires a probability mass function.

In contrast, we employ cumulative distribution functions (cdfs), which do not require assumptions on the data distribution and do not have free parameters. Further, they can be computed in closed form. In theory, smoother estimates can be obtained through cdf kernel estimation [LY08]. However, similar to pdf kernel estimation, performance depends on the chosen kernel. In addition, though theoretical optimal bandwidth selection exists, its realization needs estimation.

Interactions among dimensions can encompass different types of relationship among dimensions; one of which is correlation. Our work here deals with multivariate discretization and does not directly address such correlation analysis. However, it benefits correlation analysis in the sense that we also aim to preserve interactions among dimensions during the discretization process. Further, many correlation measures are based on Shannon entropy, and hence, rely on discrete data. With multivariate discretization, we aim at a general contribution to enhance a variety of techniques, e.g., mutual information and total correlation, rather than proposing a single solution improving one specific notion of correlation directly on real-valued data.

8.3.4. Other related work

Lakshmanan et al. [LNW+02] and Bu et al. [BLN05] studied grouping cells of data cubes satisfying a given property, e.g., frequencies higher than a pre-specified threshold. The methods are designed for cell properties for which the validation does not involve other cells. In other words, they do not check if cells interact, and hence, do not address the issue of preserving interactions.

Aue et al. [AHHR09] discussed detecting changes in multivariate time series. Their problem setting is different from ours in two main aspects: (a) they focus on covariance matrices, i.e., second-order pairwise interactions, and (b) break points signify changes in all dimensions, i.e., each set of cut points correspond to one data point. [PP13] focus on a similar problem, but instead of covariance, targets autocovariance. Like [AHHR09], it is also constrained to pairwise interactions.

The work by [All83, AF94] discusses a representation of time that uses temporal intervals as primitives. It can be used to derive intervals, and for event change detection. While the exact relationship between events can be unknown, some prior knowledge on how they could be temporally related is required.

8.4. Interaction Distance

To construct IPD, we start by introducing our interaction distance, ID. Quickly going over its properties, ID does not require any prior knowledge such as class labels, and hence, satisfies Property 1. In the following we will prove that ID meets Properties 2 and 3, as well as show that ID is computed in closed form on empirical data, i.e., it does not need to make any prior assumption on the data distribution. Therefore, it also meets Property 4. The details of ID are as follows.

In principle, when discretizing $X_i \in \mathbf{A}$, to preserve its interactions with all other dimensions, we only consider merging two consecutive bins B^g and B^f of X_i if the distributions of $\mathbf{A} \setminus \{X_i\}$ over the objects identified by B^g and B^f are similar. That is, we instantiate $G(B^g, B^f)$ by $\mathit{diff}(p(\mathbf{A} \setminus \{X_i\}|B^g), p(\mathbf{A} \setminus \{X_i\}|B^f))$.

Typically, the diff function measures the difference between two multivariate pdfs corresponding to two consecutive bins of any dimension. Formally, we consider two pdfs p and q defined on the set of variables $\mathbf{A} = \{X_1, \ldots, X_m\}$. In practice, we want to measure the differences over $\mathbf{A} \setminus \{X_i\}$. For notational convenience, we however write $\mathit{diff}(p(\mathbf{A}), q(\mathbf{A}))$ instead of $\mathit{diff}(p(\mathbf{A} \setminus \{X_i\}), q(\mathbf{A} \setminus \{X_i\}))$. The domain of $\mathbf{A}$ is $\Omega = [min_1, max_1] \times \ldots \times [min_m, max_m]$. Though the analysis below considers all dimensions, we note that our discussion holds for any $\mathbf{A} \setminus \{X_i\}$.

To solve the problem of measuring $\mathit{diff}(p(\mathbf{A}), q(\mathbf{A}))$ we propose ID, a function for quantifying the difference between distributions. In consistence with the chapters presented so far, our goal is to design ID such that it does not require any prior assumption and allows non-parametric computation on empirical data. Hence, we formulate ID using cumulative distributions that can be determined directly from data samples. This helps ID to address Property 4. In particular, let $P(\mathbf{A})$ and $Q(\mathbf{A})$ be the cumulative distribution function of $p(\mathbf{A})$ and $q(\mathbf{A})$, respectively. That is, for any vector $\mathbf{a} = \{a_1, \ldots, a_m\} \in \Omega$, we have

$$P(\mathbf{a}) = \int_{min_1}^{a_1} \ldots \int_{min_m}^{a_m} p(x_1, \ldots, x_m) dx_1 \cdots dx_m$$

and similarly for $Q(\mathbf{a})$. We define our function ID as follows:

Definition 20. ***Interaction Distance*** ID*:*
The interaction distance ID between $p(\mathbf{A})$ and $q(\mathbf{A})$, denoted as $ID(p(\mathbf{A}) \mid\mid q(\mathbf{A}))$, is defined as $\sqrt{\int_{\Omega} (P(\mathbf{a}) - Q(\mathbf{a}))^2 d\mathbf{a}}$.

In other words, ID quantifies the difference between $p(\mathbf{A})$ and $q(\mathbf{A})$ by (a) integrating the squared difference of their respective cumulative distributions, and (b) taking the square root of this integral (we go for square root because it is useful later for proving that ID is a metric distance). Thus, ID is reminiscent of the correlation measure proposed in Chapter 4 and those in [RSX+11, SRPP11], i.e., the class of quadratic measure. And similarly to many of those measures, shortly we will show that ID permits computation on empirical data in closed form.

Now we need to show that ID properly quantifies the difference of distributions, and hence, is suitable for analyzing the interaction of any two bins. To accomplish this, we derive the following theorem directly from Definition 20:

Theorem 23. *$ID(p(\mathbf{A}) \mid\mid q(\mathbf{A})) \geq 0$ with equality iff $p(\mathbf{A}) = q(\mathbf{A})$.*

Based on Theorem 23, we can see that ID meets Properties 2 and 3, i.e., it is appropriate for interaction analysis. One side question that we are interested in knowing is that whether ID is a metric distance. To answer this question, in the following we prove that ID satisfies the triangle inequality. In particular, let $r(\mathbf{A})$ be a pdf defined on $\mathbf{A}$ and $R(\mathbf{A})$ is its cdf. We have:

Theorem 24. *$ID(p(\mathbf{A}) \mid\mid r(\mathbf{A})) + ID(r(\mathbf{A}) \mid\mid q(\mathbf{A})) \geq ID(p(\mathbf{A}) \mid\mid q(\mathbf{A}))$.*

Proof. Let $H(\mathbf{A}) = P(\mathbf{A}) - R(\mathbf{A})$ and $G(\mathbf{A}) = R(\mathbf{A}) - Q(\mathbf{A})$. The inequality becomes

$$\sqrt{\int_{\Omega} H^2(\mathbf{a}) d\mathbf{a}} + \sqrt{\int_{\Omega} G^2(\mathbf{a}) d\mathbf{a}} \geq \sqrt{\int_{\Omega} \left(H(\mathbf{a}) + G(\mathbf{a})\right)^2 d\mathbf{a}} \quad ,$$

which in turn is equivalent to

$$\sqrt{\int_{\Omega} H^2(\mathbf{a}) d\mathbf{a} \cdot \int_{\Omega} G^2(\mathbf{a}) d\mathbf{a}} \geq \int_{\Omega} H(\mathbf{a}) G(\mathbf{a}) d\mathbf{a} \quad ,$$

which is also known as Hölder's inequality. □

Following Theorems 23 and 24, and the fact that ID is symmetric, we conclude it is a distance metric. This characteristic ensures easy-to-interpret distance values, as well as reliable assessment of multivariate distributions. And this is also the reason why we use square root in the formulatiom of ID: It helps us to achieve a metric distance rather than a mere divergence measure. Besides being a metric, another advantage of ID for multivariate discretization is that its values on empirical data can be described in closed form. In particular, assume that the empirical data forming $p(\mathbf{A})$ contains data points $\{R_1, \ldots, R_k\}$ of $\mathbf{DB}$. Analogously, we denote $\{S_1, \ldots, S_l\}$ as the data points forming $q(\mathbf{A})$. For each $R \in \{R_1, \ldots, R_k\}$, we write R^i for R projected onto the dimension X_i. We define S^i similarly for any $S \in \{S_1, \ldots, S_l\}$. We prove the following result:

Theorem 25. *$ID(p(\mathbf{A}) \mid\mid q(\mathbf{A}))$ equals to*

$$\left(\frac{1}{k^2}\sum_{j_1=1}^{k}\sum_{j_2=1}^{k}\prod_{i=1}^{m}\left(max_i - \max(R_{j_1}^i, R_{j_2}^i)\right)\right.$$
$$-\frac{2}{kl}\sum_{j_1=1}^{k}\sum_{j_2=1}^{l}\prod_{i=1}^{m}\left(max_i - \max(R_{j_1}^i, S_{j_2}^i)\right)$$
$$\left.+\frac{1}{l^2}\sum_{j_1=1}^{l}\sum_{j_2=1}^{l}\prod_{i=1}^{m}\left(max_i - \max(S_{j_1}^i, S_{j_2}^i)\right)\right)^{1/2} \quad .$$

Proof. Our proof is based on that in Chapter 4 and those in [RSX+11, SRPP11]. Let $ind(\alpha)$ be an indicator function with value 1 if α is true and 0 otherwise. It holds that

$$P(\mathbf{a}) = \int_{min_1}^{max_1} \dots \int_{min_m}^{max_m} ind(x_1 \leq a_1) \cdots ind(x_m \leq a_m) p(x_1, \dots, x_m) dx_1 \cdots dx_m$$

Using empirical data, we hence have

$$P(\mathbf{a}) = \frac{1}{k}\sum_{j=1}^{k}\prod_{i=1}^{m} ind(R_j^i \leq a_i) \quad , \quad \text{and} \quad Q(\mathbf{a}) = \frac{1}{l}\sum_{j=1}^{l}\prod_{i=1}^{m} ind(S_j^i \leq a_i) \quad ,$$

and therefore $[ID(p(\mathbf{A}) \mid\mid q(\mathbf{A}))]^2$ equals to

$$\int_{min_1}^{max_1} \dots \int_{min_m}^{max_m} \left(\frac{1}{k}\sum_{j=1}^{k}\prod_{i=1}^{m} ind(R_j^i \leq a_i) - \frac{1}{l}\sum_{j=1}^{l}\prod_{i=1}^{m} ind(S_j^i \leq a_i)\right)^2 da_1 \cdots da_m$$

Expanding the above term and bringing the integrals inside the sums, we have

$$\frac{1}{k^2}\sum_{j_1=1}^{k}\sum_{j_2=1}^{k}\prod_{i=1}^{m}\int_{min_i}^{max_i} ind(\max(R_{j_1}^i, R_{j_2}^i) \leq a_i) da_i$$
$$-\frac{2}{kl}\sum_{j_1=1}^{k}\sum_{j_2=1}^{l}\prod_{i=1}^{m}\int_{min_i}^{max_i} ind(\max(R_{j_1}^i, S_{j_2}^i) \leq a_i) da_i$$
$$+\frac{1}{l^2}\sum_{j_1=1}^{l}\sum_{j_2=1}^{l}\prod_{i=1}^{m}\int_{min_i}^{max_i} ind(\max(S_{j_1}^i, S_{j_2}^i) \leq a_i) da_i \quad ,$$

by which we arrive at the final result. □

By Theorem 25 we can see that *ID* permits direct computation on empirical data in closed form, without assumptions on the data distribution. That is, *ID* meets Property 4. In the remainder of this chapter we will use *ID* to implement *diff* in Definition 19. In particular, we set $diff(p(\mathbf{A}), q(\mathbf{A}))$ to $ID(p(\mathbf{A}) \mid\mid q(\mathbf{A}))$, and

will employ it in our practical score (Sec. 8.5.2) for identifying good interaction-preserving discretizations.

8.5. Identifying the Optimal Discretization

Our overall goal is to find a discretization that balances preserving interactions and detail of the data. We introduce a global objective function for identifying the optimal multivariate discretization—which can be used to compare fairly between *any* discretization—and a practical variant that allows for easier optimization.

We regard the problem of discretization as a model selection problem. Hence, in order to select the best model we need an appropriate model selection criterion. As we explicitly do not want to assume prior distributions, the Minimum Description Length (MDL) principle [Ris78] makes for a natural and well-founded choice.

Loosely speaking, MDL identifies the best model as the model that obtains the best lossless compression of the data. More formally, given a set of models $\mathcal{M}$, the best model $M \in \mathcal{M}$ is identified as the one that minimizes

$$L(\mathbf{DB}, M) = L(M) + L(\mathbf{DB} \mid M)$$

where $L(M)$ is the length, in bits, of the description of the model M, and $L(\mathbf{DB} \mid M)$ is the length, again in bits, of the description of the data $\mathbf{DB}$ as encoded by M. That is, MDL helps select a model that yields the best balance between goodness of fit and model complexity. To ensure fair comparison, MDL requires lossless encodings.

In the following, we will define interaction preserving discretization in terms of MDL. First, we will discuss our *ideal objective function* in Section 8.5.1, which allows for fair comparison between any discretization; yet, however, does not lend itself for fast optimization. To facilitate efficient search, in Section 8.5.2 we extend it into a *practical score* that uses ID and does allow for efficient search.

8.5.1. MDL for interaction-preserving discretization

In the context of discretization, let dsc be a discretization of $\mathbf{DB}$, and $dsc(\mathbf{DB})$ be the discretized data that dsc creates on $\mathbf{DB}$ (see Figure 8.2). To search for the optimal interaction-preserving discretization, we aim at constructing an objective function based on lossless MDL encoding. To achieve this, for each candidate discretization dsc, we need to encode (a) dsc itself, (b) $dsc(\mathbf{DB})$, and (c) the cost of reaching the continuous-valued entries of $\mathbf{DB}$ from the discrete values in $dsc(\mathbf{DB})$. The third component is very important for a lossless encoding. We denote it as $\mathbf{DB} \ominus dsc(\mathbf{DB})$. Based on our analysis, we formulate the following objective function (which we name the *ideal score*):

$$L(\mathbf{DB}, dsc) = \underbrace{L(dsc) + L(dsc(\mathbf{DB}))}_{L(M)} + \underbrace{L(\mathbf{DB} \ominus dsc(\mathbf{DB}))}_{L(\mathbf{DB}|M)}$$

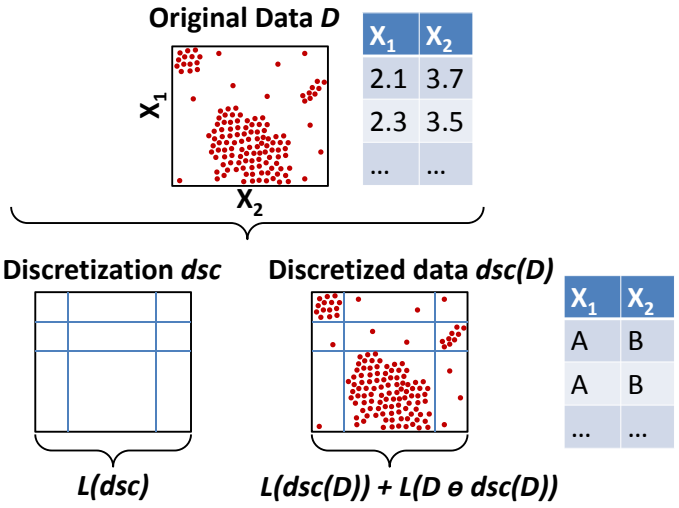


Figure 8.2.: Example of discretization component costs. $L(dsc)$ is the encoding cost of a discretization dsc of $\mathbf{DB}$. In this example, dsc is a discretization of *both* X_1 and X_2, i.e., here dsc is a 2-dimensional grid. $L(dsc(\mathbf{DB}))$ is the cost of encoding the discretized data $dsc(\mathbf{DB})$. Finally, $L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ is the cost for encoding the exact data points within each bin.

where $L(dsc)$ is the encoding cost of dsc, $L(dsc(\mathbf{DB}))$ is the encoding cost of the discretized data $dsc(\mathbf{DB})$ of $\mathbf{DB}$, and $L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ is the encoding cost of $\mathbf{DB} \ominus dsc(\mathbf{DB})$.

We now give the intuition on how our ideal score helps to identify the best interaction-preserving discretization. First, assume a discretization dsc that is too detailed, i.e., it splits dimensions into overly many bins. That is, it is difficult (up to impossible) to detect interactions in $dsc(\mathbf{DB})$—i.e., $L(dsc(\mathbf{DB}))$ will be very high. At the same time, as the bins are small identifying exact values within them is easy—i.e., $L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ will be low.

Alternatively, assuming dsc is very coarse, i.e., data points are grouped into too few, or even only 1 bin. $dsc(\mathbf{DB})$ will now show interactions even when there are none in $\mathbf{DB}$. It will be easy to compress this data, and $L(dsc(\mathbf{DB}))$ will be low. However, there are now many more possible values per bin, and hence, encoding the exact values of the data costs many more bits—$L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ will be very high. The optimal discretization dsc^* is the discretization that is neither too detailed or too coarse: the one that maintains the true interactions of $\mathbf{DB}$. Hence, with the ideal score we can identify the best interaction-preserving discretization dsc^*: It is the one that minimizes our score function. Below we explain in details how to quantify each component of this score.

Encoding the discretization. Encoding the discretization grid dsc means encoding the number of bins, and their cut points, per dimension X_i. Technically, we first encode the number of bins. Assume that X_i has k_i bins, i.e., $(k_i - 1)$ cut points. When encoding the number of bins k_i, our goal is that the larger this integer is, the

more bits it costs. Thus, we use $L_{\mathbb{N}}$, the MDL-optimal encoding of integers [Ris83]. It is defined for $z \geq 1$, with $L_{\mathbb{N}}(z) = \log^*(z) + \log c_0$, where $\log^*(z) = \log(z) + \log\log(z) + \ldots$, where we only include the positive terms, and c_0 is set to 2.8654 to make it a valid encoding by ensuring that all probabilities sum to 1. We can see that the larger an integer is, the larger its $L_{\mathbb{N}}$ value, i.e., our objective is met.

Next, to encode the locations of the cut points, we note that digitally stored data is recorded at a finite resolution [KM07], and hence, the resolution res_i of a dimension X_i is data-dependent. For example, let us assume that dimension X_i has domain $[0, 1]$. From the data we can observe that it is encoded with, e.g., 2 significant digits, or, at resolution of $res_i = 0.01$. Given the resolution res_i for a dimension X_i, we have that every possible value of X_i belongs to the set $\mathcal{V}_i = \{min_i + z \cdot res_i : z = 0, 1, \ldots, n_i\}$ where $n_i = \frac{max_i - min_i}{res_i}$.

As we have no prior expectation on their location, any set of $(k_i - 1)$ distinct cut points is equally likely, and hence, data-to-model codes are optimal [VV04]. A data-to-model code is an index into a canonically ordered enumeration of all possible data (i.e., values) given the model (the provided information). Here, we know $(k_i - 1)$ cut points have to be selected out of n_i candidates; a choice for which there are $\binom{n_i}{k_i - 1}$ possibilities. Assuming a canonical order, $\log\binom{n_i}{k_i - 1}$ gives the number of bits to identify the actual set of cut points. As such, we have

$$L(dsc) = \sum_{i=1}^{m} L_{\mathbb{N}}(k_i) + \log\binom{n_i}{k_i - 1}$$

for the number of bits required to encode a discretization dsc. Next, we discuss how to encode the discretized data, $dsc(\mathbf{DB})$.

Encoding the discretized data. To encode discrete data, in general we need to choose the best lossless compressor C out of a set $\mathcal{C}$ of all lossless compressors applicable to discrete data of the form $dsc(\mathbf{DB})$ [LV93, VvS11]. We denote the cost of encoding $dsc(\mathbf{DB})$ using C as $L(dsc(\mathbf{DB}), C)$. The cost of identifying C from $\mathcal{C}$ is $\log|\mathcal{C}|$, assuming that each compressor is equally likely. The length of the discretized data $dsc(\mathbf{DB})$ in bits, $L(dsc(\mathbf{DB}))$, is thus given by

$$L(dsc(\mathbf{DB})) = \log|\mathcal{C}| + \min_{C \in \mathcal{C}} L(dsc(\mathbf{DB}), C) \quad .$$

This cost function is ideal as it minimizes over all possible compressors. That is, it can detect and reward *any* interaction present in the discretized data. However, this general form is not very practical: We do not have access to all applicable compressors, nor the time to evaluate them all. To use the score, we will have to instantiate $\mathcal{C}$.

To this end, any compressor suited for categorical data can be used. naïvely, we can even encode dimensions independently using prefix codes [CT06]. However, as our aim is to find interaction-preserving discretizations, we should rather use a compressor that is interaction-aware. That is, one that can detect and reward correlations over dimensions. For instance, we could first serialize the data row

per row, and then use GZIP or one of its many variants. This, however, would be very sensitive to the serialization order of the data. Better choices hence include modern itemset-based compressors, such as KRIMP [VvS11], COMPREX [ATVF12], MTV [MVT12], or PACK [TV08], as these can detect interactions among dimensions independently to the order of the data. Each can be used as plug in for $L(dsc(\mathbf{DB}), C)$—in our experiments we will evaluate using a number of applicable compressors.

For the following, let us assume we have chosen a suited compressor, i.e., that we can calculate $L(dsc(\mathbf{DB}))$. We will now finalize the score by discussing how to reconstruct the continuous input data $\mathbf{DB}$ given the discretized data $dsc(\mathbf{DB})$.

Encoding the errors. In order to make our score lossless, a necessary requirement in MDL, we have to formalize how to compute the encoding cost of reaching the continuous-valued entries of $\mathbf{DB}$ from the discrete values in $dsc(\mathbf{DB})$.

Let us write $L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ for the number of bits required to identify the exact data points within their respective multivariate cell. This cost can be factorized per dimension, that is, per univariate bin. Hence, over all data points, we have

$$L(\mathbf{DB} \ominus dsc(\mathbf{DB})) = \sum_{i=1}^{m} L(X_i \ominus dsc(X_i)) \quad .$$

For computation purposes, it is more convenient to aggregate this cost per bin. Let $\{B_i^1, \ldots, B_i^{k_i}\}$ be the set of bins induced by dsc on dimension X_i. We write $|B_i^j|$ as the number of values of X_i that B_i^j contains. We then have

$$L(X_i \ominus dsc(X_i)) = \sum_{j=1}^{k_i} |B_i^j| \log \left(\left\lfloor \frac{ub(B_i^j) - lb(B_i^j)}{res_i} \right\rfloor + 1 \right)$$

as the cost of reaching the actual values for a dimension X_i given the discretized representation $dsc(X_i)$, where $ub(B_i^j)$ is the upper bound of bin B_i^j, $lb(B_i^j)$ its lower bound. With the resolution res_i of X_i, we have $\left(\left\lfloor \frac{ub(B_i^j) - lb(B_i^j)}{res_i} \right\rfloor + 1 \right)$ as the number of possible values in B_i^j.

Summing Up. With the above three elements we can construct our ideal score:

$$L(\mathbf{DB}, dsc) = \underbrace{L(dsc) + L(dsc(\mathbf{DB}))}_{L(M)} + \underbrace{L(\mathbf{DB} \ominus dsc(\mathbf{DB}))}_{L(\mathbf{DB}|M)} \quad .$$

The best discretization dsc^* is identified as the one that minimizes this score. Another benefit of this score is that it allows for fair and unbiased comparison between *any* discretization discovered by *any* discretization method—simply by instantiating it using different compressors, and comparing the total number of bits. We will use it as such in our experiments in Section 8.7.3.

Though ideal for identifying the optimal discretization, the score does not lend itself for fast optimization towards that goal. For example, it does not factor over

dimensions, and so we would have to discretize all dimensions concurrently. However, for multivariate data the search space is exponential to both n and m, and hence, restrictively large in practice. Moreover, we do not have access to the optimal compressor C^*, and can hence not compute $L(dsc(\mathbf{DB}))$ directly. In theory we could approximate C^* by instantiation $\mathcal{C}$ with a collection of compressors, but this could lead to erratic behavior. Most importantly, though, is that we aim for a fast general approach for high quality interaction-preserving discretization, and hence, want to avoid including computationally expensive heuristics in our objective that only reward specific types of interaction, such as [VvS11, ATVF12]. To accomplish this, in the next section, we will take the ideal score and adapt it into a practical score that is independent of specific compressors, can detect and reward interactions in general, and does allow for efficient optimization.

8.5.2. A fast optimizable score

In this section we will discuss our *practical score*, which maintains key properties of our ideal score, yet does allow for efficient optimization. From the ideal score, we can see that the source of inefficiency comes from the fact that the score is not factorizable per dimension, which requires us to process all dimensions simultaneously. Thus, to enable efficient processing, we need a score that allows us to identify the optimal discretization per dimension, while still considering its interactions with the other dimensions. Second, the score should not use any specific compressor to avoid biases. Our practical score is designed for to meet these goals. In short, it is defined per dimension and hence permits efficient optimization. In addition, it relies on ID to evaluate whether interactions are maintained, independent of any specific compressor. Third, to avoid problems of insufficient data for meaningful statistical assessment [LV07], it considers the data at the level of *micro bins* instead of the individual objects.

In particular, we form micro bins for X_i by partitioning its interval into T_i fine-grained bins (e.g., by clustering). For each X_i, let $\mathbf{M}_i$ be the corresponding set of micro bins. To avoid confusion, we refer to bins B_i^j of dsc_i as *macro* bins.

With T_i micro bins, we have $(T_i - 1)$ cut points. Merging these micro bins into k_i macro bins $B_i^1, \ldots, B_i^{k_i}$ (each B_i^j contains $|B_i^j|$ micro bins) means choosing $(k_i - 1)$ out of $(T_i - 1)$ cut points. As such, a discretization for dimension X_i corresponds to a subset of all cut points, where the empty subset corresponds to merging all micro bins of X_i into just one macro bin, and the full set corresponds to the input micro bins. Given a discretization dsc_i of X_i, we denote $dsc_i(\mathbf{M}_i)$ as the resulting discretized data of X_i, i.e., the resulting set of macro bins.

The building blocks of the score are analogue to the ideal score (Equation 8.5.1), yet now defined per dimension and defined over micro bins. We will discuss its terms in detail below.

Encoding the discretization. The intuition for encoding dsc_i is identical to the

ideal score. We have

$$L(dsc_i) = L_{\mathbb{N}}(k_i) + \log \binom{T_i - 1}{k_i - 1} \quad ,$$

where we encode the number of bins, and identify the cut points from $(T_i - 1)$ options.

Encoding the discretized data. Our goal is to avoid optimizing towards a specific compressor. Thus, we will use ID to determine how well a discretization dsc_i maintains the interactions of the data, independent of any particular compressor. Hence, we define the cost of encoding the discretized data $dsc_i(\mathbf{M}_i)$ as

$$L(dsc_i(\mathbf{M}_i)) = L_{bid}(dsc_i(\mathbf{M}_i)) + L_{mh}(dsc_i(\mathbf{M}_i))$$

where $L_{bid}(.)$ is the cost of the discretized data under the independence model, and $L_{mh}(.)$ is a penalty on the multivariate heterogeneity of the discretization. The intuition of $L_{mh}(.)$ is that two micro bins with different data distributions, as identified by ID, should stay separate. If a discretization combines them, we penalize accordingly. With this, we guarantee interaction preservation. More details are given below.

Encoding the macro bin ids. Encoding the discretized data means encoding the macro bin id per micro bin. We do this by assigning optimal prefix codes to the macro bins, the lengths of which we calculate by Shannon entropy. The code length of the id of macro bin B_i^j then is $-\log \frac{|B_i^j|}{T_i}$. Over all macro bins, we have

$$L_{bid}(dsc_i(\mathbf{M}_i)) = \sum_{j=1}^{k_i} \left(L_{\mathbb{N}}(|B_i^j|) - \log \frac{|B_i^j|}{T_i} - |B_i^j| \log \frac{|B_i^j|}{T_i} \right) \quad .$$

where the first term encodes the number of micro bins in B_i^j, the second is the cost of the macro bin code in the dictionary, and the third term is the cost of using this code to identify the macro bin per associated micro bin.

Penalizing multivariate heterogeneity. The above encoding is lossless, but unaware of interactions. To make it interaction-aware, we include a penalty term based on ID. The intuition is to reward regions with similar distributions to be in the same hypercubes (cubes in the space formed by all dimensions), and vice-versa. That is, two micro bins with different data distributions, as identified by ID, should stay separate. If a discretization combines them, we penalize accordingly.

We penalize on the number of *break points* in a macro bin—the indexes of consecutive micro bins within the macro bin for which the interaction distance is large. Intuitively, more break points should result in a higher encoding cost. More formally, let us consider an arbitrary macro bin B_i^j with micro bins $b_i^{j,1}, \ldots, b_i^{j,|B_i^j|}$. For each pair of consecutive micro bins $b_i^{j,w}$ and $b_i^{j,w+1}$ in B_i^j, if their interaction distance is large, we will encode index w to represent a break point between the

distributions. We write $I(B_i^j)$ for the set of indices of these break points, with

$$I(B_i^j) = \{w \in [1, |B_i^j| - 1] \mid \\ ID(p(\mathbf{A} \setminus \{X_i\}|b_i^{j,w}) \mid\mid p(\mathbf{A} \setminus \{X_i\}|b_i^{j,w+1})) \text{ is large}\} \quad .$$

This allows us to include the cost of encoding $I(B_i^j)$ in $L_{mh}(dsc_i(\mathbf{M}_i))$. (We will discuss how to decide if an interaction distance is large in Section 8.6.)

To ensure we only penalize when interactions are broken, L_{mh} includes only those macro bins for which $I(B_i^j)$ is non-empty. Formally, we define

$$L_{mh}(dsc_i(\mathbf{M}_i)) = \sum_{j=1:\ |I(B_i^j)|>0}^{k_i} L_{\mathbb{N}}(|I(B_i^j)|) + |I(B_i^j)| \log(|B_i^j| - 1)$$

where we encode the number of break points by $L_{\mathbb{N}}$, and encode $I(B_i^j)$ using optimal prefix codes. Here, this entails identifying the index of each break point out of $(|B_i^j| - 1)$ possible pairs, which hence costs $\log(|B_i^j| - 1)$ bits per index. This penalty captures our intuition: The more micro bins with different multivariate distributions in a macro bin, the higher its cost. Combined, L_{bid} and L_{mh} tell us how well a discretization maintains the interactions and detail of the data.

Encoding the errors. With the above we know the discrete data. The final step is to reconstruct the data up to the micro bin ids. As we know the number of micro bins per macro bin from $dsc_i(\mathbf{M}_i)$, here we only have to identify the ids of the micro bins. Using optimal prefix codes, we have

$$L(\mathbf{M}_i \ominus dsc_i(\mathbf{M}_i)) = \sum_{j=1}^{k_i} |B_i^j| \log |B_i^j| \quad .$$

Note that we do not have to reconstruct the original data up till the exact values of X_i for fair model selection. This has two reasons. First, recall that our practical score only considers data up to the resolution of the micro bins: it does not 'see' the data in higher detail. Second, the cost of encoding the exact values of X_i for a given set of micro bins $\mathbf{M}_i$ is constant over all models. Hence, we can safely ignore it here.

Summing Up. We have now completed the definition of our practical score, which aims to identify the best interaction-preserving discretization *per* dimension. Formally, we aim at finding the discretization dsc_i^* per dimension X_i that minimizes

$$L(\mathbf{M}_i, dsc_i) = L(dsc_i) + L(dsc_i(\mathbf{M}_i)) + L(\mathbf{M}_i \ominus dsc_i(\mathbf{M}_i)) \quad .$$

It is easy to see that its terms are analogue to the ideal score (Equation 8.5.1), and though there exists no formal connection between our two scores, the general intuition is identical: both reward and punish similarly. Intuitively, a good solution under the practical score is also a good solution under the ideal score.

We do note that our practical score is independent of res_i. In addition, it ad-

Algorithm 5: IPD

1 **for** *each dimension* $X_i \in \mathbf{A}$ **do**
2 $\quad A_i \leftarrow \mathbf{A} \setminus \{X_i\}$
3 $\quad$ Form micro bins $\{b_i^1, \ldots, b_i^{T_i}\}$ for X_i
4 $\quad$ **for** $w = 1$ **to** $T_i - 1$ **do**
5 $\quad\quad id_i^w = ID(p(A_i|b_i^w) \mid\mid p(A_i|b_i^{w+1}))$
6 $\quad$ Macro bins $\{B_i^1, \ldots, B_i^{k_i}\} \leftarrow$ Merge of $\{b_i^1, \ldots, b_i^{T_i}\}$ using $\{id_i^1, \ldots, id_i^{T_i-1}\}$

dresses the lack of the optimal compressor, and can be optimized independently per dimension. Furthermore, it can be instantiated by any interaction distance. In this chapter, we use ID as an instantiation since it yields a good combination of theoretical correctness and foundations, simplicity, and ease of computation.

8.6. The IPD Algorithm

Having introduced the theoretical model of IPD, which consists of our interaction distance ID and our MDL-based score, we now detail our algorithmic approach. In order, we will first give two efficient bin merge strategies, then discuss parameter settings, and finally we will analyze the time complexity of our algorithms.

8.6.1. Algorithms

We will now discuss the IPD algorithm, for which we give the pseudo code as Algorithm 5. First, we pre-process the data to obtain micro bins (Line 3), after which for every pair of consecutive micro bins we use ID to calculate their interaction. The most important step in IPD is the bin merge strategy on Line 6. Given T_i micro bins, there are 2^{T_i-1} merge possibilities, which is too many to evaluate exhaustively. For example, for $T_i = 50$ we already have more than 1 trillion options. To tackle this problem, we prove that the optimal merge of micro bins can be found in polynomial time by dynamic programming. The details are as follows.

Optimal bin merge strategy IPD$_{opt}$. We show that the search space of bin merges for a dimension X_i is structured, i.e., intermediate results can be re-used to avoid redundant computation. In particular, let $F(c, k)$ be the minimum total encoding cost over all merges of the first c micro bins of X_i ($1 < c \leq T_i$) producing k macro bins ($1 < k \leq c$). For each l with $k-1 \leq l < c$, consider a merge of the first c micro bins that combines the first l of them into $(k-1)$ macro bins, and combines the remaining $(c-l)$ micro bins into its kth macro bin $B_i^{k,l}$. We arrive at the following theorem showing a recursive formulation of $F(c, k)$, which gives way to efficiently computing it using dynamic programming, and hence, to efficiently identifying the optimal merge of micro bins.

Theorem 26. *$F(c, k)$ is equal to*

$$\begin{aligned}
&\min_{k-1\leq l<c}\{F(l,k-1)\\
&+L_{\mathbb{N}}(k)+\log\binom{c-1}{k-1}-L_{\mathbb{N}}(k-1)-\log\binom{l-1}{k-2}\\
&+L_{\mathbb{N}}(|B_i^{k,l}|)-\log\frac{|B_i^{k,l}|}{c}-(k-1)\log\frac{c-|B_i^{k,l}|}{c}\\
&-|B_i^{k,l}|\log\frac{|B_i^{k,l}|}{c}-(c-|B_i^{k,l}|)\log\frac{c-|B_i^{k,l}|}{c}\\
&+L_{\mathbb{N}}(|I(B_i^{k,l})|)+|I(B_i^{k,l})|\log(|B_i^{k,l}|-1)\\
&+|B_i^{k,l}|\log|B_i^{k,l}|\Big\}
\end{aligned}$$

Informally speaking, with regard to our practical score we can consider term (26) to represent $L(dsc_i)$, terms (26) and (26) to correspond with $L_{bid}(dsc_i(\mathbf{M}_i))$, term (26) with $L_{mh}(dsc_i(\mathbf{M}_i))$, and term (26) with $L(\mathbf{M}_i \ominus dsc_i(\mathbf{M}_i))$.

Theorem 26 permits an algorithm based on dynamic programming, since the solution of the first (left-most) c micro bins can be derived from that of the first $l < c$ micro bins. Using dynamic programming, the search for the optimal bin merge is feasible in a polynomial time: For each k such that $1 < k \leq T_i$, we find the optimal bin merge w.r.t. our practical score producing k macro bins on X_i using dynamic programming. When $k = 1$, there is no need to apply the algorithm. Finally, the one yielding the minimum cost across all $k \geq 1$ is selected as the final output. Note that X_i could end up with only one bin. One possible interpretation of this is that X_i contains no significant interaction with other dimensions since, e.g., X_i is a noisy dimension where data values are randomly scattered.

Though dynamic programming is an efficient strategy for traversing an exponential search space, it may require prohibitively long runtime for large data. In addition to this optimal solution, we therefore propose a fast greedy heuristic.

Greedy bin merge strategy IPD$_{gr}$. Our greedy bin merge is as follows. Starting with the T_i micro bins of X_i, in each step, it searches for two bins whose merge minimizes the practical MDL-based score. If this score is less than the current score, i.e., their merge is beneficial, the greedy algorithm merges these two bins and continues. Otherwise, it terminates. To quantify the quality of IPD$_{gr}$ w.r.t. IPD$_{opt}$, we have two performance bounds of IPD$_{gr}$ on X_i as follows.

Theorem 27. *Asymptotically* IPD$_{gr}$ *is a* 2*-approximation algorithm of* IPD$_{opt}$.

Proof. Consider a discretization dsc_i on dimension X_i with k_i macro bins

$\{B_i^1, \ldots, B_i^{k_i}\}$. We have

$$\begin{aligned} L(\mathbf{M}_i, dsc_i) &\geq L_{bid}(dsc_i(\mathbf{M}_i)) + L(\mathbf{M}_i \ominus dsc_i(\mathbf{M}_i)) \\ &\geq \left(\sum_{j=1}^{k_i} L_{\mathbb{N}}(|B_i^j|) + (|B_i^j| + 1) \log \frac{T_i}{|B_i^j|} \right) + \sum_{j=1}^{k_i} |B_i^j| \log |B_i^j| \\ &\geq (T_i + k_i) \log T_i - \sum_{j=1}^{k_i} \log |B_i^j| \\ &\geq T_i \log T_i \quad . \end{aligned}$$

Let $dsc_i^{T_i}$ be the discretization that puts each micro bin into a separate macro bin. We have

$$L(\mathbf{M}_i, dsc_i^{T_i}) = L_{\mathbb{N}}(T_i) + T_i \log c_0 + 2T_i \log T_i \quad .$$

Let dsc_i^{opt} and dsc_i^{gr} be the discretization yielded by IPD$_{opt}$ and IPD$_{gr}$, respectively.

Let dsc_i be a discretization that merges two micro bins with a low interaction distance into the same macro bin and places each of the other micro bins into a separate macro bin. It holds that

$$L(\mathbf{M}_i, dsc_i) = L_{\mathbb{N}}(T_i - 1) + \log(T_i - 1) + (T_i - 1) \log c_0 + 2T_i \log T_i - \log T_i \quad .$$

Thus, $L(\mathbf{M}_i, dsc_i) < L(\mathbf{M}_i, dsc_i^{T_i})$, i.e., merging two consecutive micro bins with a low interaction distance in the first place will yield an encoding cost lower than that of $dsc_i^{T_i}$. Thus, IPD$_{gr}$ will proceed after this step. Hence, $L(\mathbf{M}_i, dsc_i^{gr}) \leq L(\mathbf{M}_i, dsc_i)$. We have $\frac{L(\mathbf{M}_i, dsc_i^{gr})}{L(\mathbf{M}_i, dsc_i^{opt})} \leq \frac{L(\mathbf{M}_i, dsc_i)}{T_i \log T_i}$. This leads to

$$\frac{L(\mathbf{M}_i, dsc_i^{gr})}{L(\mathbf{M}_i, dsc_i^{opt})} \leq \frac{L_{\mathbb{N}}(T_i - 1) + \log(T_i - 1) + (T_i - 1) \log c_0 + 2T_i \log T_i - \log T_i}{T_i \log T_i} \quad . \tag{8.1}$$

Let RHS be the right hand side of (8.1). It holds that

$$\lim_{T_i \to \infty} RHS = 2$$

as $\lim_{T_i \to \infty} \frac{L_{\mathbb{N}}(T_i - 1)}{T_i \log T_i} = 0$ [Grü07]. In other words, as $T_i \to \infty$, $\frac{L(\mathbf{M}_i, dsc_i^{gr})}{L(\mathbf{M}_i, dsc_i^{opt})} \leq 2$. Therefore, asymptotically IPD$_{gr}$ is a 2-approximation algorithm of IPD$_{opt}$. □

Theorem 28. *Let dsc_i^{gr} be the discretization yielded by* IPD$_{gr}$ *on X_i. Further, let dsc_i^1 be the discretization that merges all micro bins of X_i into one single macro bin. Assuming $L(\mathbf{M}_i, dsc_i^{gr}) \leq L(\mathbf{M}_i, dsc_i^1)$, for $\epsilon \in [0, 1]$ such that $(T_i - 1) \cdot \epsilon$ pairs of consecutive micro bins of X_i have low interaction distance, we asymptotically have* IPD$_{gr}$ *as a $(2 - \epsilon)$-approximation algorithm of* IPD$_{opt}$.

Proof. We assume that there are $(T_i - 1)\epsilon$ pairs of consecutive micro bins of X_i that have low interaction distance ($0 \leq \epsilon \leq 1$), i.e., $(T_i - 1)(1 - \epsilon)$ pairs have a large

interaction distance. We have

$$\begin{aligned} L(\mathbf{M}_i, dsc_i^1) &= \log c_0 + L_{\mathbb{N}}(T_i) + L_{\mathbb{N}}\left((T_i-1)(1-\epsilon)\right) + \\ &\quad (T_i-1)(1-\epsilon)\log(T_i-1) + T_i \log T_i \quad . \end{aligned}$$

This means $\frac{L(\mathbf{M}_i, dsc_i^{gr})}{L(\mathbf{M}_i, dsc_i^{opt})} \leq$

$$\frac{\log c_0 + L_{\mathbb{N}}(T_i) + L_{\mathbb{N}}\left((T_i-1)(1-\epsilon)\right) + (T_i-1)(1-\epsilon)\log(T_i-1) + T_i \log T_i}{T_i \log T_i}. \tag{8.3}$$

Let RHS be the right hand side of (8.3). Note that $\lim_{T_i \to \infty} RHS = 2 - \epsilon$. In other words, as $T_i \to \infty$, $\frac{L(\mathbf{M}_i, dsc_i^{gr})}{L(\mathbf{M}_i, dsc_i^{opt})} \leq 2 - \epsilon$. □

As in general ϵ will be larger than zero, the bound in Theorem 28 improves over Theorem 27. For instance, when $\epsilon = 1/3$, IPD$_{gr}$ is a 1.67-approximation algorithm of IPD$_{opt}$. We observe that the assumption made in Theorem 28 holds for all data sets tested in the experiments. In fact, the results show that IPD$_{gr}$ achieves an approximation factor of about 1.1 of IPD$_{opt}$, while being up to an order of magnitude faster. Overall, we find that in practice IPD$_{gr}$ strikes a very good balance between time and quality. Still, we note that both variants are both more efficient and produce higher quality discretizations than existing techniques.

8.6.2. Parameter settings

Setting the number of micro bins. To set T_i, we rely on a recent result by [RRF+11]. They show that to avoid inflated pairwise correlation scores when discretizing data, the number of bins in each dimension must be $\leq n^{1-\delta}$, with n being the number of data points of **DB** and $\delta \in (0, 1)$. Based on this result, and our own preliminary empirical analysis, we use $T_i = n^{0.5}$ in the remainder of this chapter.

Setting a threshold for interaction distances. To use ID in our practical score, i.e., to compute $L_{mh}(dsc_i(\mathbf{M}_i))$, we need to be able to decide which interactions distances are 'large'. This is a difficult problem in general, also for other distance functions. The naïve way is to use a fixed cutoff threshold. However, preliminary analysis showed this does not work well. That is, in practice there is no global threshold suitable for all dimensions and all data sets.

Instead, we propose a data-driven approach: sort the distances between consecutive micro bins in ascending order and pick a threshold equal to a quantile t_q of the distance values. We thus make the threshold dependent on the distance distribution of each dimension. Preliminary experiments showed that the first tertile is a good choice. Of course, one can adjust t_q, e.g., by analyzing the distance values, to reflect the level of detail one is willing to keep. One can just set t_q as we do, and let our discretization methods handle the task of merging bins appropriately. Throughout all experiments in this chapter we will use the first tertile.

8.6.3. Complexity analysis

The computational complexity of IPD consists of three parts 1) pre-sorting the data per dimension, 2) computing interaction distances, and 3) bin merging.

Sorting the data costs $O(n \log n)$ per dimension. For each dimension X_i we compute the distances between all pairs of consecutive micro bins, with an individual cost of $O(\frac{mn^2}{T_i^2})$ (cf., Theorem 25). As there are $T_i - 1$ pairs of micro bins, the cost per dimension is $O(\frac{mn^2}{T_i}) = O(mn^{1.5})$ (cf., Section 8.6.2). Bin merging takes $O(T_i^3) = O(n^{1.5})$ for the dynamic programming method, and $O(T_i^2) = O(n)$ for the greedy method. In sum, we see that for both bin merge strategies the total theoretical complexity of IPD is $O(m^2 n^{1.5})$.

It is interesting to compare this result to existing techniques. When we do so, we find that with regard to size MVD [Bay01], CPD [MPY05], SD [FI93], and CHIM [Ker92] all have a complexity of $O(n^2)$, while UD potentially scales cubicly to n. With regard to dimensionality, CPD requires $O(m^3)$ time for performing PCA, and potentially exponential time with m for mining itemsets inside the bins. Similarly, in the worst case MVD scales exponentially to m due to its use of contrast set mining. UD, SD, and CHIM scale linearly to m, as they do not analyze interactions with the other $(m - 1)$ dimensions. Overall, in terms of worst case complexity, we find that IPD is at least as efficient as its multivariate competitors. However, the empirical results show that in practice IPD is much faster than both its univariate and multivariate competitors.

8.7. Experiments

In our experiments, we study the ability of IPD to maintain multivariate interactions. We test its two variants: optimal IPD (IPD$_{opt}$) using dynamic programming and greedy IPD (IPD$_{gr}$), which employs our greedy bin merge strategy.

All experiments were conducted on Intel i5-2520M machines with 8GB RAM.

8.7.1. Setup

We perform four sets of experiments. We first evaluate, using synthetic data, if our methods preserve known interactions (Sec. 8.7.2). Next, we evaluate on real-world data using three representative use cases for multivariate discretization: pattern-based compression (Sec. 8.7.3), outlier detection (Sec. 8.7.4), and classification (Sec. 8.7.5).

We compare IPD$_{opt}$ and IPD$_{gr}$ against state of the art methods in both supervised and unsupervised discretization. Table 8.1 summarizes their characteristics. UD [KM07] performs unsupervised univariate discretization, CPD [MPY05] is for unsupervised multivariate discretization, and SD [FI93] is for supervised discretization. For each method, we optimize parameter settings according to their respective papers. We create the initial micro bins on the basis of equal-frequency,

	Unsupervised	Multivariate	Interaction Preserving
UD [KM07]	✓	-	-
CPD [MPY05]	✓	✓	*
SD [FI93]	-	-	-
IPD	✓	✓	✓

Table 8.1.: Characteristics of methods. (*) means partially.

Data	N	D	Classes
Climate	35601	251	-
Crime	1993	101	-
Gas	13910	128	7
KDD	311029	41	38
Energy	48501	540	2
Mini	130064	50	2
PAMAP	1686000	42	15
PAMAP2	1662216	51	18
Parkinson	5875	18	-
SatImage	6435	36	6

Table 8.2.: Characteristics of the real data sets.

similar to [MPY05], and hence, allow for fair comparison. For IPD, we always fix t_q to the first tertile.

We experiment on 10 real data sets. We draw six of them from the UCI Machine Learning Repository, the publicly available PAMAP database[1], and two further data sets on Energy and Climate data. The Energy data set contains hourly energy consumption indicators (e.g., water, heating, electricity) of different buildings in KIT university campus, recorded from 2006 to 2011. The Climate data set contains climate data of an office building in Frankfurt, Germany, collected from 2004 to 2012 [WLV+14]. Note that SD requires labels and is hence, inapplicable on Climate, Crime, and Parkinson. We summarize the characteristics of these data sets in Table 8.2.

8.7.2. Preserving interactions

To show that our methods are able to preserve known dimension interactions, we first experiment on synthetic data.

Synthetic Case 1. First, we generate data according to the $R+I+S$ parity problem, which is the continuous version of the parity problem. That is, each data set consists of (a) R dimensions uniformly distributed in the range $[-0.5, 0.5]$, (b)

[1] http://www.pamap.org/demo.html

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8–X_{100}
Ideal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-
IPD$_{opt}$	-0.04	0.03	-0.04	-0.05	0.02	0.04	-0.04	-
IPD$_{gr}$	-0.05	0.05	-0.04	-0.05	0.04	0.05	-0.05	-
UD	-	-	-	-	-	-	-	-
SD	-	-	-	-	-	-	-0.04	-
CPD	0.28	0.24	0.16	-0.21	-0.11	-0.24	0.22	+
	0.41	0.37	0.34	0.39	0.33	0.10	0.36	

Table 8.3.: Preserving interactions on Synthetic Case 1. The ideal outcome: one cut point at 0.0 for dimensions X_1–X_7, no cut points for X_8–X_{100}. (-) means no cut point. (+) means has at least one cut point.

another dimension whose value is a random number drawn from $(0, 0.5]$ if an even number of values of the first R dimensions are positive, and drawn from $[-0.5, 0]$ otherwise, and (c) S irrelevant dimensions. For each R, we create a data set of 10000 points, including 10% noise. We present representative results with $R = 6$ and $S = 93$, i.e., 100 dimensions in total. For SD, we create the class label as follows: If the $(R + 1)$th dimension is positive, the class is 1, and 0 otherwise.

Please note that, in the ideal solution, each of the dimensions X_1 to X_7 only has one cut point at 0.0 (i.e., two bins) while no irrelevant dimension has a cut point. Table 8.3 shows that IPD$_{opt}$ and IPD$_{gr}$ produce the results closest to the ideal. UD is univariate and oblivious to dimension interactions, and hence, here creates only one bin for X_1 to X_7. SD also yields only one bin for X_1 to X_6. This is because it only considers the interaction of each dimension with the class label, while in this case, interactions among multiple dimensions are required to find proper cut points. CPD in turn introduces spurious cut points for all dimensions, including the irrelevant ones.

To assess robustness of ID with regard to high dimensional interactions, we evaluated on data with $R = 60$ and $S = 300$. The result are consistent with those above.

Synthetic Case 2. Next, we generate data according to multivariate histograms. Each data set created has $(R+2R+3R+20R)$ dimensions in the range $[-5.0, 5.0]$. Each of the first R dimensions (R-dimensional interaction) has one cut point at 0.0. Each of the next $2R$ dimensions ($2R$-dimensional interaction) has three cut points at -3.0, 0.0, 3.0. Each of the next $3R$ dimensions ($3R$-dimensional interaction) has five cut points at -3.0, -1.0, 0.0, 1.0, 3.0. The remaining $20R$ dimensions are irrelevant. For each group of relevant dimensions with n bins per dimension, we pick n multivariate cells that do not overlap in any univariate bin. For instance, assuming that $R = 3$, cells $(0, 1, 0)$ and $(1, 0, 1)$ of the first group do not have a common univariate bin. For each cell picked, we assign a multivariate Gaussian distribution with a dimensionally matching mean vector and covariance

	IPD$_{opt}$	**IPD**$_{gr}$	**UD**	**CPD**	**SD**
5-dimensional interaction	✓	✓	-	-	n/a
10-dimensional interaction	✓	✓	-	-	n/a
15-dimensional interaction	✓	✓	-	-	n/a
100 irrelevant dimensions	✓	✓	✓	-	n/a

Table 8.4.: Preserving interactions on Synthetic Case 2. "✓" means the respective method discovers the correct discretization over all dimensions of the given group, and "-" if otherwise. The ideal outcome: "✓" in all groups. SD is not applicable as the task is unsupervised.

matrix. To create a new data point o, from each relevant group we pick a cell (with equal probability) and sample the values of o in the respective dimensions accordingly. In the irrelevant dimensions, the values of o are sampled uniformly randomly from $[-5.0, 5.0]$. To increase complexity we add 10% random data points to the unselected cells. Using our procedure, we ensure each data set to follow a known histogram, i.e., known ground-truth cut points.

For a given data set, a discretization method is considered to produce a correct result for a group iff (a) it produces the correct number of cut points in all member dimensions, and (b) if the group contains relevant dimensions, each cut point is within a distance $\delta = 0.5$ to the correct cut point. Table 8.3 shows the results on a data set with $R = 5$, and containing 10000 points. For brevity, we only show if methods correctly identify cut points for different groups. Only IPD$_{opt}$ and IPD$_{gr}$ produce the correct discretizations for all groups. This implies that IPD, and hence, ID are robust w.r.t. high dimensional interactions.

Overall, the experiments on synthetic data show that our methods successfully identify and preserve synthetic interactions among dimensions.

8.7.3. Compression

Next, we examine whether IPD preserves interactions in real-world data. To this end, we use our ideal score (Sec. 8.5.1) to fairly compare between discretizations. The resolutions per dimension needed in $L(dsc)$ and $L(\mathbf{DB} \ominus dsc(\mathbf{DB}))$ are determined from the data following [KM07].

We instantiate the score with three different compressors. First, we use GZIP, a general purpose compressor. To apply it, we serialize the data per row, per column in the original order of the data. Further, we use KRIMP [VvS11] and COMPREX [ATVF12], two pattern-based methods that compress data using itemsets—which allows these methods to detect and reward multivariate interactions. In addition, COMPREX can exploit correlations by grouping and encoding highly interacting attributes together. For GZIP and COMPREX we use default parameters, for KRIMP we use a minimal support of 10%. Overall, for each compressor, the total encoding cost $L(\mathbf{DB}, dsc)$ will be low only if the interactions are preserved well.

	GZIP		KRIMP		COMPREX	
Data	**IPD**$_{opt}$	**IPD**$_{gr}$	**IPD**$_{opt}$	**IPD**$_{gr}$	**IPD**$_{opt}$	**IPD**$_{gr}$
Climate	19413k	21018k	14092k	15501k	11247k	12597k
Crime	754k	786k	486k	511k	282k	305k
Gas	3867k	4358k	2929k	3134k	2753k	3028k
KDD	3178k	3178k	2838k	2923k	2724k	2751k
Energy	28367k	30197k	22743k	22971k	13246k	13908k
Mini	56033k	58360k	19348k	20509k	9757k	10635k
PAMAP	113062k	118373k	92215k	94982k	80187k	85800k
PAMAP2	132673k	137874k	105967k	114444k	92146k	104125k
Parkinson	294k	328k	210k	228k	169k	186k
SatImage	688k	754k	491k	530k	420k	462k

Table 8.5.: [Lower is better] The total compression costs in bits of IPD$_{opt}$ and IPD$_{gr}$, $L(\mathbf{DB}, dsc)$, using different compressors.

We present the results in Fig. 8.3. The plots show the relative compression rates, with IPD$_{opt}$ as base, per dataset, for each of the considered discretization methods, using respectively GZIP, KRIMP, and COMPREX as compressor. The absolute total compression costs for IPD$_{opt}$ and IPD$_{gr}$ are reported in Table 8.5.

From the figures, we see that IPD$_{opt}$ yields the best results across all data sets and all three compressors. The performance of IPD$_{gr}$ is very close to that of IPD$_{opt}$. Further, our methods provide about 100% saving in bits compared to SD, and even over 200% compared to UD and CPD. This implies that our methods preserve dimension interactions well, aiding interaction-aware methods like KRIMP and COMPREX to detect patterns over multiple attributes, leading to lower compression costs.

Concerning the competition, UD did not finish within 6 days for the PAMAP data sets. Although multivariate in nature, CPD did not obtain very good scores, which indicates it either does not maintain all strong interactions, or that spurious interactions are introduced. Overall, the results show that IPD$_{opt}$ and IPD$_{gr}$ best preserve complex patterns hidden in different real-world data sets.

We also use pattern-based compression to study the sensitivity of our methods to input parameters. Recalling from Section 8.6.2, our methods receive two such parameters: (a) the number of micro bins T and (b) the interaction distance cutoff t_q. Setting T on the other hand depends on the setting of δ, i.e., we study the parameterization of δ instead. To study the performance of our methods w.r.t. δ, we fix $t_q = 1/3$ and choose the compression cost at $\delta = 0.5$ as the base. To study the performance of our methods w.r.t. t_q, we fix $\delta = 0.5$ and choose the compression cost at $t_q = 1/3$ as the base. We display the results on the Climate data set with COMPREX as the compressor for illustration purposes. The results of IPD$_{opt}$ are in Figure 8.4 and those of IPD$_{gr}$ are in Figure 8.5. We can see that the performance of our methods (in terms of relative compression cost) improves as δ decreases.

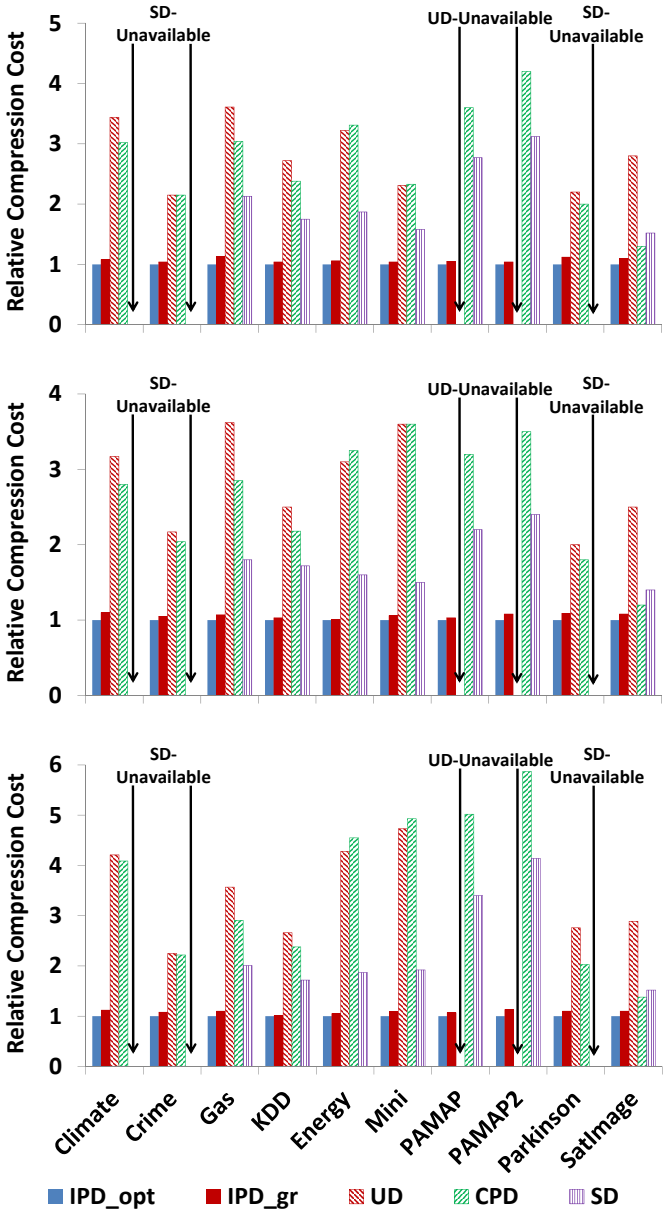
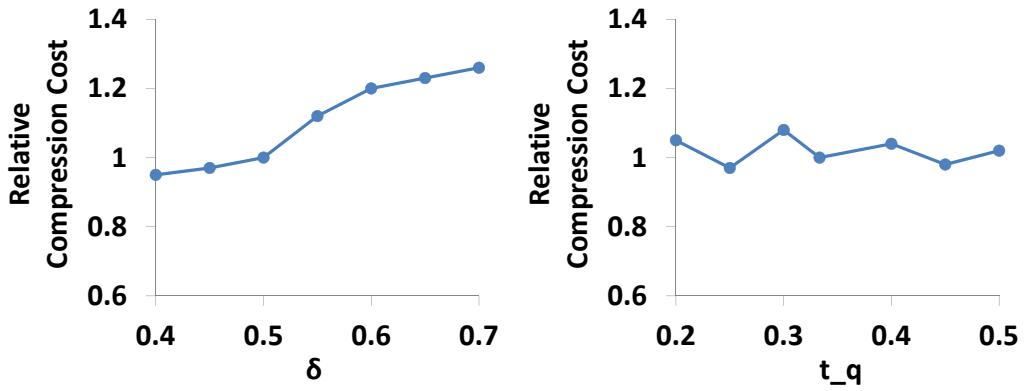


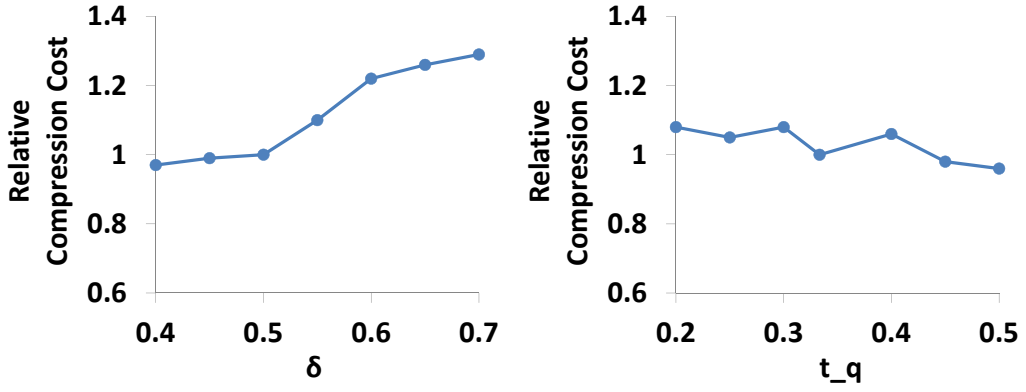
Figure 8.3.: [Lower is better] Relative total compression costs for all data sets, using GZIP (top), KRIMP (middle), and COMPREX (bottom) as compressor. The compression costs of IPD_{opt} are the bases. SD is not applicable on unlabeled data. UD did not finish within 6 days on the PAMAP data.

However, the change when δ varies between 0.4 and 0.5 is small. On the other hand, the runtime becomes higher with lower values of δ (see Section 8.6.3). Thus, we use $\delta = 0.5$ in the remainder of this chapter. Setting t_q on the other hand seems to be harder. Yet, we use $t_q = 1/3$ as this value yields good performance compared to other values. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable



(a) Compression cost vs. δ (b) Compression cost vs. t_q

Figure 8.4.: [Lower is better] Relative compression cost of IPD_{opt} vs. δ (i.e., T) and t_q on the Climate data set.



(a) Compression cost vs. δ (b) Compression cost vs. t_q

Figure 8.5.: [Lower is better] Relative compression cost of IPD_{gr} vs. δ (i.e., T) and t_q on the Climate data set.

parameterization might be required.

8.7.4. Outlier detection

The previous experiments showed our methods preserve interactions. Next, we investigate how well outliers can still be identified in the discretized data. If a discretization is too detailed, all values will be unique, and hence, all records are 'outliers'. If the discretization is too coarse, no outliers will be detectable.

To detect outliers we use COMPREX with the different discretization methods as pre-processing step. As argued by [ATVF12], patterns that compress the majority of the data well define its norm, and hence, data points that cannot be compressed well can be safely regarded as an outlier. To evaluate performance we use labeled data: one class as the 'normal' objects and another as the 'outliers'. The evaluation metric is the average precision, computed as the area under the precision-recall

Data	IPD$_{opt}$	IPD$_{gr}$	UD	CPD	SD	LOF
Gas	**0.74**	0.72	0.36	0.42	0.48	0.64
KDD	**0.54**	0.53	0.14	0.19	0.03	0.44
Energy	**0.70**	0.69	0.33	0.45	0.45	0.36
Mini	**0.79**	**0.79**	0.30	0.53	0.51	0.60
PAMAP	**0.82**	0.79	*	0.38	0.24	0.54
PAMAP2	**0.84**	0.82	*	0.41	0.27	0.53
SatImage	**0.41**	**0.41**	0.21	0.28	0.15	0.33
Average	**0.69**	0.68	0.27	0.38	0.30	0.49

Table 8.6.: [Higher is better] Average precision (area under precision-recall curve) for outlier mining. LOF ran on the original continuous data. Highest values are in **bold**. (*) means the result is unavailable due to excessive runtime.

curve. It is the average of the precision values obtained across recall levels. As standard baseline, we run LOF [BKRTN00] on the original data. Climate, Crime, and Parkinson are unlabeled, and hence, not applicable in this experiment.

We present the results in Table 8.6. Both IPD$_{opt}$ and IPD$_{gr}$ obtain very high average precision, outperforming the competition with a broad margin. In fact, a Friedman test [Dem06] at $\alpha = 0.05$ shows that the observed differences are significant. A Nemenyi test in the post-hoc analysis learns us that: (a) IPD$_{opt}$ significantly outperforms UD, CPD, and SD, and (b) IPD$_{gr}$ significantly outperforms UD and SD. Using a Wilcoxon signed rank test with $\alpha = 0.05$ to compare IPD$_{gr}$ and CPD, we find IPD$_{gr}$ to be significantly better.

Interestingly, IPD$_{opt}$ and IPD$_{gr}$ beat LOF with a wide margin—significant under a Wilcoxon signed rank test—despite that discretized data contains inherently less information. By weeding out irrelevant associations, IPD provides COMPREX the chance to outperform LOF.

We are aware that outlier detection methods exist that may outperform LOF. However, our goal here is not to push the envelope in outlier detection, but to compare the quality of the discovered discretizations. Moreover, LOF is often used as baseline.

8.7.5. Classification

Next, we evaluate the methods in the context of a supervised task: classification. To evaluate performance, we train Random Forests [Bre01] on the discretized data, and consider accuracy as the performance metric. In addition, as a baseline, we also report its performance on the continuous data (RF). We use the implementation in the WEKA toolkit with default parameters. All results are obtained by performing 10-fold cross validation. For the unsupervised methods IPD$_{opt}$, IPD$_{gr}$, UD, and CPD, we do not show the class labels during discretization. As above,

Data	IPD$_{opt}$	IPD$_{gr}$	UD	CPD	SD	RF
Gas	**.99 ± .00**	**.99 ± .00**	.56 ± .01	.71 ± .01	.98 ± .00	**.99 ± .00**
KDD	**.98 ± .00**	**.98 ± .00**	.58 ± .00	.70 ± .00	**.98 ± .00**	**.98 ± .00**
Energy	**.97 ± .01**	.96 ± .01	.48 ± .02	.68 ± .02	.94 ± .01	.93 ± .01
Mini	**.92 ± .00**	.91 ± .00	.75 ± .00	.72 ± .00	.89 ± .00	.91 ± .00
PAMAP	**.92 ± .01**	.90 ± .01	*	.71 ± .01	.87 ± .01	-
PAMAP2	**.98 ± .01**	**.98 ± .01**	*	.66 ± .00	.86 ± .01	-
SatImage	**.89 ± .01**	.87 ± .01	.82 ± .01	.81 ± .01	.86 ± .01	**.89 ± .01**
Average	**0.95**	0.94	0.64	0.71	0.91	0.94

Table 8.7.: [Higher is better] Classification accuracy. RF ran on the original continuous data. Highest values are in **bold**. (*) means the result is unavailable due to excessive runtime. (-) means the result is unavailable due to memory overflow.

the unlabeled data sets are not applicable. For PAMAP, RF did not finish due to memory overflows.

We present the results in Table 8.7. We report both mean and standard deviation over the cross-validation folds. Considering the results, we see that the supervised methods SD and plain RF obtain much higher accuracies than UD and CPD. Interestingly, and somewhat surprisingly, both IPD$_{opt}$ and IPD$_{gr}$ consistently outperform SD, and perform as least as good as plain RF—even though they were unaware of the class labels. A possible explanation is that RF and SD only maintain interactions between individual dimensions and the class label, and by making decisions locally may misalign bins of interacting dimensions. Our methods, however, are able to detect and maintain the multivariate structure of the data, which if it correlates with the class label, will aid classification.

By applying a Friedman test at $\alpha = 0.05$, we find the observed differences between the discretization methods to be significant. A Nemenyi test in the post-hoc analysis shows IPD$_{opt}$ and IPD$_{gr}$ perform significantly better than UD and CPD. A Wilcoxon signed rank test between IPD$_{opt}$ and SD, shows IPD$_{opt}$ to be significantly better. Repeating this test between IPD$_{gr}$ and SD, we find IPD$_{gr}$ to be significantly better. The difference between IPD$_{opt}$, resp. IPD$_{gr}$, and RF is not significant.

Note that we are aware of other modern classifiers (e.g., SVMs), which may outperform RF. However, for us, state of the art classification is not the goal, but simply a means for evaluating how well discretization techniques maintain interactions.

8.7.6. Runtime

Last, we evaluate runtime. In Figure 8.6, we show the relative runtimes of all methods on all data sets considered. We pick the runtimes of IPD$_{opt}$ as the bases. The wall-clock runtimes of IPD$_{opt}$ and IPD$_{gr}$ are in Table 8.8. The results show

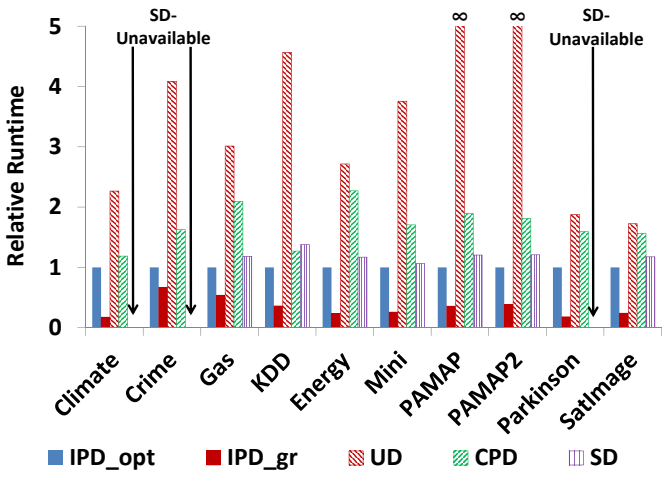


Figure 8.6.: [Lower is better] Relative runtimes of all methods compared to IPD_{opt}. SD is only applicable on labeled data.

that in practice, both our methods are faster than the competition, with IPD_{gr} by far the fastest method overall. UD did not finish within 6 days on the PAMAP data.

8.8. Discussion

The experiments show that IPD provides very high quality discrete data, maintaining the key structures and interactions of the original continuous data. We found that it consistently outperforms its competitors in maintaining interactions and patterns, allowing for the identification of outliers, as well as in classification accuracy. Moreover, the runtime experiment shows that it is faster than both state of the art univariate and multivariate competitors. The improved performance of IPD compared to existing techniques can be traced back to its three main components, (a) our interaction distance ID, (b) our MDL-based balancing of preserving dimension interactions and the information within dimensions, and (c) our efficient bin merge strategies. In sum, IPD provides a powerful approach for unsupervised multivariate discretization.

This is not to say that the problem of interaction-preserving multivariate discretization is now solved. IPD involves some design choices and there is room for alternative solutions and improvements, which are beyond the scope of this article. For instance, we form micro bins on the basis of equal-frequency. It is interesting to see whether more advanced techniques for forming micro bins, such as UD, may lead to better overall interaction preservation.

MDL here guides us to very good discretizations. However, constructing an encoding involves choices that determine which structure is rewarded. We formalized a general MDL framework for interaction-preserving discretization; if one is willing to make explicit assumptions on the distribution of the data or structure of the

Data	n	m	**IPD**$_{opt}$	**IPD**$_{gr}$
Climate	35601	251	61242	10708
Crime	1993	101	47	32
Gas	13910	128	1267	682
KDD	311029	41	10200	3674
Energy	48501	540	107233	25532
Mini	130064	50	15208	3923
PAMAP	1686000	42	217835	77798
PAMAP2	1662216	51	284778	109530
Parkinson	5875	18	32	6
SatImage	6435	36	77	18

Table 8.8.: [Lower is better] The single-threaded, wall-clock runtimes in seconds of IPD$_{opt}$ and IPD$_{gr}$.

ideal discretization, other encodings or model selection techniques can be considered. Also, for a more specialized solution, one could optimize towards one specific compressor.

We do note that our MDL framework can be instantiated by any interaction distance. In this chapter, we use ID as an instantiation. Depending on the task at hand, other interaction distances may be preferred, such as the Kullback-Leibler divergence or the Jensen-Shannon divergence [CT06]. As long as one is able to find a reliable way to compute such distances, these can be used within our framework. In our context, we find that ID yields a good combination of theoretical correctness and foundations, simplicity, and ease of computation.

In this chapter we focused on discretization quality. As future work, we consider looking into highly scalable approximations to IPD. Key ideas include that ID can be sped up by considering not all dimensions, but only those within the subspace most strongly interacting with the current dimension. That is, we can factor the full space into smaller subspaces that can then be considered independently. This reduces the complexity for ID, and allows for efficient implementation by parallelization. Additionally, as IPD discretizes dimensions independently of the others this can be trivially parallelized, as can the computation of ID between consecutive micro bins.

8.9. Conclusion

In this chapter, we proposed IPD, an information-theoretic method for unsupervised discretization of multivariate data, that specifically focuses on the preservation of interactions. That is, for each dimension we consider the distribution of the data over all others. In particular, we only merge consecutive regions if their multivariate distributions are statistically similar and the merge reduces the MDL encoding cost. Empirical evaluation on both synthetic and real-world data sets

shows that IPD obtains high quality discrete data that maintains the key interactions of the original, while outperforming existing methods in a range of knowledge discovery tasks.

IPD and *ID* have high potential impact in any setting where discretization is required, be it explicitly such as for methods that only consider discrete data, or implicitly wherever cut points need to be decided. As examples, we plan to investigate the application and embedding of our methods for pattern mining, subgroup discovery, and selectivity estimation. Moreover, we are investigating highly scalable approximations that would allow considering very large databases.

9. Information-Theoretic Causality Analysis

This chapter is based on our work in progress [NV]:

H. V. Nguyen and J. Vreeken, *Information-theoretic causal discovery*, in progress.

In all of the previous chapters, we discuss mainly correlation analysis. In this chapter, we go one step forward and study efficient causal inference in high dimensional data. It is well-known in the statistics community that *"correlation does not imply causation"* [Ald95]. Thus, to perform causality analysis, we need a whole different treatment [Pea09, SGS00]. This motivates us to look at causal inference in more details here. Yet, as it will turn out, our solution is based on that of CMI++. This demonstrates another benefit of correlation-aware discretization.

The setting of our study is as follows. Under the assumption of causal sufficiency (no hidden confounders), given two multivariate random variables X and Y with some correlation relationship and with the same number of observations, we aim at efficiently inferring their causal direction. Solving this problem in fact is identified as one of the key steps towards generating causal graphs from empirical data [Pea09].

To solve the above problem, we propose a new principle for causal inference that is based on Kolmogorov complexity and which makes use of the algorithmic Markov condition. We show that our principle generalizes various existing methods that rely on the principle of plausible Markov kernels. As Kolmogorov complexity is not computable, we present NOVUM, an efficient non-parametric implementation of our principle based on cumulative and Shannon entropy. We show our method is applicable to mixed typed data, as well as how to derive a scalable causal discovery framework. One important feature of our work is that we do not restrict the type of correlation between random variables. Extensive experiments on both synthetic and real-world data show NOVUM to yield both high accuracy and high efficiency on both deterministic and noisy data.

9.1. Introduction

Considerable research effort on causal analysis has been devoted to infer whether X causes Y or Y causes X from only the joint observations of (X, Y) under the assumption of causal sufficiency (no hidden confounders) [SJS06, SJS08, ZH09, PJS10, JS10, MSJ+10, JHS10, ZJZ11, JMZ+12, CZC13]. Recently, there is a focus on inferring the causal direction between multivariate random variables X and Y [JHS10, ZJZ11, CZC13]. Existing methods consider linear correlation and invertible functional correlation. Correlations in real-world data, however, range from non-linear to complex non-functional [RRF+11]. Furthermore, existing work is not designed to handle mixed typed data. That is, X and Y may not contain both numerical and categorical components. Lastly, existing techniques for handling non-linear functional correlations are computationally expensive, and do not scale to large data. In this work, we propose a framework and an instantiation that alleviates each of these issues.

More in particular, we propose a new principle for causal inference based on Kolmogorov complexity, and which embraces the principle of plausible Markov kernels [SJS06, SJS08, JS10, MSJ+10, ZJZ11].

Intuitively, the principle of plausible Markov kernels, which we name the Markov kernel principle for short, states that the factorization of the joint distribution $p(cause, effect)$ into $p(cause)p(effect \mid cause)$ typically yields a model of lower total *complexity* than the factorization into $p(effect)p(cause \mid effect)$ [ZJZ11]. A straightforward implementation of this principle is to take only the complexity of $p(.)$ into account. However, as the (conditional) pdfs are not available at hand, they need to be estimated from empirical data. This estimation exercise in turn induces an estimation of the joint probability $p(cause, effect)$. Deriving our inference principle from Kolmogorov complexity [LV93, AV09], we argue that this complexity of also needs to be included in the total complexity.

Based on this observation, we define causal indicators that consider the complexity of marginal and conditional pdfs *and* estimated joint distributions. On a conceptual level, we instantiate the notion of complexity using Kolmogorov complexity [LV93]. We show that our principle yields a generalization of various existing approaches [SJS06, SJS08, JS10, MSJ+10].

As a practical instantiation, we introduce NOVUM, an efficient method that implements our principle using entropy divergence. We use this method to infer the causal direction between multivariate random variables X and Y, without having to make any assumption on their distribution nor their correlation—be it linear or non-linear, functional or non-functional. In addition, similar to most of the existing work on causal inference by the Markov kernel principle [SJS06, SJS08, JS10], we do not have to make an explicit assumption on the presence of noise. This does not mean we claim that we are impervious to stochastic relationships. However, our experiments reveal that our method is robust to noise, showing very good performance on a wide range of noisy real-world and synthetic data sets.

Lastly, we explain how to extend our method to handle mixed typed data, as well as how to derive a scalable causal discovery framework from it. We show that the latter yields a more general and a more efficient alternative to the well-known constraint-based causal discovery [Coo97].

Overall, our contributions include:

- We propose a new principle for causal inference using Kolmogorov complexity.
- We introduce an efficient implementation of our principle to infer the causal direction between two multivariate real-valued random variables X and Y. Our method does not require any assumption on the relationship between X and Y.
- We extend our method to handle categorical random variables, and hence, enable causal analysis to mixed typed data sets.
- We present an automatic and scalable framework for mining cause-effect univariate pairs together with their causal directions in large and high dimensional data. We show that this framework is suitable for data mining applications where it is not mandatory to find the complete causal graph of the data [SBMU00].

In this chapter, we make the assumption of causal sufficiency, i.e. there are no hidden confounders. In addition, we pursue causal discovery by purely analyzing non-experimental data.

9.2. Preliminaries

We consider a D-dimensional random variables $X = \{X_1, \ldots, X_D\}$ and k-dimensional random variables $Y = \{Y_1, \ldots, Y_k\}$. We write $\mathbf{X}$ to denote the data we have for X. If we have collected N observations for X, $\mathbf{X}$ then represents the $N \times D$ data matrix of X. Similarly, we use $\mathbf{X}_i$ for the data of X_i, and slightly abusing notation, analogue we say $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_D\}$.

We write $p(X)$ for the probability distribution of X. If X contains real-valued (resp. integer-valued) components only, we call X a real-valued (resp. integer-valued) multivariate random variable, and $p(X)$ the probability density function (pdf) of X. If its components are all categorical we say X is a categorical multivariate random variable, and call $p(X)$ the probability mass function (pmf) of X. We write $p(x)$ as a short form for $p(X = x)$. We define $p(Y)$ similarly.

If X is numerical, we assume that the domain of $X_i \in X$ is $[min(X_i), max(X_i)]$.

Considering a univariate numerical random variable X, we write the cumulative distribution function (cdf) of X as $P(X)$, with $P(x)$ as the short form of $P(X \leq x)$.

9.3. Principle

Consider two multivariate random variables $X = \{X_1, \ldots, X_D\}$ and $Y = \{Y_1, \ldots, Y_k\}$ for which we know there exists a correlation relationship—identified by either domain experts or an appropriate test. Assume that we have N observations for each variable. Our task is then to infer the causal relationship between X and Y given the data $\mathbf{X}$ and $\mathbf{Y}$ collected for these variables. We assume causal sufficiency, and hence, the decision comes down to which of $X \to Y$ and $Y \to X$ is more plausible.

Our inference principle is a generalization of the Markov kernel principle [SJS06, SJS08, JS10, MSJ+10]. Intuitively, the Markov kernel principle states that the factorization of the joint distribution $p(cause, effect)$ into $p(cause)p(effect \mid cause)$ typically yields a model of lower total complexity than the factorization into $p(effect)p(cause \mid effect)$. A well-founded way to implement the notion of complexity is to use Kolmogorov complexity. Following the Markov kernel principle, one would compute the total complexity of two factorizations of $p(X, Y)$ as

$$K(p(X)) + K(p(Y \mid X)) \tag{9.1}$$

and

$$K(p(Y)) + K(p(X \mid Y)) \quad , \tag{9.2}$$

respectively, where $K(p(\cdot))$ is the Kolmogorov complexity of $p(\cdot)$. Then, one infers the causal direction corresponding to the smaller complexity.

Setting aside the non-computable property of Kolmogorov complexity, Eqs. (9.1) and (9.2) ignore that the pdfs and conditional pdfs over X and Y are generally not available: We only have empirical (observational) data $\mathbf{X}$ and $\mathbf{Y}$. As a result, $p(X)$ resp. $p(Y)$ first needs to be estimated. This exercise yields, in turn, an estimation of the joint probability $p(X, Y)$; see [SJS06] for an example. We write $\widehat{p}_{X \to Y}(X, Y)$ as the joint distribution induced by the factorization of $p(X, Y)$ into $p(X)$ and $p(Y \mid X)$. We define $\widehat{p}_{Y \to X}(X, Y)$ similarly.

In the large data limit (the amount of empirical data approaches infinity), $p(.)$ can be estimated with arbitrarily high accuracy [Sil86, TP95]. Hence, the estimated joint probability functions induced by different factorizations can be assumed to all fit the data approximately equally well. In practice, however, the sample size of the data is never infinite, and generic (conditional) pdfs tend to be hard to estimate with limited observed data. Consequently, the complexity of the estimated joint probability $\widehat{p}_{X \to Y}(X, Y)$ will become non-negligible, and hence, $K(\widehat{p}_{X \to Y}(X, Y))$ will need to be included in the inference rule to allow a fair call on the causal direction. This observation is in line with the discussion in [JS10] (Section III.C, Equation (31)). Consequently, for the hypothesis $X \to Y$, we propose its causal indicator $\Delta_{X \to Y}$ as below:

$$\Delta_{X \to Y} = K(p(X)) + K(p(Y \mid X)) + K(\widehat{p}_{X \to Y}(X, Y))$$

We define $\Delta_{Y\to X}$ similarly. If $\Delta_{X\to Y} < \Delta_{Y\to X}$, we infer that $X \to Y$ is more plausible. Otherwise, if $\Delta_{Y\to X} < \Delta_{X\to Y}$, we infer that $Y \to X$.

In our causal indicators, for instance, $\Delta_{X\to Y}$, the sum $K(p(X)) + K(p(Y \mid X))$ can be perceived as the cost of the model $K(M)$. Thus, we denote this sum as $K(M(\Delta_{X\to Y}))$. On the other hand, $K(\widehat{p}_{X\to Y}(X,Y))$ can be perceived as the cost of the data given the model $K(\mathbf{D} \mid M)$. Thus, we write this term as $K(\mathbf{D} \mid M(\Delta_{X\to Y}))$. We use this interpretation to show that our principle is a generalization of various existing approaches. For a more complete discussion of related work we refer the reader to Section 9.8.

First, we consider [SJS06], a method for causal inference based on the Markov kernel principle. In short, this method takes into account only $K(\mathbf{D} \mid M(\Delta_{X\to Y}))$ to infer causal direction. They instantiate K by the maximum log-likelihood function, inferring the probabilistic model by Maximum Entropy [Jay82]. Analogously, [MSJ$^+$10] first estimate $p(X)$ and $p(Y \mid X)$, giving preference to functions and distributions of low complexity. Inferring the causal direction then becomes selecting the joint distribution with larger likelihood. Though the latter implicitly favour simple models, neither of these proposals explicitly consider the complexity $K(M(\Delta_{X\to Y}))$.

In contrast, [SJS08] consider only $K(M(\Delta_{X\to Y}))$, instantiating K by means of seminorms in a reproducing kernel Hilbert space. [JS10] also mainly consider $K(M(\Delta_{X\to Y}))$, though they remark that the cost of the data given the model may become significant under limited observed data. As their study was largely theoretical, they did not provide a specific instantiation for K.

Having formulated our principle and discussed its key relations to existing work, we now proceed introducing our instantiation of our inference principle.

9.4. Implementing our Causal Inference Rules for Numerical Data

Kolmogorov complexity provides a rather nice framework for reasoning about data, yet its practical application is limited as it is not computable [LV93]. The Minimum Description Length (MDL) principle [Ris78, Grü07] provides a practical variant. Given a data set $\mathbf{D}$ and model class $\mathcal{M}$, MDL identifies the best model $M \in \mathcal{M}$, the best available description of the data, as the model that minimizes $L(M)+L(\mathbf{D} \mid M)$ where $L(M)$ is the length, in bits, of the description of the model, and $L(\mathbf{D} \mid M)$ is the length, in bits, for describing the data given the model. This is also known as two-part, or *crude*[1] MDL, and as Adriaans and Vitányi recently showed has a very tight relation to two-part Kolmogorov complexity [AV09].

Our goal is to obtain computable causal indicators. Thus, we will implement them using MDL. In particular, we will consider the sum of complexities over $p(X)$,

[1] As opposed to *refined* MDL, where model and data are encoded together. While this avoids possibly arbitrary choices in the definition of the encoding, refined MDL is only computable in special cases.

$p(Y \mid X)$, and $\widehat{p}_{X\to Y}(X,Y)$, respectively denoted by $L(p(X))$, $L(p(Y \mid X))$, and $L(\widehat{p}_{X\to Y}(X,Y))$. We will implement these scores non-parametrically.

As such, over the course of this section we will implement the following rule

$$\Delta_{X\to Y} = \underbrace{\frac{L(p(X)) + L(p(Y \mid X))}{L_U(p(X)) + L_U(p(Y))}}_{K(M(\Delta_{X\to Y}))} + \underbrace{\frac{L(\widehat{p}_{X\to Y}(X,Y))}{L_U(\widehat{p}_{X\to Y}(X)) + L_U(\widehat{p}_{X\to Y}(Y))}}_{K(\mathbf{D}|M(\Delta_{X\to Y}))} \tag{9.3}$$

and analogue for $\Delta_{Y\to X}$. Below we will explain Equation (9.3) in details. For now, let us simply assume that all terms are non-negative and that both the left-hand and right-hand terms are scaled into $[0,1]$, i.e. that $L(p(X)) + L(p(Y \mid X)) \leq L_U(p(X)) + L_U(p(Y))$ and $L(\widehat{p}_{X\to Y}(X,Y)) \leq L_U(\widehat{p}_{X\to Y}(X)) + L_U(\widehat{p}_{X\to Y}(Y))$.

First, the left hand side measures the relative complexity of the distributions $p(X)$ and $p(Y \mid X)$ with regard to the maximally complex distributions over the same variables, while the right hand side measures the relative complexity of the estimated joint distribution $\widehat{p}_{X\to Y}(X,Y)$, with regard to the maximally complex distribution of the same parameters. Ignoring the normalization, the score can be interpreted as the average number of bits we need per observation to describe it under the assumption $X \to Y$. In fact, a natural interpretation of the rule is that it measures the divergence between $X \to Y$ and $X \perp\!\!\!\perp Y$. The lower the score, the stronger the detected causality; the closer to its maximum, 2, the weaker the causal relationship. We normalize to ensure the left and right hand terms measure complexity in comparable domains.

For a discrete distribution (pmf), we can measures its description complexity by means of Shannon entropy [CT06]. For a pdf, we can instead use cumulative entropy (see Chapter 3). In short, cumulative entropy captures the information content, i.e., complexity, of a probability distribution. However, different from Shannon entropy, it works with (conditional) cdfs. Overall, to compute $L(p(\cdot))$ when $p(\cdot)$ is a (conditional) pdf, we will use cumulative entropy. Otherwise, we will use Shannon entropy. Our approach of using entropy to measure the complexity of a distribution indeed coincides with one of the information-geometric causal inference methods in [JMZ+12].

We consider *relative* complexities in Equation (9.3)—as opposed to summing the plain complexity terms—to ensure the two sides are scaled to the same domain. To this end we normalize with the complexities of the independence model. For generality, we normalize using upper bounds $L_U(\cdot)$ of $L(\cdot)$. A straightforward—yet suboptimal—option is to set $L_U(\cdot) = L(\cdot)$. We will formalize L and L_U using cumulative entropy. For readability, we will only consider real-valued variables X and Y in this section. We will give the details for how to compute our scores in Section 9.5, and will expand our formalization to mixed typed data in Section 9.6.

9.4.1. Implementing complexity measure L

As part of the causal indicator $\Delta_{X\to Y}$ in Equation (9.3), we need to be able to compute the complexity of $p(X)$, $p(Y \mid X)$, and $\widehat{p}_{X\to Y}(X,Y)$.

We note that $p(X)$ and $p(Y \mid X)$ are respectively pdf and conditional pdf of numerical multivariate random variables. Thus, we measure $L(p(X))$ and $L(p(Y \mid X))$ by cumulative entropy. The complexity of $\widehat{p}_{X \to Y}(X, Y)$, $L(\widehat{p}_{X \to Y}(X, Y))$, in turn depends on the specific estimation scheme used. We therefore will first describe the desiderata of $L(\widehat{p}_{X \to Y}(X, Y))$ and then boil down to a specific estimation scheme.

First, we consider the unconditioned case $L(p(X))$. As X is multivariate, and cumulative entropy is only defined for single univariate random variables (see Chapter 3), we calculate the complexity of $p(X)$ by applying a chain rule. That is, we consider the factorization of $p(X)$ into $p(X_1)$, $p(X_2 \mid X_1)$, ..., $p(X_D \mid X_1, \ldots, X_{D-1})$. We quantify the complexity of a $p(X_i \mid \cdot)$ with $i \in [1, D]$ by the cumulative entropy $h(\mathbf{X}_i \mid \cdot)$.

However, as there are $D!$ possible factorizations of $p(X)$, the question hence is: which one to choose? Our goal is to build a general formulation that is independent of any specific factorization. Thus, following the Minimum Description Length principle, we define the complexity of $p(X)$ as the *minimum* complexity over all factorizations. In particular, we write σ_X to denote a permutation of the member components of X, and have

$$\begin{aligned} &L(p(X)) \\ &= \min_{\sigma_X} h(\mathbf{X}_{\sigma_X(1)}) + h(\mathbf{X}_{\sigma_X(2)} \mid \mathbf{X}_{\sigma_X(1)}) + \cdots + h(\mathbf{X}_{\sigma_X(D)} \mid \mathbf{X}_{\sigma_X(1)}, \ldots, \mathbf{X}_{\sigma_X(D-1)}) \quad . \end{aligned}$$

Likewise, we quantify the complexity of $p(Y \mid X)$ through its factorizations of the form $p(Y_1 \mid X)$, $p(Y_2 \mid X, Y_1)$, ..., $p(Y_k \mid X, Y_1, \ldots, Y_{k-1})$, again using cumulative entropy to measure the complexity of each component. To maintain generality, we also define the complexity of $p(Y \mid X)$ as the *minimum* complexity over all of its factorizations. In other words, we have

$$\begin{aligned} &L(p(Y \mid X)) \\ &= \min_{\sigma_Y} h(\mathbf{Y}_{\sigma_Y(1)} \mid \mathbf{X}) + h(\mathbf{Y}_{\sigma_Y(2)} \mid \mathbf{X}, \mathbf{Y}_{\sigma_Y(1)}) + \cdots + h(\mathbf{Y}_{\sigma_Y(k)} \mid \mathbf{X}, \mathbf{Y}_{\sigma_Y(1)}, \ldots, \mathbf{Y}_{\sigma_Y(k-1)}) \end{aligned}$$

where σ_Y is a permutation of the member components of Y.

9.4.1.1. Measuring $L(p(Y \mid X))$

To be able to use these measures for inference and to better understand our normalization scheme, it is important to know that $L(p(Y \mid X))$ is at most $L(p(Y))$.

Lemma 16. *It holds that* $L(p(Y \mid X)) \leq L(p(Y))$.

Proof. Let σ_Y and σ'_Y be the permutations of Y that yield $L(p(Y))$ and $L(p(Y \mid X))$,

respectively. We have:

$$
\begin{aligned}
&h(\mathbf{Y}_{\sigma'_Y(1)} \mid \mathbf{X}) + h(\mathbf{Y}_{\sigma'_Y(2)} \mid \mathbf{X}, \mathbf{Y}_{\sigma'_Y(1)}) + \cdots + h(\mathbf{Y}_{\sigma'_Y(k)} \mid \mathbf{X}, \mathbf{Y}_{\sigma'_Y(1)}, \ldots, \mathbf{Y}_{\sigma'_Y(k-1)}) \\
&\leq h(\mathbf{Y}_{\sigma_Y(1)} \mid \mathbf{X}) + h(\mathbf{Y}_{\sigma_Y(2)} \mid \mathbf{X}, \mathbf{Y}_{\sigma_Y(1)}) + \cdots + h(\mathbf{Y}_{\sigma_Y(k)} \mid \mathbf{X}, \mathbf{Y}_{\sigma_Y(1)}, \ldots, \mathbf{Y}_{\sigma_Y(k-1)}) \\
&\leq h(\mathbf{Y}_{\sigma_Y(1)}) + h(\mathbf{Y}_{\sigma_Y(2)} \mid \mathbf{Y}_{\sigma_Y(1)}) + \cdots + h(\mathbf{Y}_{\sigma_Y(k)} \mid \mathbf{Y}_{\sigma_Y(1)}, \ldots, \mathbf{Y}_{\sigma_Y(k-1)}) \quad .
\end{aligned}
$$

Thus, $L(p(Y \mid X)) \leq L(p(Y))$. □

Based on Lemma 16, we derive that $L(p(X))+L(p(Y \mid X)) \leq L(p(X))+L(p(Y))$—which, under the assumption that $p(Y \mid X)$ is known, allows inference of causal direction by information-geometry [JMZ+12]. As identified above, we work under the assumption that $p(Y \mid X)$ is unavailable and needs to be estimated. Next, we implement $L(\widehat{p}_{X\to Y}(X,Y))$.

9.4.1.2. Measuring $L(\widehat{p}_{X\to Y}(X,Y))$

As aforementioned, the complexity of $\widehat{p}_{X\to Y}(X,Y)$ depends on the specific estimation scheme used. In general, to comply with the property of L for real-valued distributions, we enforce that $L(\widehat{p}_{X\to Y}(X,Y)) \leq L(\widehat{p}_{X\to Y}(X))+L(\widehat{p}_{X\to Y}(Y))$ where $\widehat{p}_{X\to Y}(X)$ and $\widehat{p}_{X\to Y}(Y)$ are resp. the marginals of $\widehat{p}_{X\to Y}(X,Y)$ on X and Y.

While there are a variety of relevant estimation schemes, in this chapter, for practical reasons we consider the case when $\widehat{p}_{X\to Y}(X,Y)$ is discrete. This can happen when we use discretization for density estimation, which is in fact the type of estimation we employ in this work (we postpone the details to Section 9.5).

Under our choice, it is natural that we use Shannon entropy to compute $L(\widehat{p}_{X\to Y}(X,Y))$. Assume that $\widehat{\mathbf{X}}$ and $\widehat{\mathbf{Y}}$ are resp. the discrete version of $\mathbf{X}$ and $\mathbf{Y}$ induced by the underlying estimation scheme. We have $L(\widehat{p}_{X\to Y}(X,Y)) = H(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ where H denotes Shannon entropy. Note that we adopt the same convention as for cumulative entropy: $H(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ here stands for the joint Shannon entropy of *discretized* version of X and Y computed over data $\widehat{\mathbf{X}}$ and $\widehat{\mathbf{Y}}$. Another justification for the use of Shannon entropy is that $H(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}}) \leq H(\widehat{\mathbf{X}}) + H(\widehat{\mathbf{Y}})$, which is in line with our convention in Section 9.4.1.1.

With the complexity of $p(X)$, $p(Y \mid X)$, and $\widehat{p}_{X\to Y}(X,Y)$, we could instantiate $\Delta_{X\to Y}$. However, we note that $L(M(\Delta_{X\to Y})) = L(p(X)) + L(p(Y \mid X))$ and $L(\mathbf{D} \mid M(\Delta_{X\to Y})) = L(\widehat{p}_{X\to Y}(X,Y))$ may be on different scales—for instance, due to the use of different complexity measures, e.g. cumulative entropy for pdfs and Shannon entropy for pmfs. The issue does not occur for Equation (9.3) since we always use Kolmogorov complexity there. To handle it, we use appropriate normalization schemes for $L(M(\Delta_{X\to Y}))$ and $L(\mathbf{D} \mid M(\Delta_{X\to Y}))$, respectively, which we define next.

9.4.2. Implementing the normalization score L_U

A key aspect of our inference rule is the normalization, as it brings $L(M(\Delta_{X\to Y}))$ and $L(\mathbf{D} \mid M(\Delta_{X\to Y}))$ to the same scale. To do so we have to define L_U, for which

there are several options. A straightforward solution comes from the fact that $L(p(X))+L(p(Y \mid X)) \leq L(p(X))+L(p(Y))$ and $L(\widehat{p}_{X\to Y}(X,Y)) \leq L(\widehat{p}_{X\to Y}(X))+L(\widehat{p}_{X\to Y}(Y))$. That is, one could set $L_U = L$. Although this can work when both X and Y are multivariate, we will point out in Section 9.5 that its discriminative power degenerates when either X or Y is univariate. We hence define L_U by a more refined scheme as follows.

We observe that $L(p(X)) \leq \sum_{i=1}^{D} h(X_i)$. Moreover, on a given interval $[a,b] \subset \mathbb{R}$, we know that the uniform distribution has the largest cumulative entropy [Rao05, CL09, Liu07], with a value of $\frac{b-a}{4}$—which gives us a natural upper bound that we use to define $L_U(p(X))$ with $X = \{X_1, \ldots, X_D\}$, as $L_U(p(X)) = \sum_{i=1}^{D} \frac{\max(X_i)-\min(X_i)}{4}$.

Likewise, given a fixed alphabet, the uniform distribution has the largest Shannon entropy. Hence, given $X = \{X_1, \ldots, X_D\}$ with all discrete components, $L(\widehat{p}_{X\to Y}(X)) \leq \sum_{i=1}^{D} \log(|X_i|)$ where $|X_i|$ denotes the number of bins of discretized X_i. So we define $L_U(\widehat{p}_{X\to Y}(X)) = \sum_{i=1}^{D} \log(|X_i|)$.

Note that if all components of discretized X and Y each has only one bin, our convention is that $\frac{0}{0} = 0$.

9.4.3. Putting it together

Let us recall the definition of our causal indicators, Equation (9.3), as

$$\Delta_{X\to Y} = \underbrace{\frac{L(p(X)) + L(p(Y \mid X))}{L_U(p(X)) + L_U(p(Y))}}_{K(\mathbf{X}')+K(\mathbf{Y}'|\mathbf{X})} + \underbrace{\frac{L(\widehat{p}_{X\to Y}(X,Y))}{L_U(\widehat{p}_{X\to Y}(X)) + L_U(\widehat{p}_{X\to Y}(Y))}}_{K(\mathbf{X},\mathbf{Y}|\mathbf{X}',\mathbf{Y}')}$$

and the analogue definition for $\Delta_{Y\to X}$. As in Section 9.3, if $\Delta_{X\to Y} < \Delta_{Y\to X}$, we infer that $X \to Y$ is more plausible. Otherwise, if $\Delta_{Y\to X} < \Delta_{X\to Y}$, we infer that $Y \to X$.

One can see that our causal indicators are reminiscent of mutual information, and hence, Kullback-Leibler (KL) divergence. In particular, KL divergence quantifies the difference of a joint distribution to the product of its marginals, i.e. comparing the joint distribution against its independence model. Likewise, the *normalized* left-hand and right-hand sides can be considered as to compare the complexity of the respective joint distribution against the complexities of its marginals. In other words, our indicators can be perceived as a variant of KL divergence defined over entropy—we hence refer to this as entropy-divergence, which to the best of our knowledge is a novel measure. We postpone an in-depth study of its properties to future work.

One can see that our causal indicators do not make any assumption on the type of correlation between X and Y, be it linear or non-linear, functional or non-functional. Besides, similar to most of the existing work on causal inference by the Markov kernel principle [SJS06, SJS08, JS10], they do not make explicit assumption on the presence of noise. This does not mean we are, or claim to be impervious to noise. Our experiments on synthetic and real-world data reveal, however, that our method does perform very well in both deterministic and noisy settings.

Another advantage of our causal indicators is that they are also applicable when $D = k = 1$, i.e. X and Y are univariate. We use the univariate case to demonstrate an additional benefit of our indicators. In particular, following Theorem 1, Chapter 3, $h(\mathbf{Y} \mid \mathbf{X}) = 0$ iff Y is a function of X, i.e. there exists a function $f : dom(X) \to dom(Y)$ such that $Y = f(X)$. If f is not invertible, then $h(\mathbf{X} \mid \mathbf{Y})$ is likely positive. In other words, the asymmetric property of cumulative entropy seems to permit causal inference: If $X \to Y$ then $h(\mathbf{Y} \mid \mathbf{X}) = 0$ while $h(\mathbf{X} \mid \mathbf{Y}) > 0$. However, this observation alone is not sufficient for causal inference. In particular, when f is invertible, we have that $h(\mathbf{Y} \mid \mathbf{X}) = h(\mathbf{X} \mid \mathbf{Y}) = 0$. So by conditional cumulative entropy alone we are unable to discover the causal direction for deterministic relations. However, by additionally considering $h(\mathbf{X})$, $L(\widehat{p}_{X\to Y}(X, Y))$, and $h(\mathbf{Y})$, $L(\widehat{p}_{Y\to X}(Y, X))$, we can break the tie.

9.5. Implementation Details for Numerical Data

We now present the implementation details of our causal indicators $\Delta_{X\to Y}$ and $\Delta_{Y\to X}$. In particular, we explain our design choice for (a) efficiently computing $\Delta_{X\to Y}$ and $\Delta_{Y\to X}$, and (b) estimating conditional cumulative entropy. We also discuss the time complexity of our implementation.

9.5.1. Efficient computation of causal indicators

As our aim is to enable causal discovery for large-scale data, we present a method to efficiently compute our causal formulations. In particular, to compute $\Delta_{X\to Y}$ efficiently essentially boils down to computing $L(p(X))$ and $L(p(Y \mid X))$ efficiently. We discuss the case of $L(p(X))$ first.

To compute $L(p(X))$ optimally, we need to exhausitively consider all $D!$ permutations, which is infeasible for high dimensional data. Instead, we propose a greedy solution. That is, we first pick X'_1 with minimal cumulative entropy, i.e., $h(\mathbf{X}'_1)$ is minimal. Then, we pick X'_2 such that $h(\mathbf{X}'_2 \mid \mathbf{X}'_1)$ is minimal. We proceed till all dimensions have been selected. We consider the permutation σ^*_X where dimensions are picked to be the approximate optimal permutation of X.

Likewise, to compute $L(p(Y \mid X))$, we pick Y'_1 such that $h(\mathbf{Y}'_1 \mid \mathbf{X})$ is minimal. Then, we pick Y'_2 such that $h(\mathbf{Y}'_2 \mid \mathbf{X}, \mathbf{Y}'_1)$ is minimal. We proceed till all dimensions of Y have been selected. We consider the permutation σ^*_Y where dimensions are picked using the above scheme as the approximate optimal permutation of Y.

9.5.2. Estimating conditional cumulative entropy

We tie the estimation of conditional cumulative entropy together with our algorithm for selecting the permutations of X and Y. For illustration, we first consider computing $L(p(X))$. In our algorithm, we estimate conditional cumulative entropy by discretization. That is, after selecting X'_1, we calculate $h(\mathbf{E} \mid \mathbf{X}'_1)$ for each dimension E left by searching for the discretization of X'_1 such that $h(\mathbf{E} \mid \mathbf{X}'_1)$ is minimal; we select dimension E with the minimal score.

For each subsequent step, when computing conditional cumulative entropy, we only discretize the dimension picked in the previous step, i.e. we do not re-discretize any earlier chosen dimensions. As shown in Chapter 3, specifically in CMI++, the discretization at each step can be done efficiently by dynamic programming.

The computation of conditional entropy in $L(p(Y \mid X))$ is done analogously. One small difference is that when searching for Y'_1, we seek for the discretization of X'_D concurrently. Thus, after processing all dimensions of Y, all but Y'_k are discretized. One can see that our computation of $L(p(X))$ and $L(p(Y \mid X))$ somehow violates the causal inference principle which requires $p(X)$ and $p(Y \mid X)$ to be algorithmically independent. However, as mentioned in [JS10], this is unavoidable when the amount of observed data is limited such that the estimated (conditional) pdfs do not coincide with the respective true distributions.

As an alternative solution to our computation of conditional entropy, one could use kernel methods, such as the one in [SP12]. However, besides not having to choose a kernel, our strategy of minimizing the conditional entropy is also in line with our greedy algorithm for selecting good dimension permutations.

9.5.3. Computing $L(\widehat{p}_{X\to Y}(X,Y))$: Revisited

For brevity, we discuss only $L(\widehat{p}_{X\to Y}(X,Y))$, as we define $L(\widehat{p}_{Y\to X}(Y,X))$ analogue. Recall that Y'_k is the last dimension of Y picked and *not* discretized. Thus, $\widehat{p}_{X\to Y}(X,Y)$ is made up by the discretized X, the discretized $Y \setminus \{Y'_k\}$, and Y'_k.

In line with our strategy to search for a minimal value of $\Delta_{X\to Y}$, it makes lots of sense that we search for the discretization of Y'_k that minimizes Shannon entropy of the joint distribution $\widehat{p}_{X\to Y}(X,Y)$. This essentially means we use only one bin for Y'_k since $H(\widehat{\mathbf{X}},\widehat{\mathbf{Y}}) \geq H(\widehat{\mathbf{X}},\widehat{\mathbf{Y}} \setminus \{\widehat{\mathbf{Y}}'_k\})$. Thus, we have $L(\widehat{p}_{X\to Y}(X,Y)) = H(\widehat{\mathbf{X}},\widehat{\mathbf{Y}} \setminus \{\widehat{\mathbf{Y}}'_k\})$. Please note here that it is not important whether $\widehat{p}_{X\to Y}(X,Y)$ is a good estimation of $p(X,Y)$. That is, this issue plays no role in our causal inference: What we care about is the complexity of $\widehat{p}_{X\to Y}(X,Y)$.

9.5.4. Normalizing, L_U: Revisited

In Section 9.4.3, we assert that setting $L_U = L$ suffers from degeneration in discriminative power when either X or Y is univariate. We now explain our claim.

For illustration purposes, let us assume that Y is univariate. Based on Equation (9.3), we then have

$$\Delta_{X \to Y} = \frac{L(p(X)) + L(p(Y \mid X))}{L(p(X)) + L(p(Y))} + 1$$

and

$$\Delta_{Y \to X} = \frac{L(p(Y)) + L(p(X \mid Y))}{L(p(Y)) + L(p(X))} + \frac{H(\widehat{\mathbf{Y}}, \widehat{\mathbf{X}} \setminus \{\widehat{\mathbf{X}}'_D\})}{H(\widehat{\mathbf{Y}}) + H(\widehat{\mathbf{X}} \setminus \{\widehat{\mathbf{X}}'_D\})} \quad .$$

It is easy to see that for $\Delta_{X \to Y}$ the normalized right-hand side term has highest possible value (i.e. 1). This, however, is not an accurate reflection of the complexity of $\widehat{p}_{X \to Y}(X, Y)$, but is rather an artifact of Y being univariate. In fact, if X would also be univariate, the right hand side of $\Delta_{Y \to X}$ would also reduce to 1, and we would be left with inferring the causal direction solely based on the left hand side terms of our causal indicator, which is not comprehensive following our analysis in Section 9.3. Furthermore, a preliminary empirical study verified this simple normalization scheme does not perform well in practice.

In Section 9.4.3, we justify the advantage of our causal indicators using the case when X and Y are both univariate. We now reuse this case to further justify our computation method. In particular, assume that X and Y are statistically independent—possibly due to a false positive correlation test. Following Theorem 2, Chapter 3, we have $h(\mathbf{Y} \mid \mathbf{X}) = h(\mathbf{Y})$ and $h(\mathbf{X} \mid \mathbf{Y}) = h(\mathbf{X})$. Moreover, under independence we expect to end up with one bin only when discretizing X to compute $h(\mathbf{Y} \mid \mathbf{X})$. And, likewise for Y when discretizing to compute $h(\mathbf{X} \mid \mathbf{Y})$. As a result, when computing the causal direction between $\mathbf{X}$ and $\mathbf{Y}$ we have

$$\Delta_{X \to Y} = \Delta_{Y \to X} = \frac{h(\mathbf{X}) + h(\mathbf{Y})}{\frac{\max(X) - \min(X)}{4} + \frac{\max(Y) - \min(Y)}{4}} \quad .$$

Thus, when $X \perp\!\!\!\perp Y$ we make no inference on the causal direction, which is correct. Not only does our method yield accurate inference in this case, our extensive experiments show that it also performs very well in complex settings, including noisy ones.

9.5.5. Complexity analysis

Here we fix $\epsilon = 0.3$ and $c = 10$ following our preliminary analysis. The cost of discretizing a dimension then is $O(N)$. The overall complexity of computing $\Delta_{X \to Y}$ and $\Delta_{Y \to X}$ is therefore $O((D^2 + k^2) \cdot N)$.

9.6. Extending the Framework

In the following, we extend our method for causal inference. First, we consider the case of mixed typed data. Second, we sketch a framework for scalable inference in large data.

9.6.1. Numerical and categorical variables

Our formulations extend straightforwardly to the case where either X or Y is categorical. We first focus on the case of inferring which of $X \to Y$ and $Y \to X$ is more plausible, provided that X is an D-dimensional real-valued random variable and Y is a k-dimensional categorical random variable. The intuition here is as before: We use cumulative entropy for real-valued data and Shannon entropy for categorical data. In particular, we have:

- $L(p(X))$ as above,
- $L(p(Y \mid X)) = H(\mathbf{Y} \mid \mathbf{X})$,
- $L(\widehat{p}_{X \to Y}(X, Y)) = H(\widehat{\mathbf{X}}, \mathbf{Y})$,
- $L(p(Y)) = H(\mathbf{Y})$,
- $L(p(X \mid Y))$ as above, and
- $L(\widehat{p}_{Y \to X}(X, Y)) = H(\widehat{\mathbf{X}} \setminus \{\widehat{\mathbf{X}}'_D\}, \mathbf{Y})$.

We compute $L(p(Y \mid X)) = H(\mathbf{Y} \mid \mathbf{X})$ by extending Sections 9.5.1 and 9.5.2 to Shannon entropy. This extension is straightforward as our reasoning there still holds when h is replaced by H. Our causal inference otherwise proceeds similar to when both X and Y are real-valued.

Note that by this formalization in practice we restrict ourselves to the case where either X or Y is categorical; when both are categorical, due to the symmetric nature of Shannon entropy, we would lose all signals for correct inference. This could be resolved using a non-symmetric definition of entropy for categorical data, or by instantiating the complexity measure using a pattern-based MDL score [VvS11, MVT12]. We postpone these extensions to future work.

9.6.2. Mixed data-type variables

Next, we consider the case where X and Y both contain a mix of numerical and categorical components. We write X^r and X^c as the sets of real-valued and categorical components of X, respectively. We denote Y^r and Y^c similarly. We redefine the followings terms as follows.

For $L(p(X))$, the complexity of $p(X)$, again our goal is to minimize it. Thus, we have $L(p(X)) = \min\{L(p(X^c)) + L(p(X^r \mid X^c)), L(p(X^r)) + L(p(X^c \mid X^r))\}$. On the other hand, for the complexity of $p(Y \mid X)$, also with the goal to minimize it, we have $L(p(Y \mid X)) = \min\{L(p(Y^c \mid X)) + L(p(Y^r \mid X, Y^c)), L(p(Y^r \mid X)) + L(p(Y^c \mid X, Y^r))\}$. The complexity of $\widehat{p}_{X \to Y}(X, Y)$ will depend on whether Y^c or Y^r comes first. If Y^c comes first, we have $L(\widehat{p}_{X \to Y}(X, Y)) = H(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}} \setminus \{\widehat{\mathbf{Y}}'_k\})$. Otherwise, we have $L(\widehat{p}_{X \to Y}(X, Y)) = H(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$.

Analogue to the case where both X and Y are real-valued, $\widehat{\mathbf{Y}}'_k$ here refers to the real-valued dimension in Y picked last and *not* discretized during the calculation of $L(p(Y \mid X))$. To calculate $\Delta_{Y \to X}$, we define $L(p(Y))$, $L(p(X \mid Y))$,

and $L(\widehat{p}_{Y\to X}(X,Y))$ analogue to above. Regarding existing work, to the best of our knowledge, only the method of [SJS08] is applicable to mixed data types—however, due to high computational cost, in practice it restricts itself to binary variables as opposed to categorical variables in general. We do not impose such a restriction.

9.6.3. Towards a scalable causal discovery framework

Given a data set $\mathbf{D}$ with N observations and D dimensions $X_1, \ldots, X_D$. We aim at efficiently detecting all cause-effect univariate pairs in $\mathbf{D}$, especially when N is large.

This problem has been studied in [Coo97, SBMU00]. Relying on the fact that data mining is concerned with discovering interesting knowledge in the data, be it complete or incomplete, they argued and demonstrated through empirical studies the benefits of efficient mining of cause-effect pairs, i.e. incomplete causal discovery, in large data.

Following their thesis, we propose to plug our method into frameworks for scalable correlated subspace mining, such as [NMB13]. Using such a method, we can efficiently discover pairs of correlated dimensions in a multivariate data set $\mathbf{D}$. Then, we use our causal indicators to decide the causal direction for each pair.

Compared to constraint-based causal discovery [Coo97], we have the advantage that we do not need to process three dimensions in order to find pairwise causal relationships. Further, we can avoid disambiguation in deciding if $X_i \to X_j$ or $X_j \to X_i$ since our formulation exactly resolves such situations.

In fact, the information-geometric approach of [JMZ+12] can be a more efficient alternative to our method in handling pairwise univariate causal inference. However, we allow for causal inference in mixed typed data, and hence, open opportunities for analyzing large-scale mixed typed data sets.

9.7. Experiments

We assess our method, NOVUM, which stands for entropy divergence-based causal inference on multivariate and mixed typed data,[2] under three different scenarios:

- causal inference for X and Y where at least one is multivariate,
- causal inference for X and Y where both are univariate,
- causal discovery in large and high dimensional data sets.

For comparison, we include LTR [JHS10], KTR [CZC13], GPI [MSJ+10], and IGCI [JMZ+12]. LTR and KTR are state of the art for causal inference for multivariate pairs, while GPI and IGCI are state of the art for univariate pairs. Table 9.1 summarizes their characteristics. We use both synthetic and real-world data containing noise. All experiments were conducted on Intel i5-2520M machines with 8GB RAM. We implemented NOVUM in Java.

[2] NOVUM is also Latin for new fact.

Method	Multivariate	Univariate	Mixed typed data
NOVUM	✓	✓	✓
LTR [JHS10]	✓	–	–
KTR [CZC13]	✓	–	–
GPI [MSJ+10]	–	✓	–
IGCI [JMZ+12]	–	✓	–

Table 9.1.: Characteristics of methods. (✓) means the method possesses the respective property, (–) means it does not.

9.7.1. Causal inference for multivariate pairs: Synthetic data

To understand the performance of NOVUM under different settings, we generate synthetic data sets simulating complex non-deterministic causal relations. To this end, we generate multivariate $X_{D\times 1} = \mathbf{A}_{D\times D} \times Z_{D\times 1}$ where $z_i \sim \textit{Gaussian}(0,1)$ and $a_{ij} \sim \textit{Uniform}[0,1]$. Then we generate multivariate $Y_{N\times 1}$ with $y_i = f(u_i) + e_i$ where $\mathbf{U}_{k\times 1} = \mathbf{B}_{k\times D} \times X_{D\times 1}$ with $b_{ij} \sim \textit{Uniform}[0, 0.5]$, f is a function describing the relationship between X (through $\mathbf{U}$) and Y, and $e_i \sim \textit{Gaussian}(0,\sigma)$ with σ being a free parameter. We use σ to control the level of noise: If $\sigma = 0$, we have deterministic relationships; larger values of σ correspond to larger noise levels. We choose three instantiations for f:

- $f_1(x) = \tanh(2x) + \tanh(3x+1) + \tanh(4x+2)$
- $f_2(x) = \sin(2x) + \sin(3x+1)$
- $f_3(x) = \sin(2x) + \sin(3x+1) + \frac{1}{3}(\tanh(2x) + \tanh(3x+1) + \tanh(4x+2))$

We choose these functions as all are non-linear, complex, and non-invertible.

We study NOVUM on three aspects: (a) accuracy against dimensionality, (b) accuracy against noise, and (c) runtime against data size and dimensionality. For comparison, we include LTR [JHS10] and KTR [CZC13], two causal inference methods for the multivariate setting. LTR assumes the correlation between X and Y to be linear. KTR relaxes this requirement but explicitly requires the relationship to be deterministic (no noise), functional, and invertible.

Per experiment, we generate 100 data sets per function, and infer the causal direction using each of the above methods. We report the accuracy, the relative number of correct inferences over all data sets.

For accuracy against dimensionality, we set $D = k$ and vary their values from 5 to 120. We fix $N = 1000$ and $\sigma = 0.5$. The results are depicted in Figures 9.1(a), 9.1(b), and 9.1(c). Looking at these, we see that NOVUM performs very well: it yields the highest inference accuracy across different numbers of dimensions and across different functions of varying complexity. Further, its performance is stable with respect to dimensionality and types of relationship between X and Y.

For accuracy against noise, analogue to the study of [JHS10], we vary σ from 0 (deterministic) to 2. We fix $N = 1000$ and $D = k = 5$. Further, for brevity we only discuss the results on f_1. From Figure 9.1(d), we can see that NOVUM is very robust to noise, outperforming both LTR and KTR. The results show NOVUM as a promising solution for causal inference in noisy settings.

With regard to scalability, we evaluate NOVUM in two scenarios. First, we fix $\sigma = 0.5$ and $D = k = 5$, and vary N from 1000 to 15000. Second, we fix $N = 1000$ and $\sigma = 0.5$, and vary $D = k$ from 5 to 120. Based on the results, depicted in Figures 9.1(e) and 9.1(f), we find that NOVUM scales linearly to data size and quadratically to dimensionality. This agrees with our analysis in Section 9.5.5. We observe that NOVUM scales better than KTR but worse than LTR. Taking into account accuracy, we find that NOVUM yields a good balance between quality and efficiency. As it scales linearly to data size it is applicable to large data sets, which we will further study in Section 9.7.4.

We now assess the parameterization of NOVUM. In particular, to study its sensitivity to ϵ, we fix $c = 10$. On the other hand, to study its sensitivity to c, we fix $\epsilon = 0.3$. For illustration, we display the results on synthetic data sets with $N = 1000$, $D = k = 5$, and $\sigma = 0.5$ generated using the f_1 function. Regarding ϵ, we see that the performance of NOVUM increases as this parameter increases. Yet, with $\epsilon \geq 0.3$, the performance is stable. Thus, we use $\epsilon = 0.3$. Regarding c, we see that NOVUM achieves the best performance at $c = 10$ and $c = 11$. Recalling smaller values of c result in better runtime, we set $c = 10$ in the remainder of this chapter. We note that these settings are suggestive to the experiments performed in this chapter only. For other scenarios, further search of a suitable parameterization might be required.

9.7.2. Causal inference for multivariate pairs: Real-world data

We examine if NOVUM is able to infer the causal direction between X and Y where at least one of them is multivariate. In order to do so, we use benchmark real-world data containing pairs of variables for which the causal direction is known. To cover different settings, we include real-valued pairs as well as pairs of mixed data types.

First, we use data on climate forecast [Zsc13]. In this case, both X and Y contain four components

$$\{\textit{air temperature, pressure at surface, sea level pressure, relative humidity}\} \quad .$$

These components were measured at the same location grid at different times. $\mathbf{X}$ contains 10266 observations on day 50 of year 2000. $\mathbf{Y}$ contains respectively 10266 observations on day 51 of year 2000. The ground truth is assumed to be $X \to Y$.

Second, we test using data on the relationship between ozone concentration and different meteorological parameters [Zsc13]. The data contains 989 observations and was collected in Neckarsulm and Heilbronn, Germany. We define

$X = \{$*ozone concentration*$\}$ and $Y = \{$*wind speed, global radiation, temperature*$\}$. Following [JHS10], the production of ozone is impacted by wind, sun radiation, and air temperature, the ground truth causal direction is $Y \to X$. We note that in this experiment, X is univariate while Y is multivariate.

Third, we use data on car efficiency in terms of miles per gallon, of 392 records [Zsc13]. We define $X = \{$*displacement, horsepower, weight*$\}$ and $Y = \{mpg, acceleration\}$. It is natural to assume that $X \to Y$.

The fourth data set contains daily mean values of ozone values and sun radiation in the last 83 days of 2009 at 16 different places in Switzerland [Zsc13]. This data set is rather small: it has only 72 records. We set X to contain 16 measures of ozone values and Y to contain 16 measures of sun radiation. As in the second data set, the causal direction is assumed to be $Y \to X$. Using this data, we want to check if NOVUM is able to perform accurate causal inference even when the number of observation is relatively small compared to the numbers of dimensions.

The fifth data set is from the medical domain and has 120 records [Zsc13]. X contains 6 symptoms of patients and Y contains diagnosis results of two diseases. We note that Y is categorical. It is expected that $X \to Y$.

The sixth data set is about weather and air pollution in Chemnitz, Germany [JHS10]. It contains 1440 observations. X has 3 dimensions on the wind direction and air temperature. Y has 6 dimensions on air quality indices, e.g. sulfur dioxid and dust concentration. The ground truth specifies that $X \to Y$.

The seventh data set is on Energy consumption on a university campus [NMB13]. X contains three components: the location of each building (categorical), the total area of the building (real-valued), and its number of staff members (integer-valued). Y contains four energy consumption indicators of buildings: electricity, gas, heating, water. We expect the causal direction to be $X \to Y$.

Finally, we utilize two additional real-world data sets from [JHS10], one on precipitation and the other on stock indices.

All in all, we summarize the results in Table 9.2. Since LTR and KTR require both X and Y to be multivariate, they are inapplicable to the ozone concentration problem with univariate X. They are also inapplicable to the symptoms & diseases, and energy consumption problems as they do not handle categorical data. Note that on the symptoms & diseases data set, [Vre15] considers the categorical columns as numerical columns while we here stick to the original data types of all columns. We will explain the relationship between NOVUM and the method of [Vre15] in more details in Section 9.8. Looking at the results, one can see that the accuracy of LTR and KTR are both 50%. NOVUM in turn achieves an accuracy of 77.78%. Further, NOVUM is applicable to all data sets, be the underlying problem purely multivariate or a mix of univariate and multivariate, be the data real-valued or a mix of different data types.

Regarding the two incorrect causal inference results of NOVUM, we see that the absolute difference between two respective causal indicators are small (0.04 for

Data	N	D	k	type	NOVUM	LTR	KTR
Climate forecast	10266	4	4	real-valued	✓	✓	–
Ozone concentration	989	1	3	real-valued	✓	(n/a)	(n/a)
Car efficiency	392	3	2	real-valued	✓	–	✓
Ozone & radiation	72	16	16	real-valued	✓	–	✓
Symptoms & diseases	120	6	2	mixed	✓	(n/a)	(n/a)
Weather & air pollution	1440	3	6	real-valued	✓	✓	–
Energy	48501	3	4	mixed	✓	(n/a)	(n/a)
Precipitation	4748	3	12	real-valued	–	✓	–
Stock indices	2394	4	3	real-valued	–	–	✓

Table 9.2.: Results of real-world cause-effect multivariate pairs with known ground truth. (✓) means the respective method infers the causal direction correctly, and (–) otherwise. (n/a) means the respective method is inapplicable to the given pair.

precipitation data and 0.03 for stock) compared to that of other data sets (minimum 0.11). We do not have yet an explanation for the case of precipitation data. For the stock indices case, [JHS10] assume that daily Asian stock indices cause daily European stock indices due to the difference in time zones. Though stock indices around the world tend to be related, each stock index is impacted by many external factors, including local events and situations, such as unemployment rates. Hence, in this data the assumption of causal sufficiency may be (strongly) violated—which could be an explanation why NOVUM does not reach the desired inference.

We note that these data sets are very noisy, especially Energy. Given the results, we hence conclude that NOVUM is applicable in non-deterministic settings.

9.7.3. Causal inference for univariate pairs: Real-world data

In this experiment, we apply NOVUM to real-world cause-effect univariate pairs with known ground truth [Zsc13]. We compare to GPI [MSJ$^+$10] and IGCI [JMZ$^+$12], two recent causal inference methods which are applicable to univariate pairs. We pick 78 pairs from various domains. All of them are more or less noisy, i.e. the relationship between each pair is non-deterministic. The result shows that NOVUM achieves an accuracy of 74.36%. The accuracy of IGCI is 70.51% while that of GPI is 62.82%. Thus, we can see that NOVUM also achieves very promising accuracy on causal inference for univariate pairs.

Besides univariate pairs of numerical domains, we also consider pairs of mixed types. To this end, we test NOVUM on three real-world pairs extracted from a Census data set [BL13]: (a) *age* (integer) and *marital status* (categorial), (b) *sex* (binary) and *income* (real-valued), and (c) *education level* (categorical) and *income* (real-valued). Common sense says *age* $\rightarrow$ *marital status*, *education level* $\rightarrow$ *income*, while various studies have shown, sadly, *sex* $\rightarrow$ *income*. Using NOVUM, we find:

- *age* $\rightarrow$ *marital status* is correctly inferred, as $\Delta_{\textit{age}\rightarrow\textit{marital status}} = 0.90$, while $\Delta_{\textit{marital status}\rightarrow\textit{age}} = 1.36$

- *education level* $\rightarrow$ *income* is recovered, with $\Delta_{\textit{education level}\rightarrow\textit{income}} = 1.13$, while $\Delta_{\textit{income}\rightarrow\textit{education level}} = 1.62$

- *sex* $\rightarrow$ *income* is confirmed, as $\Delta_{\textit{sex}\rightarrow\textit{income}} = 0.55$, while $\Delta_{\textit{income}\rightarrow\textit{sex}} = 0.96$.

In other words, NOVUM correctly infers the causal directions in all three cases. We note that GPI and IGCI are inapplicable to the three pairs as they only work with real-valued pairs. The method of [SJS08] in theory can handle mixed typed data. However, in practice, it has to convert categorical data to binary one to avoid high computational costs. While this conversion is feasible in some cases, e.g. *age* $\rightarrow$ *marital status* and *sex* $\rightarrow$ *income*, it may not be straightforward in other cases, e.g. *education level* $\rightarrow$ *income*. That is, different conversions might produce different causal directions. In conclusion, NOVUM achieves high accuracy on both real-valued univariate pairs as well as mixed typed univariate pairs.

9.7.4. Causal discovery in real-world data

Next, we evaluate whether NOVUM can be used to discover novel causal relations in non-benchmark data. To this end, we apply NOVUM on the climate data set (see Section 4.9.3, Chapter 4).

To discover pairs of correlated dimensions, we first apply 4S (cf., Chapter 4) to assess pairwise correlations. Next, we apply NOVUM on the 886 pairs discovered to decide the causal directions between the dimensions. For comparison, we also experiment with IGCI [JMZ$^+$12], which specializes in inferring the causal direction for univariate X and Y; and LCD, a constraint-based causal discovery technique [Coo97]. In short, LCD examines each triple of univariate X, Y, and W. If it holds that (a) W has no cause, (b) X and W are correlated, (c) X and Y are correlated, and (d) W and Y are independent given X, then LCD infers that $X \rightarrow Y$. For the conditional independence test of LCD, we use the nonparametric method in [SP12].

Overall, the results show that NOVUM and IGCI take about the same amount of time, less than one hour, while LCD needs about two hours, to process all candidate pairs. In terms of quality, all three methods detected sensible causal relationships. Nevertheless, the decisions of NOVUM and IGCI tend to be of higher quality since they are able to resolve the disambiguation of $X \rightarrow Y$ and $Y \rightarrow X$. We now discuss some findings of NOVUM that were *not* discovered by both IGCI and LCD.

First, for the temperature fed into (called *temp_in*) and the temperature produced by the heating system (called *temp_out*), NOVUM yields $\Delta_{\textit{temp\_in}\rightarrow\textit{temp\_out}} = 1.14$ and $\Delta_{\textit{temp\_out}\rightarrow\textit{temp\_in}} = 1.46$. Hence, we infer that *temp_in* $\rightarrow$ *temp_out*, which corresponds to common sense.

Second, NOVUM detects that the amount of air coming into the ventilator (called *air_in*) causes the amount of air coming out of the ventilator (called *air_out*). In particular, $\Delta_{air\_in \to air\_out} = 1.24$ and $\Delta_{air\_out \to air\_in} = 1.59$. This again is intuitively correct.

Likewise, NOVUM concludes that the amount of electricity consumed by the ventilator (called *elec*) causes the amount of air coming out of the ventilator, with $\Delta_{elec \to air\_out} = 0.46$ and $\Delta_{air\_out \to elec} = 0.61$.

9.8. Related Work

Traditional causal inference methods [Pea09, SGS00] are based on the causal Markov condition which states that every variable is conditionally independent of its non-effects, given its direct causes. Thus, to detect causal relationships, these methods rely on conditional independence tests, and hence, require at least three observed random variables; they are not designed to infer the causal direction for just two observed random variables X and Y. Recent developments in this approach, such as [TGS09, ZPJS11], focus mainly on proposing new conditional independence tests, e.g. the kernel-based conditional independence test (KCI).

The algorithmic information-theoretic approach to causal inference [LD06, LS10, JS10, LJ13] formulates the problem of inferring causal direction on the foundations of algorithmic information theory, or, Kolmogorov complexity, and postulates that $X \to Y$ is only acceptable if $p(X)$ and $p(Y \mid X)$ are algorithmically independent, i.e. the shortest description of $p(X, Y)$ is given by separate descriptions of $p(X)$ and $p(Y \mid X)$.[3] As Kolmogorov complexity is not computable, these inference rules require practical implementations. To this end it has been proposed to infer whether X causes Y based on whether $p(X)$ and $p(Y \mid X)$ are independent. In particular, [JMZ+12] proposed multiple information-geometric approaches to detect (in)dependencies between $p(X)$ and $p(Y \mid X)$, with specialization to the case when X and Y are univariate and deterministically related (no noise). [JHS10, ZJZ11] studied causal inference for strictly multivariate X and Y linked by linear relationship $Y = \mathbf{A} \times X + \mathbf{E}$. They modeled the property of $p(X)$ through the covariance matrix $\sum_{XX}$ of X and detected (in)dependencies between $\sum_{XX}$ and $\mathbf{A}$. [CZC13] improved upon [JHS10, ZJZ11] by allowing non-linear correlations, but, require correlations to be deterministic, functional, and invertible.

Causal inference based on additive noise model (ANM) [SHHK06, HJM+08, MJHS11, PJS11] postulates that if $Y = f(X) + E$ with X being the cause, Y being the effect, and E being an additive error term that is statistically independent of X, then typically there is no additive noise model for the direction which maps the Y to X. [ZH09] generalizes ANM to the post-nonlinear model (PNL) where $Y = h(f(X) + E)$. The intuition of both ANM and PNL can be justified by the above algorithmic information-theoretic approach, in particular the algorithmic independence postulate.

[3] This approach in fact handles causal DAGs—we use the case of pairwise inference for illustration purposes.

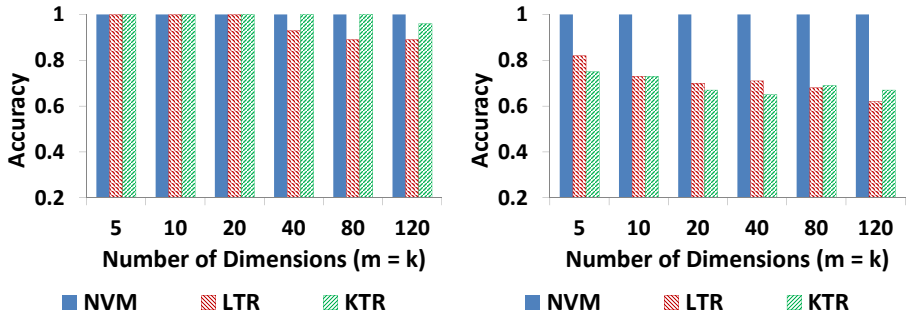
Apart from the above types of approach, there exists work based on the Markov kernel principle [SJS06, SJS08, JS10, MSJ+10]. As shown in Section 9.3, these define special cases of our inference principle.

In terms of data types, [SJS06, SHHK06, HJM+08, ZH09, MSJ+10, JHS10, ZJZ11, MJHS11, PJS11, JMZ+12, CZC13] mainly focus on numerical (real/integer-valued) data. [SJS08] allows for categorical data, however, due to high computational cost, restrict themselves to the binary case.

Recently, ERGO, an independent development of NOVUM was published as [Vre15]. ERGO is developed after NOVUM was created. To clarify, ERGO and this thesis were submitted for review roughly at the same time. Both methods are similar in spirit, in the sense that they derive causal indicators from Kolmogorov complexity. How they do so differs between the methods, in particular over which parts of the data complexity w.r.t. Kolmogorov complexity need to be included in the causality assessment, as well as how to do so. Nevertheless, they are not exclusive and should be considered as different approaches towards the same goal; similarly as that Shannon entropy and cumulative entropy provide different ways to quantify the entropy of the data, while not excluding each other. To the contrary, as we have shown in Chapters 5 and 6 both types of entropy in combination feature a powerful approach for correlation analysis. Our preliminary analysis shows that in general NOVUM and ERGO have similar performance on synthetic and real-world data sets, which corroborates our argument that they do not exclude each other. Another point to highlight is that NOVUM is applicable to mixed data types while ERGO currently focuses exclusively on real-valued data (its extension to mixed data types still is open). As we mentioned at the beginning of this chapter, development of NOVUM is on-going and there are many interesting ways to extend/modify it, besides the alternate complexity consideration that ERGO proposes.

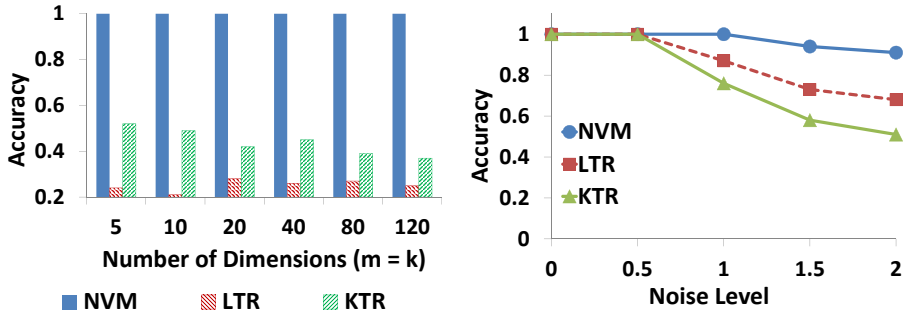
9.9. Conclusion

In this chapter, we focus on inferring the causal direction of two random variables X and Y. To tackle the problem, we have proposed and argued for a new causal inference principle. On a conceptual level, we model our principle after Kolmogorov complexity. From a practical point of view, we implement it by means of cumulative entropy and Shannon entropy. Accordingly, we introduce NOVUM for efficient causal inference based on our principle. NOVUM allows for flexible processing. It does not make any assumption on the form of correlation between X and Y, be it linear or non-linear, functional or non-functional. Further, it allows X and Y to contain mixed data types, and to be either univariate and multivariate. To verify the merits of NOVUM, we have performed extensive experiments on both synthetic and real-world data sets. The results show that NOVUM achieves both high accuracy and high efficiency, and hence, is suited for causal inference in large and high dimensional data.



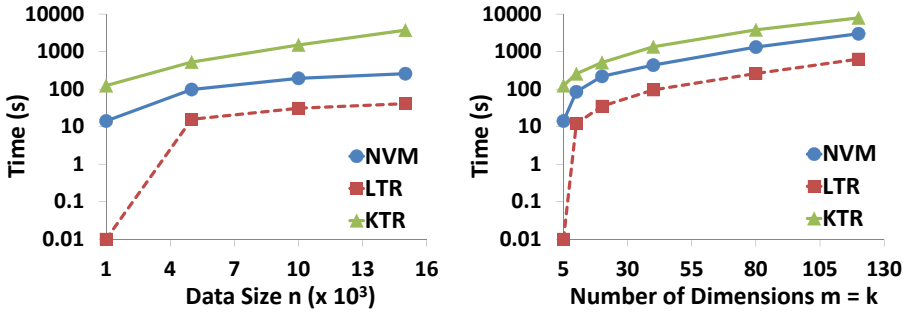
(a) (Higher is better) Accuracy against $D = k$ with functional dependency f_1

(b) (Higher is better) Accuracy against $D = k$ with functional dependency f_2



(c) (Higher is better) Accuracy against $D = k$ with functional dependency f_3

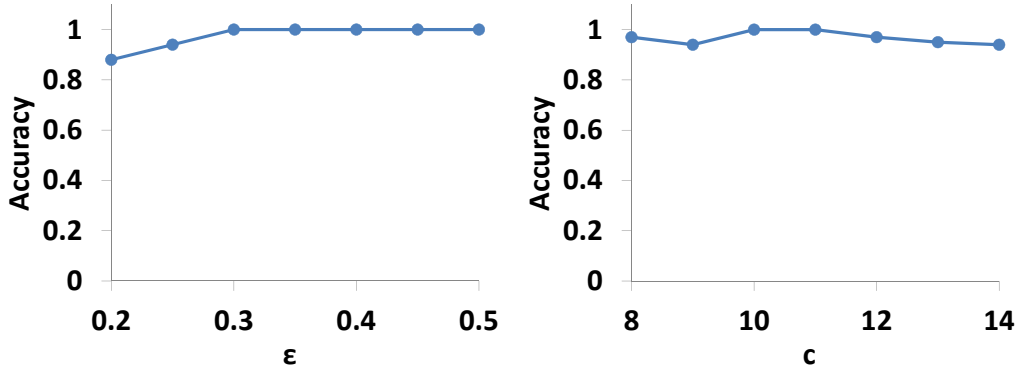
(d) (Higher is better) Accuracy against σ with $N = 1000$ and $D = k = 5$



(e) (Lower is better) Runtime against N with $\sigma = 0.5$ and $D = k = 5$

(f) (Lower is better) Runtime against $D = k$ with $N = 1000$ and $\sigma = 0.5$

Figure 9.1.: Results on synthetic data: Scalability to noise (by varying σ), data size (by varying N), and number of dimensions (by varying $D = k$). f_1, f_2, and f_3 are the functions describing the relationship between X and Y (please refer to Section 9.7.1 for their definition). For the runtime plots, the vertical axes are in log-scale.



(a) Accuracy vs. ϵ (b) Accuracy vs. c

Figure 9.2.: [Higher is better] Sensitivity of NOVUM to ϵ and c on synthetic data sets with $N = 1000$ and $D = k = 5$.

Part VI.

Summary

10. Conclusions

In this chapter, we first summarize the main contributions of this thesis. Then, we give an internal comparison of the methods we proposed in this thesis. Finally, we provide pointers to future work.

10.1. Summarization

Correlation analysis is one of the key elements of modern statistics and has wide impacts across many areas of applied science, e.g., databases, data mining, machine learning, signal processing, biology, to name a few. Considering the fact that nowadays in many real-world applications, data is collected in increasingly multivariate spaces having hundreds of dimensions, correlation analysis, aiming at analyzing the dependencies among dimensions which in turn are reliable indicators of the existence of interesting patterns, becomes a promising tool for extracting useful knowledge out of high dimensional data.

With such a motivation, in this thesis, we studied the problem of correlation analysis in multivariate data, with a special focus on mining correlated subspaces. We identified five major challenges specifically relevant for correlated subspace mining, which are complex types of correlation, computability, efficiency, mixed data types, and redundancy of output subspaces. Accordingly, we proposed several non-parametric methods to tackle all of the challenges. Overall, the main contributions are as follows.

- We introduced CMI and CMI++, two new multivariate correlation measures. We built them based on cumulative entropy. Since they do not make any assumption on the type of correlation, they are able to capture both linear and non-linear, functional and non-functional correlations. Further, we proved that both measures possess all important properties that total correlation [CT06] has on discrete/categorical data. We also presented effective and efficient methods for computing CMI and CMI++. Especially, with CMI++,

we demonstrated the benefits of correlation-aware discretization for the first time in this thesis. To mine correlated subspaces, we applied both measures to the Apriori subspace search scheme proposed in [KMB12]. Experiments on both synthetic and real-world data sets showed that our methods bring significant quality improvement for both clustering and outlier detection—two famous data mining tasks.

- We identified the issues associated with the Apriori search scheme: (a) it is not highly scalable, (b) it tends to miss high dimensional correlated subspaces, (c) it fragments them into many redundant lower dimensional subspaces, and (d) it is prone to the curse of dimensionality. Because of these issues, the Apriori search scheme does not achieve high scalability and high quality. Thus, we developed a new search scheme based on jump search. This scheme is able to jump directly to the relevant high dimensional subspaces by analyzing the statistics of their two dimensional projections. However, to further improve scalability, one needs to mine the two dimensional subspaces efficiently. Therefore, we studied a correlation measure belonging to the class of quadratic measures of (in)dependence. This measure also is based on cumulative distribution functions (cdfs) and does not assume a specific type of correlation. Most importantly, its computation on empirical data is fully in closed form. We showed that this closed form allows an efficient approximation based on sketching. Combining the jump search scheme and the efficient method for mining two dimensional subspaces, 4S—our new method for correlated subspace mining—achieved high scalability in very high dimensional data. We also incorporated into 4S an MDL-based phase of merging subspaces, which ensures succinctness of output as well as retrieves fragmented high dimensional correlated subspaces. Experiments showed 4S to detect high quality correlated subspaces that are useful for clustering, outlier mining, and classification.
- All methods above are designed specifically for real-valued data. To handle mixed typed data, we proposed DECOREL which takes advantage of the knowledge gained from the CMI measures and the scalable search scheme of 4S. For correlation assessment, since we followed the search scheme of 4S, we proposed a pairwise correlation measure. It nicely combines the good properties of both mutual information and CMI++. We also employed correlation-aware discretization to compute this correlation measure. Besides, we also modified the search scheme of 4S to better fit mixed data types and the multi-relational nature of the data. Extensive experiments showed that DECOREL well discovered groups of correlated columns in relational databases with heterogeneous data types. We also demonstrated that the groups found by DECOREL are beneficial for many database applications, e.g., selectivity estimation and schema extraction.
- We studied multivariate maximal correlation analysis, which we generalize from various existing methods for correlation analysis. Computing total correlation through correlation-aware discretization is an instantiation of this general notion. We introduced MAC to solve this task. Further, we extended

MAC to SUPMAC which incorporates external information, and MIXEDMAC which handles mixed typed data. As total correlation has a wide impact in several areas of applied science, MAC and its extensions open various venues of applications.

- We furthered our study on correlation-aware discretization by proposing IPD. It is not restricted to any specific notion of correlation. Instead, it aims to preserve more general interactions among dimensions by analyzing their multivariate distributions in consecutive data regions. IPD belongs to the class of multivariate discretization techniques. However, unlike existing work, we introduced an objective function for this task. This function successfully balances between preserving interactions of dimensions and the detail of the dimension under discretization. We developed two efficient algorithms for solving the objective function; one is optimal, the other is a 2-approximation of the optimal. Furthermore, we proposed a novel distance function for assessing the difference between two multivariate distributions. Extensive experiments on both synthetic and real-world data demonstrated the superiority of IPD compared to state of the art methods in pattern-based compression, outlier detection, and classification.
- We investigated causality analysis from an information-theoretic point of view. In particular, given two multivariate random variables X and Y with some correlation relationship and with the same number of observations, we aimed at efficiently inferring their causal direction. We addressed this by proposing a new principle for causal inference that is based on Kolmogorov complexity and which makes use of the algorithmic Markov condition. We showed that our principle generalizes various existing methods that rely on the principle of plausible Markov kernels. As Kolmogorov complexity is not computable, we presented NOVUM—an efficient non-parametric implementation of our principle based on cumulative and Shannon entropy. Through extensive experiments on both synthetic and real-world data, we showed that NOVUM is applicable to mixed typed data as well as has good scalability. Further, it yielded both high accuracy and high efficiency on both deterministic and noisy data.

In conclusion, this thesis advances the research of non-parametric correlation analysis in multivariate data. It enables scalable mining of correlated subspaces in high dimensional data. In addition, it shows that correlated subspaces are indeed beneficial for many tasks of data mining. Besides, it improves correlation-aware as well as interaction-preserving discretization. Last but not least, it contributes to multivariate causality analysis.

10.2. Comparison of Our Methods

In terms of correlation measures, we see a strong relationship between CMI measures and MAC. In particular, they all are information-theoretic measures: CMI/CMI++ is defined based on cumulative entropy while MAC is defined based

on Shannon entropy. Further, CMI++ and MAC both yield normalized scores, facilitating unbiased correlation assessment. We conceive both measures to be highly applicable for correlation analysis. Likewise, MIXEDMAC and the correlation measure of DECOREL are related: The former handles mixed data types using Shannon entropy while the latter combines both types of entropy. As cumulative entropy is relatively new compared to Shannon entropy, we see that more research effort is required to fully study cumulative entropy, and hence, to help us understand more about the characteristics of CMI++ and the correlation measure of DECOREL. This in turn will assist us in better judging in which scenarios one measure is more applicable than the rests. Compared to information-theoretic measures, in particular CMI measures and MAC, the quadratic measure of dependence proposed in Chapter 4 has both plus and minus. That is, it is handier thanks to its closed form computation on empirical data. Yet, it is more prone to the curse of dimensionality due to its reliance on the full space cdf. So the dimensionality of the data at hand may be an important factor in deciding which which measure to use.

In terms of search schemes, we note that though the jump search can find much higher dimensional correlated subspaces than the Apriori search scheme, it is only becomes really effective when the knowledge is located in high dimensional subspaces. That is, if interesting patterns are mostly in low dimensional subspaces (consisting of, e.g., 2 to 3 dimensions), the Apriori search scheme might already be sufficient. Nevertheless, one could still apply the jump search in such scenarios and extract low dimensional correlated subspaces in the post-analysis phase.

In terms of processing relational databases, DECOREL is dedicated for this task. However, we note that 4S may be effective as well if the database under consideration (a) contains only a few tables of medium sizes (i.e., storing the universal table does not cause much overhead), and (b) all columns are numeric. In such a case, as 4S has an efficient mining of pairwise correlations, it potentially is more efficient than DECOREL. Yet, for relational databases with heterogeneous data types, DECOREL remains a better choice.

In terms of discretization, CMI++, DECOREL, MAC and its variants aim at discretizing the data to optimize correlation w.r.t. their specific correlation measures. IPD in turn is a generic discretization technique. That is, it optimize the discretization towards interactions of dimensions without being constrained to any correlation measure. Thus, IPD is more suitable to scenarios where the correlation analysis task needs not be limited to any particular correlation measure. On the other hand, in very many applications where total correlation has been applied, MAC is natural choice.

10.3. Future Work

In this work, we focused on the non-parametric approach for tackling the issue of correlation analysis in multivariate data. It is interesting to study parametric and kernel methods as well. In particular, parametric methods are useful when

we know the type of data distribution, e.g., if we know that the distribution is a mixture of exponential distributions, we can learn it from the data. On the other hand, kernel estimation is highly suitable for the pairwise case as it does not suffer the curse of dimensionality, and it offers reliable computation on empirical data.

We considered cumulative distribution functions (cumulative entropy) and advanced discretization as key tools for computing correlation measures. It is worth to investigate other approaches, such as statistical estimation methods based on nearest neighbor analysis. While such methods require the data to be embedded in a metric space, they become effective when a relevant metric distance function is well defined for the data. In addition, one could also consider using the theory of copula to model and mine correlation pattens.

We introduced the novel notion of multivariate maximal correlation analysis. Further, we solved a specific instantiation of this notion, which is total correlation. Solving other instances is interesting as well. For example, generalizing CCA for the multivariate non-linear setting is an important research direction. While for CCA, we already have the objective function to optimize, it is still open how to form the optimization problem for other correlation measures as well as other sub-problems of correlation analysis.

Our research mostly targets the unsupervised setting. However, our work on SUPMAC has demonstrated the potential of incorporating external knowledge to the correlation analysis process. Generalizing our findings, one can examine constraint-based correlation analysis. The constraints can be first extracted out of the background knowledge and then fed into the computation of correlation measures, as we did in SUPMAC. As constraint-based clustering has been applied successfully in practice, we hypothesize that constraint-based correlation analysis could also bring about a similar impact on real-world applications.

In addition, we plan to apply the research here to the area of graphical modeling [Whi90, WJ08, PLM13]. In a nutshell, graphical modeling aims at "modeling and studying complex dependencies among random variables, and building large-scale multivariate statistical models" [WJ08]. To construct a graph representing the correlations among variables, one needs to compute the correlation score of each pair of variables. Thus, the first application of our research is to use the correlation measures covered in this thesis to construct such a graph. In fact, the notion of independence graph we discussed in Section 5.4, Chapter 5 is a type of graphical model [Whi90]: It shows how our measures can help to factorize a complex multivariate probability distribution into two dimensional distributions (which are easier to model), and hence, facilitate large-scale statistical analysis. The second application of our work to graphical modeling is to build *conditional independence graphs* [WJ08]. In a zero- and first-order conditional independence graph, an edge exists between two nodes (two variables) if and only if (a) they are correlated, and (b) they are not conditionally independent given any other variable. Our aim is to build such a graph in a non-parametric manner. As our measures already are non-parametric, we need non-parametric measures of conditional independence. To this end, we can apply a measure recently proposed in [SP12]. However, this

measure is kernel-based and is thus sensitive to the choice of kernel function. Hence, our goal is to develop new measures of conditional independence that are neither parametric nor kernel-based. To this end, we see potential opportunities from extending our existing correlation measures to capture conditional independencies. As the third application of our work to graphical modeling, we aim to use the subspaces detected by our methods together with their correlation scores to build graphs of variables that can represent more information on their dependencies. In particular, instead of just using pairwise (conditional) (in-)dependencies, we target at integrating groups containing three or more correlated variables into the process of graph construction. We hypothesize that this would lead to a better graphical representation of the relationships of variables.

Bibliography

[AABL13] G. Andrew, R. Arora, J. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *ICML (3)*, 2013, pp. 1247–1255.

[ABK+07a] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek, "On exploring complex relationships of correlation clusters," in *SSDBM*, 2007, p. 7.

[ABK+07b] ——, "Robust, complete, and efficient correlation clustering," in *SDM*, 2007, pp. 413–418.

[AC10] N. Ailon and B. Chazelle, "Faster dimension reduction," *Communications of the ACM*, vol. 53, no. 2, pp. 97–104, 2010.

[Ach08] S. Achard, "Asymptotic properties of a dimension-robust quadratic dependence measure," *Comptes Rendus Mathematique*, vol. 346, no. 3, pp. 213–216, 2008.

[AF94] J. F. Allen and G. Ferguson, "Actions and events in interval temporal logic," *J. Log. Comput.*, vol. 4, no. 5, pp. 531–579, 1994.

[AGGR98] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *SIGMOD*, 1998, pp. 94–105.

[AHHR09] A. Aue, S. Hörmann, L. Horváth, and M. Reimherr, "Break detection in the covariance structure of multivariate time series models," *Annals of Statistics*, vol. 37, no. 6B, pp. 4046–4087, 2009.

[AHS+09] B. Ahmadi, M. Hadjieleftheriou, T. Seidl, D. Srivastava, and S. Venkatasubramanian, "Type-based categorization of relational attributes," in *EDBT*, 2009, pp. 84–95.

[AKMS07] I. Assent, R. Krieger, E. Müller, and T. Seidl, "DUSC: Dimensionality unbiased subspace clustering," in *ICDM*, 2007, pp. 409–414.

[AKMS08a] I. Assent, R. Krieger, E. Müller, and T. Seidl, "EDSC: efficient density-based subspace clustering," in *CIKM*, 2008, pp. 1093–1102.

[AKMS08b] ——, "INSCY: Indexing subspace clusters with in-process-removal of redundancy," in *ICDM*, 2008, pp. 719–724.

[AL97] I. A. Ahmad and Q. Li, "Testing independence by nonparametric kernel method," *Statistics and Probability Letters*, vol. 34, no. 2, pp. 201–210, 1997.

[AL13] R. Arora and K. Livescu, "Multi-view cca-based acoustic features for phonetic recognition across speakers and domains," in *ICASSP*, 2013, pp. 7135–7139.

[Ald95] J. Aldrich, "Correlations genuine and spurious in pearson and yule," *Statistical Science*, vol. 10, no. 4, pp. 364–376, 1995.

[All83] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[AMS96] N. Alon, Y. Matias, and M. Szegedy, "The space complexity of approximating the frequency moments," in *STOC*, 1996, pp. 20–29.

[APW+99] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park, "Fast algorithms for projected clustering," in *SIGMOD*, 1999, pp. 61–72.

[AR10] M. S. Aziz and C. K. Reddy, "A robust seedless algorithm for correlation clustering," in *PAKDD*, 2010, pp. 28–37.

[AS94] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *VLDB*, 1994, pp. 487–499.

[ATVF12] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos, "Fast and reliable anomaly detection in categorical data," in *CIKM*, 2012, pp. 415–424.

[AV09] P. Adriaans and P. Vitányi, "Approximation of the two-part MDL code," *IEEE Transactions on Information Theory*, vol. 55, no. 1, pp. 444–457, 2009.

[AY00] C. C. Aggarwal and P. S. Yu, "Finding generalized projected clusters in high dimensional spaces," in *SIGMOD Conference*, 2000, pp. 70–81.

[AY01] ——, "Outlier detection for high dimensional data," in *SIGMOD*, 2001, pp. 37–46.

[Bay01] S. D. Bay, "Multivariate discretization for set mining," *Knowledge and Information Systems*, vol. 3, no. 4, pp. 491–512, 2001.

[BCK08] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in *SDM*, 2008, pp. 243–254.

[BF85] L. Breiman and J. H. Friedman, "Estimating optimal transformations for multiple regression and correlation," *J. Am. Stat. Assoc.*, vol. 80, no. 391, pp. 580–598, 1985.

[BGRS99] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *ICDT*, 1999, pp. 217–235.

[BKKZ04] C. Böhm, K. Kailing, P. Kröger, and A. Zimek, "Computing clusters of correlation connected objects," in *SIGMOD Conference*, 2004, pp. 455–466.

[BKRTN00] M. M. Breunig, H.-P. Kriegel, and J. S. Raymond T. Ng, "LOF: Identifying density-based local outliers," in *SIGMOD*, 2000, pp. 93–104.

[BL13] K. Bache and M. Lichman, "UCI machine learning repository," 2013.

[BLN05] S. Bu, L. V. S. Lakshmanan, and R. T. Ng, "Mdl summarization with holes," in *VLDB*, 2005, pp. 433–444.

[Bou06] M. Boullé, "MODL: A bayes optimal discretization method for continuous attributes," *Machine Learning*, vol. 65, no. 1, pp. 131–165, 2006.

[BP99] S. D. Bay and M. J. Pazzani, "Detecting change in categorical data: Mining contrast sets," in *KDD*, 1999, pp. 302–306.

[BPZL12] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *JMLR*, vol. 13, pp. 27–66, 2012.

[Bre01] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[Car68] J. D. Carroll, "Generalization of canonical correlation analysis to three or more sets of variables," in *Proceedings of the American Psychological Association*, 1968, pp. 227–228.

[CFZ99] C. H. Cheng, A. W.-C. Fu, and Y. Zhang, "Entropy-based subspace clustering for mining numerical data," in *KDD*, 1999, pp. 84–93.

[CGA+01] G. Chechik, A. Globerson, M. J. Anderson, E. D. Young, I. Nelken, and N. Tishby, "Group redundancy measures reveal redundancy reduction in the auditory pathway," in *NIPS*, 2001, pp. 173–180.

[CKKZ13] B. Chang, U. Krüger, R. Kustra, and J. Zhang, "Canonical correlation analysis based on hilbert-schmidt independence criterion and centered kernel target alignment," in *ICML (2)*, 2013, pp. 316–324.

[CL09] A. D. Crescenzo and M. Longobardi, "On cumulative entropies," *Journal of Statistical Planning and Inference*, vol. 139, no. 2009, pp. 4072–4087, 2009.

[Coo97] G. F. Cooper, "A simple constraint-based algorithm for efficiently mining observational databases for causal relationships," *Data Min. Knowl. Discov.*, vol. 1, pp. 203–224, 1997.

[CT06] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley-Interscience New York, 2006.

[CYZR10] P. Chanda, J. Yang, A. Zhang, and M. Ramanathan, "On mining statistically significant attribute association information," in *SDM*, 2010, pp. 141–152.

[CZC13] Z. Chen, K. Zhang, and L. Chan, "Nonlinear causal discovery for high dimensional data: A kernelized trace method," in *ICDM*, 2013, pp. 1003–1008.

[DAN+14] X. H. Dang, I. Assent, R. T. Ng, A. Zimek, and E. Schubert, "Discriminative features for identifying and interpreting outliers," in *ICDE*, 2014, pp. 88–99.

[DB04] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *JMLR*, vol. 5, no. 8, pp. 909–921, 2004.

[DB14] X. H. Dang and J. Bailey, "Generating multiple alternative clusterings via globally optimal subspaces," *Data Min. Knowl. Discov.*, vol. 28, no. 3, pp. 569–592, 2014.

[DCSL02] M. Dash, K. Choi, P. Scheuermann, and H. Liu, "Feature selection for clustering - a filter solution," in *ICDM*, 2002, pp. 115–122.

[Dem06] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[DGGR02] A. Dobra, M. N. Garofalakis, J. Gehrke, and R. Rastogi, "Processing complex aggregate queries over data streams," in *SIGMOD Conference*, 2002, pp. 61–72.

[EKSX96] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226–231.

[ELS10] D. Eppstein, M. Löffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time," in *ISAAC (1)*, 2010, pp. 403–414.

[FB05] S. Ferrandiz and M. Boullé, "Multivariate discretization by recursive supervised bipartition of graph," in *MLDM*, 2005, pp. 253–264.

[FI93] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *IJCAI*, 1993, pp. 1022–1029.

[GDJ11] Y. Guan, J. G. Dy, and M. I. Jordan, "A unified probabilistic model for global and local unsupervised feature selection," in *ICML*, 2011, pp. 1073–1080.

[GFM+11] S. Günnemann, I. Färber, E. Müller, I. Assent, and T. Seidl, "External evaluation measures for subspace clustering," in *CIKM*, 2011, pp. 1363–1372.

[GFVS12] S. Günnemann, I. Färber, K. Virochsiri, and T. Seidl, "Subspace correlation clustering: finding locally correlated dimensions in subspace projections of the data," in *KDD*, 2012, pp. 352–360.

[GHP+14] S. Goebl, X. He, C. Plant, , and C. Böhm, "Finding the optimal subspace for clustering," in *ICDM*, 2014.

[GN09] B. Guo and M. S. Nixon, "Gait feature subset selection by mutual information," *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 39, no. 1, pp. 36–46, 2009.

[Grü07] P. D. Grünwald, *The Minimum Description Length Principle*. MIT Press, 2007.

[Hal00] M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *ICML*, 2000, pp. 359–366.

[Han78] T. S. Han, "Nonnegative entropy measures of multivariate symmetric correlations," *Information and Control*, vol. 36, no. 2, pp. 133–156, 1978.

[HCN05] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *NIPS*, 2005, pp. 507–514.

[HFK+14] X. He, J. Feng, B. Konte, S. T. Mai, and C. Plant, "Relevant overlapping subspace clusters on categorical data," in *KDD*, 2014, pp. 213–222.

[HJM+08] P. O. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf, "Nonlinear causal discovery with additive noise models," in *NIPS*, 2008, pp. 689–696.

[HK01] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kaufmann, 2001.

[Ho98] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[Hot36] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[HQRA+13] A. Heise, J.-A. Quiané-Ruiz, Z. Abedjan, A. Jentzsch, and F. Naumann, "Scalable discovery of unique column combinations," *PVLDB*, vol. 7, no. 4, pp. 301–312, 2013.

[HSST04] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[Hua97] Z. Huang, "A fast clustering algorithm to cluster very large categorical data sets in data mining," in *DMKD*, 1997.

[HZW+13] G. Herman, B. Zhang, Y. Wang, G. Ye, and F. Chen, "Mutual information-based method for selecting informative feature sets," *Pattern Recognition*, vol. 46, no. 12, pp. 3315–3327, 2013.

[IMH+04] I. F. Ilyas, V. Markl, P. J. Haas, P. Brown, and A. Aboulnaga, "CORDS: Automatic discovery of correlations and soft functional dependencies," in *SIGMOD*, 2004, pp. 647–658.

[Jay82] E. T. Jaynes, "On the rationale of maximum-entropy methods," *Proceedings of the IEEE*, vol. 70, no. 9, pp. 939–952, 1982.

[JHS10] D. Janzing, P. O. Hoyer, and B. Schölkopf, "Telling cause from effect based on high-dimensional observations," in *ICML*, 2010, pp. 479–486.

[JMK09] X. Jiang, B. Mandal, and A. C. Kot, "Complete discriminant evaluation and feature extraction in kernel space for face recognition," *Mach. Vis. Appl.*, vol. 20, no. 1, pp. 35–46, 2009.

[JMZ+12] D. Janzing, J. Mooij, K. Zhang, J. Lemeire, J. Zscheischler, P. Daniušis, B. Steudel, and B. Schölkopf, "Information-geometric approach to inferring causal directions," *Artif. Intell.*, vol. 182-183, pp. 1–31, 2012.

[JP09] D. Jiang and J. Pei, "Mining frequent cross-graph quasi-cliques," *TKDD*, vol. 2, no. 4, 2009.

[JS10] D. Janzing and B. Schölkopf, "Causal inference using the algorithmic markov condition," *IEEE Trans. Inf. Theor.*, vol. 56, no. 10, pp. 5168–5194, 2010.

[KC04] L. A. Kurgan and K. J. Cios, "CAIM discretization algorithm," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 2, pp. 145–153, 2004.

[Ken38] M. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1-2, pp. 81–89, 1938.

[Ker92] R. Kerber, "ChiMerge: Discretization of numeric attributes," in *AAAI*, 1992, pp. 123–128.

[Ket71] J. R. Kettenring, "Canonical analysis of several sets of variables," *Biometrika*, vol. 58, no. 3, pp. 433–451, 1971.

[KG10] A. Koufakou and M. Georgiopoulos, "A fast outlier detection strategy for distributed high-dimensional data sets with mixed attributes," *Data Min. Knowl. Discov.*, vol. 20, no. 2, pp. 259–289, 2010.

[KGKB03] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchtold, "Efficient biased sampling for approximate clustering and outlier detection in large data sets," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 5, pp. 1170–1187, 2003.

[KJ97] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artif. Intell.*, vol. 97, no. 1-2, pp. 273–324, 1997.

[KKK04] P. Kröger, H.-P. Kriegel, and K. Kailing, "Density-connected subspace clustering for high-dimensional data," in *SDM*, 2004.

[KKKW03] K. Kailing, H.-P. Kriegel, P. Kröger, and S. Wanka, "Ranking interesting subspaces for clustering high dimensional data," in *PKDD*, 2003, pp. 241–252.

[KKRW05] H.-P. Kriegel, P. Kröger, M. Renz, and S. H. R. Wurst, "A generic framework for efficient subspace clustering of high-dimensional data," in *ICDM*, 2005, pp. 250–257.

[KKSZ12] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Outlier detection in arbitrarily oriented subspaces," in *ICDM*, 2012, pp. 379–388.

[KM07] P. Kontkanen and P. Myllymäki, "MDL histogram density estimation," *Journal of Machine Learning Research - Proceedings Track*, vol. 2, pp. 219–226, 2007.

[KMB12] F. Keller, E. Müller, and K. Böhm, "HiCS: High contrast subspaces for density-based outlier ranking," in *ICDE*, 2012, pp. 1037–1048.

[KN03] J. Kang and J. F. Naughton, "On schema matching with opaque column names and data values," in *SIGMOD*, 2003, pp. 205–216.

[KSG04] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Phys. Rev. E*, vol. 69, no. 6, pp. 1–16, 2004.

[KSZK09] H.-P. Kriegel, E. Schubert, A. Zimek, and P. Kröger, "Outlier detection in axis-parallel subspaces of high dimensional data," in *PAKDD*, 2009, pp. 831–838.

[KWL+06] Y. Kang, S. Wang, X. Liu, H. Lai, H. Wang, and B. Miao, "An ICA-based multivariate discretization algorithm," in *KSEM*, 2006, pp. 556–562.

[LB01] E. Levina and P. J. Bickel, "The earth mover's distance is the mallows distance: Some insights from statistics," in *ICCV*, 2001, pp. 251–256.

[LD06] J. Lemeire and E. Dirkx, "Causal models as minimal descriptions of multivariate systems," 2006.

[Liu07] J. Liu, "Information theoretic content and probability," Ph.D. dissertation, University of Florida, Gainesville, FL, USA, 2007. [Online]. Available: http://ufdc.ufl.edu/UFE0019613/

[LJ13] J. Lemeire and D. Janzing, "Replacing causal faithfulness with algorithmic independence of conditionals," *Minds and Machines*, vol. 23, no. 2, pp. 227–249, 2013.

[LK05] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *KDD*, 2005, pp. 157–166.

[LNW+02] L. V. S. Lakshmanan, R. T. Ng, C. X. Wang, X. Zhou, and T. Johnson, "The generalized mdl approach for summarization," in *VLDB*, 2002, pp. 766–777.

[LS10] J. Lemeire and K. Steenhaut, "Inference of graphical causal models: Representing the meaningful information of probability distributions," in *NIPS Causality: Objectives and Assessment*, 2010, pp. 107–120.

[LV93] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 1993.

[LV07] J. A. Lee and M. Verleysen, *Nonlinear Dimensionality Reduction*. Springer, New York, 2007.

[LW08] G. Liu and L. Wong, "Effective pruning techniques for mining quasi-cliques," in *ECML/PKDD (2)*, 2008, pp. 33–49.

[LY08] R. Liu and L. Yang, "Kernel estimation of multivariate cumulative distribution function," *Journal of Nonparametric Statistics*, vol. 20, no. 8, pp. 661–677, 2008.

[MAG+09] E. Müller, I. Assent, S. Günnemann, R. Krieger, and T. Seidl, "Relevant subspace clustering: Mining the most interesting non-redundant concepts in high dimensional data," in *ICDM*, 2009, pp. 377–386.

[MAGS11] E. Müller, I. Assent, S. Günnemann, and T. Seidl, "Scalable density-based subspace clustering," in *CIKM*, 2011, pp. 1077–1086.

[MAK+09] E. Müller, I. Assent, R. Krieger, S. Günnemann, and T. Seidl, "DensEst: Density estimation for data mining in high dimensional spaces," in *SDM*, 2009, pp. 175–186.

[MAS09] E. Müller, I. Assent, and T. Seidl, "HSM: Heterogeneous subspace mining in high dimensional data," in *SSDBM*, 2009, pp. 497–516.

[MGAS09] E. Müller, S. Günnemann, I. Assent, and T. Seidl, "Evaluating clustering in subspace projections of high dimensional data," *PVLDB*, vol. 2, no. 1, pp. 1270–1281, 2009.

[MJHS11] J. M. Mooij, D. Janzing, T. Heskes, and B. Schölkopf, "On causal discovery with cyclic additive noise models," in *NIPS*, 2011, pp. 639–647.

[MNDA13] B. Micenková, R. T. Ng, X. H. Dang, and I. Assent, "Explaining outliers by subspace separability," in *ICDM*, 2013, pp. 518–527.

[MPY05] S. Mehta, S. Parthasarathy, and H. Yang, "Toward unsupervised correlation preserving discretization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1174–1185, 2005.

[MSJ+10] J. M. Mooij, O. Stegle, D. Janzing, K. Zhang, and B. Schölkopf, "Probabilistic latent variable models for distinguishing between cause and effect," in *NIPS*, 2010, pp. 1687–1695.

[MSS11] E. Müller, M. Schiffer, and T. Seidl, "Statistical selection of relevant subspace projections for outlier ranking," in *ICDE*, 2011, pp. 434–445.

[MV13] M. Mampaey and J. Vreeken, "Summarizing categorical data by clustering attributes," *Data Min. Knowl. Discov.*, vol. 26, no. 1, pp. 130–173, 2013.

[MVT12] M. Mampaey, J. Vreeken, and N. Tatti, "Summarizing data succinctly with the most informative itemsets," *TKDD*, vol. 6, no. 4, pp. 1–44, 2012.

[NCRB14] X. V. Nguyen, J. Chan, S. Romano, and J. Bailey, "Effective global approaches for mutual information based feature selection," in *KDD*, 2014, pp. 512–521.

[NGC01] H. S. Nagesh, S. Goil, and A. N. Choudhary, "Adaptive grids for clustering massive data sets," in *SDM*, 2001, pp. 1–17.

[NMAB14] H. V. Nguyen, E. Müller, P. Andritsos, and K. Böhm, "Detecting correlated columns in relational databases," in *SSDBM*, 2014, p. 30.

[NMB13] H. V. Nguyen, E. Müller, and K. Böhm, "4S: Scalable subspace search scheme overcoming traditional apriori processing," in *BigData Conference*, 2013, pp. 359–367.

[NMB14] ——, "A near-linear time subspace search scheme for unsupervised selection of correlated features," *Big Data Research*, vol. 1, pp. 37–51, 2014.

[NMV+13] H. V. Nguyen, E. Müller, J. Vreeken, F. Keller, and K. Böhm, "CMI: An information-theoretic contrast measure for enhancing subspace cluster and outlier detection," in *SDM*, 2013, pp. 198–206.

[NMV+14] H. V. Nguyen, E. Müller, J. Vreeken, P. Efros, and K. Böhm, "Multivariate maximal correlation analysis," in *ICML*, 2014, pp. 775–783.

[NMVB14] H. V. Nguyen, E. Müller, J. Vreeken, and K. Böhm, "Unsupervised interaction-preserving discretization of multivariate data," *Data Min. Knowl. Discov.*, vol. 28, no. 5-6, pp. 1366–1397, 2014.

[NV] H. V. Nguyen and J. Vreeken, "Information-theoretic causal discovery," in progress.

[OGP06] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast distributed outlier detection in mixed-attribute data sets," *Data Min. Knowl. Discov.*, vol. 12, no. 2-3, pp. 203–228, 2006.

[OO04] C. Ordonez and E. Omiecinski, "Efficient disk-based K-means clustering for relational databases," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 8, pp. 909–921, 2004.

[Pea09] J. Pearl, *Causality: Models, Reasoning and Inference*, 2nd ed. Cambridge University Press, 2009.

[PJS10] J. Peters, D. Janzing, and B. Schölkopf, "Identifying cause and effect on discrete data using additive noise models," in *AISTATS*, 2010, pp. 597–604.

[PJS11] ——, "Causal inference on discrete data using additive noise models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2436–2450, 2011.

[PLD05] H. Peng, F. Long, and C. H. Q. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, 2005.

[PLM13] N. Piatkowski, S. Lee, and K. Morik, "Spatio-temporal random fields: compressible representation and distributed estimation," *Machine Learning*, vol. 93, no. 1, pp. 115–139, 2013.

[PLP+10] M. Paradies, C. Lemke, H. Plattner, W. Lehner, K.-U. Sattler, A. Zeier, and J. Krüger, "How to juggle columns: an entropy-based approach for table compression," in *IDEAS*, 2010, pp. 205–215.

[PP12] N. Pham and R. Pagh, "A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data," in *KDD*, 2012, pp. 877–885.

[PP13] H. D. Philip Preuß, Ruprecht Puchstein, "Detection of multiple structural breaks in multivariate time series," *arXiv preprint*, vol. arXiv:1309.1309v1, 2013.

[PWR89] S. Peleg, M. Werman, and H. Rom, "A unified approach to the change of resolution: Space and gray-level," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 739–742, 1989.

[Rac84] S. T. Rachev, "The monge-kantorovich mass transference problem and its stochastic applications," *Theory Probab. Appl.*, vol. 29, no. 4, pp. 647–676, 1984.

[Rao05] M. Rao, "More on a new concept of entropy and information," *Journal of Theoretical Probability*, vol. 18, no. 14, pp. 967–981, 2005.

[RCVW04] M. Rao, Y. Chen, B. C. Vemuri, and F. Wang, "Cumulative residual entropy: A new measure of information," *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1220–1228, 2004.

[RD08] F. Rusu and A. Dobra, "Sketches for size of join estimation," *ACM Trans. Database Syst.*, vol. 33, no. 3, 2008.

[Ren59] A. Renyi, "On measures of dependence," *Acta Mathematica Academiae Scientiarum Hungarica*, vol. 10, no. 3-4, pp. 441–451, 1959.

[Ris78] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 1, pp. 465–471, 1978.

[Ris83] ——, "Modeling by shortest data description," *Annals of Statistics*, vol. 11, no. 2, pp. 416–431, 1983.

[RLHEC10] I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C. S. Cruz, "Quadratic programming feature selection," *JMLR*, vol. 11, pp. 1491–1516, 2010.

[RRF+11] D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti, "Detecting novel associations in large data sets," *Science*, vol. 334, no. 6062, pp. 1518–1524, 2011.

[RS11] A. Reiss and D. Stricker, "Towards global aerobic activity monitoring," in *PETRA*, 2011, p. 12.

[RSX+11] M. Rao, S. Seth, J.-W. Xu, Y. Chen, H. Tagare, and J. C. Príncipe, "A test of independence based on a generalized correlation function," *Signal Processing*, vol. 91, no. 1, pp. 15–27, 2011.

[SBHR06] Y. Sismanis, P. Brown, P. J. Haas, and B. Reinwald, "GORDIAN: Efficient and scalable discovery of composite keys," in *VLDB*, 2006, pp. 691–702.

[SBMU00] C. Silverstein, S. Brin, R. Motwani, and J. Ullman, "Scalable techniques for mining causal structures," *Data Min. Knowl. Discov.*, vol. 4, no. 2-3, pp. 163–192, 2000.

[SCH10] M. Sridhar, A. G. Cohn, and D. C. Hogg, "Unsupervised learning of event classes from video," in *AAAI*, 2010.

[SCRR11] L. Schietgat, F. Costa, J. Ramon, and L. D. Raedt, "Effective feature construction by maximum common subgraph sampling," *Machine Learning*, vol. 83, no. 2, pp. 137–161, 2011.

[SGS00] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*. MIT press, 2000.

[SGSS07] A. J. Smola, A. Gretton, L. Song, and B. Schölkopf, "A hilbert space embedding for distributions," in *ALT*, 2007, pp. 13–31.

[SHHK06] S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. J. Kerminen, "A linear non-gaussian acyclic model for causal discovery," *JMLR*, vol. 7, pp. 2003–2030, 2006.

[Sil86] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. London: Chapman & Hall/CRC, 1986.

[SJS06] X. Sun, D. Janzing, and B. Schölkopf, "Causal inference by choosing graphs with most plausible markov kernels," in *ISAIM*, 2006.

[SJS08] ——, "Causal reasoning by evaluating the complexity of conditional densities with kernel methods," *Neurocomputing*, vol. 71, no. 7-9, pp. 1248–1256, 2008.

[SM11] B. Schowe and K. Morik, "Fast-ensembles of minimum redundancy feature selection," in *Ensembles in Machine Learning Applications*, 2011, pp. 75–95.

[SNJ+13] J. D. Scargle, J. P. Norris, B. Jackson, , and J. Chiang, "Studies in astronomical time series analysis. vi. bayesian block representations," *Astrophysical Journal*, vol. 764, no. 2, 2013.

[SP12] S. Seth and J. C. Príncipe, "Assessing granger non-causality using nonparametric measure of conditional independence," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 23, no. 1, pp. 47–59, 2012.

[Spe87] C. Spearman, "The proof and measurement of association between two things," *The American Journal of Psychology*, vol. 15, no. 1, pp. 72–101, 1987.

[SR09] G. J. Székely and M. L. Rizzo, "Brownian distance covariance," *Annals of Applied Statistics*, vol. 3, no. 4, pp. 1236–1265, 2009.

[SRPP11] S. Seth, M. Rao, I. Park, and J. C. Príncipe, "A unified framework for quadratic measures of independence," *IEEE Transactions on Signal Processing*, vol. 59, no. 8, pp. 3624–3635, 2011.

[SSG+07] L. Song, A. J. Smola, A. Gretton, K. M. Borgwardt, and J. Bedo, "Supervised feature selection via dependence estimation," in *ICML*, 2007, pp. 823–830.

[SSTP09] M. C. Schmidt, N. F. Samatova, K. Thomas, and B.-H. Park, "A scalable, parallel algorithm for maximal clique enumeration," *J. Parallel Distrib. Comput.*, vol. 69, no. 4, pp. 417–428, 2009.

[SZ04] K. Sequeira and M. J. Zaki, "SCHISM: A new approach for interesting subspace mining," in *ICDM*, 2004, pp. 186–193.

[TDJ11] K. Tzoumas, A. Deshpande, and C. S. Jensen, "Lightweight graphical models for selectivity estimation without independence assumptions," *PVLDB*, vol. 4, no. 11, pp. 852–863, 2011.

[TGS09] R. E. Tillman, A. Gretton, and P. Spirtes, "Nonlinear directed acyclic structure learning with weakly additive noise models." in *NIPS*, 2009, pp. 1847–1855.

[TP95] A. Treves and S. Panzeri, "The upward bias in measures of information derived from limited data samples," *Neural Computation*, vol. 7, no. 2, pp. 399–407, 1995.

[TV08] N. Tatti and J. Vreeken, "Finding good itemsets by packing data," in *ICDM*, 2008, pp. 588–597.

[vdVT12] M. van de Velden and Y. Takane, "Generalized canonical correlation analysis with missing values," *Computational Statistics*, vol. 27, no. 3, pp. 551–571, 2012.

[Vre15] J. Vreeken, "Causal inference by direction of information," in *SDM*, 2015.

[VV04] N. K. Vereshchagin and P. M. B. Vitányi, "Kolmogorov's structure functions and model selection," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3265–3290, 2004.

[VvLS11] J. Vreeken, M. van Leeuwen, and A. Siebes, "Krimp: mining itemsets that compress," *Data Min. Knowl. Discov.*, vol. 23, no. 1, pp. 169–214, 2011.

[VvS11] J. Vreeken, M. van Leeuwen, and A. Siebes, "KRIMP: Mining itemsets that compress," *Data Min. Knowl. Discov.*, vol. 23, no. 1, pp. 169–214, 2011.

[Whi90] J. Whittaker, *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons, 1990.

[WJ08] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[WLV+14] A. Wagner, T. Lützkendorf, K. Voss, G. Spars, A. Maas, and S. Herkel, "Performance analysis of commercial buildings—results and experiences from the german demonstration program 'energy optimized building (EnOB)'," *Energy and Buildings*, vol. 68, pp. 634–638, 2014.

[Yin04] X. Yin, "Canonical correlation analysis based on information theory," *Journal of Multivariate Analysis*, vol. 91, no. 2, pp. 161–176, 2004.

[YLO09] M. Ye, X. Li, and M. E. Orlowska, "Projected outlier detection in high-dimensional mixed-attributes data set," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 7104–7113, 2009.

[YM05] M. L. Yiu and N. Mamoulis, "Iterative projected clustering by subspace mining," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 176–189, 2005.

[YPS11] X. Yang, C. M. Procopiuc, and D. Srivastava, "Summary graphs for relational database schemas," *PVLDB*, vol. 4, no. 11, pp. 899–910, 2011.

[YWWY02] J. Yang, W. Wang, H. Wang, and P. S. Yu, "delta-Clusters: Capturing subspace correlation in a large data set," in *ICDE*, 2002, pp. 517–528.

[ZH09] K. Zhang and A. Hyvärinen, "On the identifiability of the post-nonlinear causal model," in *UAI*, 2009, pp. 647–655.

[ZHO+11] M. Zhang, M. Hadjieleftheriou, B. C. Ooi, C. M. Procopiuc, and D. Srivastava, "Automatic discovery of attributes in relational databases," in *SIGMOD*, 2011, pp. 109–120.

[ZJZ11] J. Zscheischler, D. Janzing, and K. Zhang, "Testing whether linear equations are causal: A free probability theory approach," in *UAI*, 2011, pp. 839–846.

[ZL07] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *ICML*, 2007, pp. 1151–1157.

[ZPJS11] K. Zhang, J. Peters, D. Janzing, and B. Schölkopf, "Kernel-based conditional independence test and application in causal discovery," in *UAI*, 2011, pp. 804–813.

[ZPWN08] X. Zhang, F. Pan, W. Wang, and A. B. Nobel, "Mining non-redundant high order correlations in binary data," *PVLDB*, vol. 1, no. 1, pp. 1178–1188, 2008.

[Zsc13] J. Zscheischler, "Database with cause-effect pairs," 2013.