Summer 2012

Effective and efficient correlation analysis with application to market basket analysis and network community detection

Lian Duan
University of Iowa

EFFECTIVE AND EFFICIENT CORRELATION ANALYSIS WITH APPLICATION TO MARKET BASKET ANALYSIS AND NETWORK COMMUNITY DETECTION

by

Lian Duan

An Abstract

Of a thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Business Administration
in the Graduate College of
The University of Iowa

July 2012

Thesis Supervisor: Professor W. Nick Street

ABSTRACT

Finding the most interesting correlations among items is essential for problems in many commercial, medical, and scientific domains. For example, what kinds of items should be recommended with regard to what has been purchased by a customer? How to arrange the store shelf in order to increase sales? How to partition the whole social network into several communities for successful advertising campaigns? Which set of individuals on a social network should we target to convince in order to trigger a large cascade of further adoptions? When conducting correlation analysis, traditional methods have both effectiveness and efficiency problems, which will be addressed in this dissertation. Here, we explore the effectiveness problem in three ways. First, we expand the set of desirable properties and study the property satisfaction for different correlation measures. Second, we study different techniques to adjust original correlation measure, and propose two new correlation measures: the Simplified χ^2 with Continuity Correction and the Simplified χ^2 with Support. Third, we study the upper and lower bounds of different measures and categorize them by the bound differences. Combining with the above three directions, we provide guidelines for users to choose the proper measure according to their situations. With the proper correlation measure, we start to solve the efficiency problem for a large dataset. Here, we propose a fully-correlated itemset (FCI) framework to decouple the correlation measure from the need for efficient search. By wrapping the desired measure in our FCI framework, we take advantage of the desired measure's superiority in evaluating itemsets, eliminate itemsets with irrelevant items, and achieve good computational performance. In addition, we identify a 1-dimensional monotone property of the upper bound of any good correlation measure, and different 2-dimensional

monotone properties for different types of correlation measures. We can either use the 2-dimensional search algorithm to retrieve correlated pairs above a certain threshold, or our new Token-Ring algorithm to find top-k correlated pairs to prune many pairs without computing their correlations. In order to speed up FCI search, we build an enumeration tree to save the fully-correlated value (FCV) for all the FCIs under an initial threshold. We can either efficiently retrieve the desired FCIs for any given threshold above the initial threshold or incrementally grow the tree if the given threshold is below the initial threshold. With the theoretical analysis on correlation search, we applied our research to two typical applications: Market Basket Analysis and Network Community Detection.

Abstract Approved: _____
Thesis Supervisor

_____
Title and Department

_____
Date

EFFECTIVE AND EFFICIENT CORRELATION ANALYSIS WITH APPLICATION TO MARKET BASKET ANALYSIS AND NETWORK COMMUNITY DETECTION

by

Lian Duan

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Business Administration
in the Graduate College of
The University of Iowa

July 2012

Thesis Supervisor: Professor W. Nick Street

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Lian Duan

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Business Administration at the July 2012 graduation.

Thesis Committee: _____
W. Nick Street, Thesis Supervisor

_____
Gautam Pant

_____
Jeffrey W Ohlmann

_____
Padmini Srinivasan

_____
Sam Burer

ACKNOWLEDGEMENTS

I would like to sincerely thank my advisor Prof. Nick Street for his continuous support of my Ph.D research, and his patience and immense knowledge to refine my teaching and research skills. I cannot ask for a better advisor for my Ph.D study. Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Gautam Pant, Prof. Jeffrey W Ohlmann, Prof. Padmini Srinivasan, and Prof. Sam Burer, for their help and insightful comments. I thank my friends in the University of Iowa: Mohammad Khoshneshin, Michael Rechenthin, Brian Almquist, Si-Chi Chin, Chen Yang, Senay Yasar Saglam, Ray Hylock, Mahtab Ghazizadeh, Chris Harris, Jieqiu Chen, Zhe Song, Wenjun Wang, Bob Arens, and Viet Ha-Thuc, for their help and the time we were working together. Last but not the least, I would like to thank my parents (Shanfa Duan and Youxi Ke) and wife (Ningning Zhang) for supporting me spiritually throughout my life.

ABSTRACT

Finding the most interesting correlations among items is essential for problems in many commercial, medical, and scientific domains. For example, what kinds of items should be recommended with regard to what has been purchased by a customer? How to arrange the store shelf in order to increase sales? How to partition the whole social network into several communities for successful advertising campaigns? Which set of individuals on a social network should we target to convince in order to trigger a large cascade of further adoptions? When conducting correlation analysis, traditional methods have both effectiveness and efficiency problems, which will be addressed in this dissertation. Here, we explore the effectiveness problem in three ways. First, we expand the set of desirable properties and study the property satisfaction for different correlation measures. Second, we study different techniques to adjust original correlation measure, and propose two new correlation measures: the Simplified χ^2 with Continuity Correction and the Simplified χ^2 with Support. Third, we study the upper and lower bounds of different measures and categorize them by the bound differences. Combining with the above three directions, we provide guidelines for users to choose the proper measure according to their situations. With the proper correlation measure, we start to solve the efficiency problem for a large dataset. Here, we propose a fully-correlated itemset (FCI) framework to decouple the correlation measure from the need for efficient search. By wrapping the desired measure in our FCI framework, we take advantage of the desired measure's superiority in evaluating itemsets, eliminate itemsets with irrelevant items, and achieve good computational performance. In addition, we identify a 1-dimensional monotone property of the upper bound of any good correlation measure, and different 2-dimensional

monotone properties for different types of correlation measures. We can either use the 2-dimensional search algorithm to retrieve correlated pairs above a certain threshold, or our new Token-Ring algorithm to find top-k correlated pairs to prune many pairs without computing their correlations. In order to speed up FCI search, we build an enumeration tree to save the fully-correlated value (FCV) for all the FCIs under an initial threshold. We can either efficiently retrieve the desired FCIs for any given threshold above the initial threshold or incrementally grow the tree if the given threshold is below the initial threshold. With the theoretical analysis on correlation search, we applied our research to two typical applications: Market Basket Analysis and Network Community Detection.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF ALGORITHMS

Algorithm

LIST OF LEMMAS

Lemma

CHAPTER 1
INTRODUCTION

Correlation generally refers to a broad class of statistical dependence relationships among variables. The dependent phenomena include the products being purchased together, and the trend between price and sales. Correlations are very useful because they can be exploited for prediction in practice. However, many correlation patterns are under-exploited in practice because they are buried in a huge amount of data and not easy to find. Therefore, research on correlation analysis helps us find these under-exploited correlation patterns. The current correlation research mainly focuses on correlated itemset search [26, 27] for binary transaction data and canonical correlation search [45] for numeric data. In this thesis, we only focus on the correlation search for binary data.

1.1 Correlation Measures

With the development of scanning devices, the Internet, and computer storage technologies, retailing companies like Walmart compile large databases on consumers' past transactions, and online social network sites like Facebook store friend relationships among people. There are many interesting business questions to these companies. For example, what kinds of items should be recommended with regard to what has been purchased by a customer? How to arrange the store shelf in order to increase sales? How to partition the whole social network into several communities for successful advertising campaigns? Which set of individuals on a social network should we target to convince in order to trigger a large cascade of further adoptions? All the above questions are related to correlation search.

Although there are numerous measures available for evaluating correlations, different correlation measures provide drastically different results. Piatetsky-Shapiro [68] provided three mandatory properties for any reasonable correlation measure and Tan et al. [82] proposed several properties to categorize correlation measures; however, it is still hard for users to choose the desirable correlation measures according to their needs. In order to solve this problem, we explore the effectiveness problem in three ways. First, we expand the set of desirable properties and study the property satisfaction for different correlation measures. Second, we study different techniques to adjust original correlation measure, and propose two new correlation measures: the Simplified χ^2 with Continuity Correction and the Simplified χ^2 with Support. Third, we study the upper and lower bounds of different measures and categorize them by the bound differences. Combining with the above three directions, we provide guidelines for users to choose the proper measure according to their situations.

1.2 Applications

With the proper correlation measure, we apply our correlation analysis to market basket analysis. Finding the most interesting correlations in a collection of items is essential for problems in many commercial, medical, and scientific domains. Much previous research focuses on finding correlated pairs instead of correlated itemsets in which all items are correlated with each other. Though some existing methods find correlated itemsets of any size, they suffer from both efficiency and effectiveness problems in large datasets. Here, we propose a fully-correlated itemset (FCI) framework to decouple the correlation measure from the need for efficient search. By wrapping the desired measure in our FCI framework, we take advantage of the desired measure's superiority in evaluating itemsets, eliminate itemsets with irrelevant items,

and achieve good computational performance. However, FCIs must start pruning from 2-itemsets unlike frequent itemsets which can start the pruning from 1-itemsets. When the number of items in a given dataset is large and the support of all the pairs cannot be loaded into the memory, the IO cost $O(n^2)$ for calculating correlation of all the pairs is very high. In addition, users usually need to try different correlation thresholds and the cost of processing the Apriori procedure each time for a different threshold is very high. Consequently, we propose two techniques to solve the efficiency problem. With respect to correlated pair search, we identify a 1-dimensional monotone property of the upper bound of any good correlation measure, and different 2-dimensional monotone properties for different types of correlation measures. We can either use the 2-dimensional search algorithm to retrieve correlated pairs above a certain threshold, or our new Token-Ring algorithm to find top-k correlated pairs to prune many pairs without computing their correlations. In addition, in order to speed up FCI search, we build an enumeration tree to save the fully-correlated value (FCV) for all the FCIs under an initial threshold. We can either efficiently retrieve the desired FCIs for any given threshold above the initial threshold or incrementally grow the tree if the given threshold is below the initial threshold.

In addition, we try to apply our correlation analysis to community detection, a hot research topic related to social networks. Modularity [62] is by far the most used and the best community detection method. The modularity function uses the same idea of the correlation measure Leverage which calculates the difference between the actual number and the expected number of co-ocurrences. Inspired by other correlation measures, we can easily change the objective function to get different results. Since it is very hard to find the exact community information in real life network data, we make use of benchmark network simulation [50] to test the quality

of detected communities obtained by different objective functions.

The remainder of this thesis is organized as follows. In Chapter 2, important correlation properties and different correlation measures are discussed. The correlation properties are used to guide the choice of different correlation measures. A fully-correlated itemset (FCI) framework together with other speed-up techniques is proposed to decouple the correlation measure from the need for efficient search for correlated itemsets in Chapter 3. Then, we explore the future opportunity of making use of correlation analysis to change the objective function of classical community detection methods for different needs in Chapter 4. Finally, we draw a conclusion and discuss the future work in Chapter 5.

CHAPTER 2
CORRELATION MEASURES

2.1 Introduction and Related Work

Since the current correlation research for binary data mainly focuses on correlated itemset search, we discuss correlation measures in the context of market basket analysis. Each record in a typical market basket transaction dataset corresponds to a transaction, which contains a unique identifier and a set of items bought by a given customer. The analysis of relationships between items is fundamental in many data mining problems. For example, the central task of association analysis [2] is to discover a set of items that co-occur frequently in a transaction database. Regardless of how the relationships are defined, such analysis requires a proper measure to evaluate the dependencies among items.

Perhaps the most useful tactic for measuring the level of correlation of a set of items is by measuring the lack of independence of the items. Although there are numerous measures [37, 81] available for evaluating correlation, many of them provide conflicting information about the correlation of a given itemset. One of the most straightforward examples of this strategy is the popular χ^2 test for independence from statistics. An example (from [48]) of this test is as follows. We want to check for correlation between the purchase of beer and the purchase of diapers at a supermarket. We first produce a contingency matrix for the four possible events in Table 2.1. Then, we produce a similar matrix of expected values, based on an assumption of independence of the different events, in Table 2.2. The expected number of purchases of beer without diapers (given the assumption of independence) is $5729.9 = 6323 \cdot 50311/55519$; the expected number of purchases of beer with diapers is 593.1. Next,

we compute the χ^2 statistic, which is a sum over the four different cells r_{ij} in the contingency matrix: $\chi^2 = \sum_i \sum_j \frac{(r_{ij} - E(r_{ij}))^2}{E(r_{ij})}$. In our case, this sum is 2732.8 which is large compared to the χ^2 distribution. The probability of independence is nearly zero. We reject the null hypothesis and testify to the correlation between the purchase of beer and diapers.

	Diapers	No diapers	$\sum$ Row
Beer	1734	4589	6323
No beer	3474	45722	49196
$\sum$ Column	5208	50311	55519

Table 2.1: Actual sale of beer and diapers

	Diapers	No diapers	$\sum$ Row
Beer	593.1	5729.9	6323
No beer	4614.9	44581.1	49196
$\sum$ Column	5208	50311	55519

Table 2.2: Expected sale of beer and diapers

Although we are, in general, interested in correlated sets of arbitrary size, most of the published work with regard to correlation is related to finding correlated pairs [37, 81]. Related work with association rules [14, 15, 66] is a special case of correlation pairs since each rule has a left- and right-hand side. Given an association rule $X \Rightarrow Y$, $Support = P(X \cap Y)$ and $Confidence = P(X \cap Y)/P(X)$ [2] [66] are often used to represent its significance. However, these can produce misleading results because of the lack of comparison to the expected probability under the

assumption of independence. In order to overcome the shortcoming, Lift [14], Conviction [15], and Leverage [68] are proposed. $Lift = P(X \cap Y)/(P(X)P(Y))$ measures the ratio of X and Y's actual co-occurrence to the expected value under the independence assumption. $Conviction = (P(X)P(\bar{Y}))/P(X \cap \bar{Y})$ compares the probability that X appears without Y if they were independent with the actual frequency of the appearance of X without Y. $Leverage = P(X \cap Y) - P(X)P(Y)$ measures the difference of X and Y appearing together in the data set and what would be expected if X and Y were statistically independent. Dunning [30] introduced a more statistically reliable measure, Likelihood Ratio, which outperforms other correlation measures. Jermaine [48] extended Dunning's work and examined the computational issue of Probability Ratio and Likelihood Ratio. Bate et al. [7] proposed a correlation measure called Bayesian Confidence Propagation Neural Network (BCPNN) which is good at searching for correlated patterns occurring rarely in the whole dataset. The above correlation measures are intuitive; however, different correlation measures provide drastically different results. Although Tan et al. [81] proposed several properties to categorize these correlation measures and Suzuki [79] mentioned four pitfalls for correct categorization, we are more interested in guidelines for users to choose the desirable correlation measures according to their needs. Piatetsky-Shapiro [68] proposed three fundamental properties for a good correlation measure. The three fundamental properties can rule out some bad correlation measures, but other bad correlation measures can still satisfy all of them. Szczech et al. [80] proposed a new property for association rules instead of itemsets. Lenca et al. [52] proposed a framework of selecting the desired measure based on the properties of interest. In order to provide better guidelines for users to choose the desirable correlation measures according to their needs, we will propose several desirable properties for correlation

measures and study the property satisfaction for different correlation measures in this chapter.

With so many correlation measures, people want to know which one is the best for their applications. For example, Zhang et al. [95] conducted experiments to select the appropriate measure to evaluate the correlation between syndrome elements and symptoms. By studying the literature related to correlation, we notice that different correlation measures are favored in different domains. In text mining area, people use likelihood ratio [30]. BCPNN is favored in the medical domain [7], while leverage is used in the social network context [20]. Our research will examine why different areas favor different measures.

Different from the previous work made by Geng and Hamilton. [37], we only focus on correlation instead of the broader topic of interestingness which includes conciseness, coverage, reliability, peculiarity, diversity, novelty, surprisingness (i.e. correlation), and utility. Different from the previous work of Tan et al. [81], we focus on providing the guidelines for choosing the proper correlation measure according to users' situations instead of categorizing correlation measures. Although this is not a survey, we have to analyze a reasonable number of correlation measures and test them against our proposed guidelines. Therefore, we choose the best or most common correlation measures instead of maintaining a complete list. In addition, we differentiate our search on the corresponding itemset cell from that on the itemset family. Given the itemset family $\{A,B,C\}$, it includes 8 cells, such as cell $\{A,B,C\}$, cell $\{A,B,\bar{C}\}$, and so on. Some measures like leverage [68] and the simplified χ^2-statistic [48] evaluate the correlation corresponding to a cell, but other measures like entropy [83] and χ^2-statistic evaluate the overall correlation of all the cells related to a given itemset family. We are not interested in the search of itemset family for

two reasons. First, it messes up the positive correlation and negative correlation. If A, B, and $\bar{C}$ are positively correlated with each other, the search on itemset family can only tell us the dependence of items A, B, and C, but we don't know whether they are positively correlated. Second, for an itemset S with the size m, we need to calculate the value for 2^m cells. Though there are some smart ways of avoiding redundant calculation [83], it is still computationally expensive for large itemsets.

The remainder of this chapter is organized as follows. In Section 2, important correlation properties and different correlation measures will be discussed. The correlation properties are used to guide the choice of different correlation measures. Section 3 shows the experimental results. Finally, we draw a conclusion in Section 4.

2.2 Correlation Properties

Since some correlation measures can only be used for pairs, we categorize correlation measures into the general and pair-only types. Both types can evaluate correlation for pairs, but only the general type can evaluate correlation for itemsets. There are some general correlation properties that both types need to satisfy. However, there are some additional properties for the pair-only measures.

2.2.1 General Correlation Properties

Given an itemset $S = \{I_1, I_2, ..., I_m\}$, the following seven properties provide the guidelines for a good correlation measure M according to users' needs:

P1: M is equal to a certain constant number C when all the items in the itemset are statistically independent.

P2: M monotonically increases with the increase of $P(S)$ when all the $P(I_i)$ remain the same.

P3: M monotonically decreases with the increase of any $P(I_i)$ when the remaining $P(I_k)$ and $P(S)$ remain unchanged.

P4: The upper bound of M gets closer to the constant C when $P(S)$ is closer to 0.

P5: M gets closer to C (including negative correlation cases) when an independent item is added to S.

P6: The lower bound of M gets closer to the lowest possible function value when $P(S)$ is closer to 0.

P7: M gets further away from C (including negative correlation cases) with increased sample size when all the $P(I_i)$ and $P(S)$ remain unchanged.

The first three properties proposed by Piatetsky-Shapiro [68] are mandatory for any good correlation measure M. The first property requires a constant C to indicate independence when the actual probability is the same as the expected probability. It is positive correlation when above C, or negative correlation when below C. The second property requires the correlation value to increase when the expected probability stays the same while the actual probability goes up. In other words, it deserves more credit when we have the same expectation but the actual performance is better. Similarly, the correlation value will decrease when the expected probability goes up while the actual probability stays the same.

The above three mandatory properties screen out some bad correlation measures, but there are still some other bad correlation measures which satisfy all the three mandatory properties. In order to solve the problem, we proposed another two desired properties [26] which are the fourth and fifth properties. The fourth property means it is impossible to find any strong positive correlation from itemsets occurring

rarely. In other words, the itemset at least has to happen several times in order to be statistically valid. We want to find the significant patterns rather than coincidences. For the fifth property, we give some penalty for adding independent items in order to make highly-correlated itemsets stand out. In the extreme case, when a lot of independent items are added, the final itemset is dominated by the independence and the correlation value should be very close to the constant C.

In addition, we proposed two optional properties which are the sixth and seventh properties. The sixth property expects the strongest negatively correlated itemsets to come from the low Support region. It is quite intuitive since the stronger negative correlation means lower chance of happening together. The fourth property checks whether correlation measures can correctly evaluate positive correlation while the sixth property checks the correct evaluation for negative correlation. Therefore, if users are only interested in positive correlation, it doesn't matter if the correlation measure cannot satisfy the sixth property. Similarly, if users are only interested in negative correlation, it doesn't matter if the correlation measure cannot satisfy the fourth property. However, we treat the fourth property as desired and the sixth property as optional for the following reason. If we consider the absence of an item I as the presence of the absence, we can find the positive correlation like $\{A, \bar{B}, C\}$ and we understand how A, B, and C are correlated with each other. From the negative correlation $\{A, B, C\}$, we don't know how A, B, and C are correlated with each other. The seventh property indicates the correlation measure should increase our confidence about the positive or negative correlation of the given itemset S when we get more sample data from the same population. However, we prefer it to be optional for two reasons. First, we can always make our correlation measures a function of the sample size isolated from other parameters. In that way, we can either keep the

sample size parameter to satisfy the last property or drop the sample size parameter. Second, we might want to compare the correlation from different sources which have different sample sizes. In order to conduct a fair comparison, we might not want the correlation measure to satisfy the last property.

	B	$\bar{B}$	$\sum$ Row
A	f_{11}	f_{10}	f_{1+}
$\bar{A}$	f_{01}	f_{00}	f_{0+}
$\sum$ Column	f_{+1}	f_{+0}	N

Table 2.3: A 2-way contingency table for variables A and B

2.2.2 Additional Properties for Pair-only Measures

In addition to the above 7 properties for both general and pair-only measures, Tan et al. [82] proposed 3 additional properties for the pair-only type measures based on operations for 2×2 contingency tables. Table 2.3 shows a typical 2×2 contingency table for a pair of binary variables, A and B. Each entry f_{ij} in this 2×2 tables denotes a frequency count.

The three proposed properties on a 2 contingency table are as follows:

AP1: M remains the same when exchanging the frequency count f_{11} with f_{00} and f_{10} with f_{01}.

AP2: M remains the same by only increasing f_{00}.

AP3: M remains the same under the row/column scaling operation from Table T to T', where T is a contingency table with frequency counts $[f_{11}; f_{10}; f_{01}; f_{00}]$, T' is a contingency table with scaled frequency counts $[k_1k_3f_{11}; k_2k_3f_{10}; k_1k_4f_{01};$

$k_2k_4f_{00}]$, and k_1, k_2, k_3, k_4 are positive constants.

The first property is the inversion property which argues that the correlation of A and B should be the same as that of $\bar{A}$ and $\bar{B}$. The second property is the null addition property. Tan et al. [81] argued that the correlation between two items should not be affected by adding unrelated data to a given data set. For example, we are interested in analyzing the relationship between a pair of words, such as *data* and *mining*, in a set of computer science papers. The association between *data* and *mining* should not be affected by adding a collection of articles about ice fishing. The third property is the scaling property. Mosteller [59] presented the following example to illustrate the scaling property. Tables 2.4 and 2.5 show the contingency tables for gender and the grades achieved by students enrolled in a particular course in 1993 and 2004 respectively. The data in these tables showed that the number of male students has doubled since 1993, while the number of female students has increased by a factor of 3. However, the male students in 2004 are not performing any better than those in 1993 because the ratio of male students who achieve a high grade to those who achieve a low grade is still the same, i.e., 3:4. Similarly, the female students in 2004 are performing the same as those in 1993.

	Male	Female	$\sum$ Row
High	30	20	50
Low	40	10	50
$\sum$ Column	70	30	100

Table 2.4: The grade-gender example from 1993

However, the purpose of these three properties proposed by Tan et al. [81] is

	Male	Female	$\sum$ Row
High	60	60	120
Low	80	30	110
$\sum$ Column	140	90	230

Table 2.5: The grade-gender example from 2004

to categorize correlation measures. They don't provide any guideline for choosing the proper correlation measure according to users' situation. In fact, we argue these three properties are not desirable correlation properties.

For the first inversion property, we argue the correlation of A and B is determined by f_{11} or the other three occurrences f_{01}, f_{10}, and f_{00} instead of f_{00} alone. In Table 2.6, we can fix the true probability and the expected probability of $A \cap B$, but alternate the values of f_{01}, f_{10}, and f_{00} to generate another Table 2.7. Since the true probability and the expected probability of A and B are the same in these two tables, the correlation of A and B in these two tables is the same. If the inversion property stands, we can conclude that the correlation of $\bar{A}$ and $\bar{B}$ in Table 2.6 is equal to the correlation of $\bar{A}$ and $\bar{B}$ in Table 2.7. However, it is controversial because the co-occurrence of $\bar{A}$ and $\bar{B}$ is different.

	B	$\bar{B}$	$\sum$ Row
A	5	4	9
$\bar{A}$	11	80	91
$\sum$ Column	16	84	100

Table 2.6: The original table

For the second null addition property, the correlation of A and B should be the

	B	$\bar{B}$	$\sum$ Row
A	5	7	12
$\bar{A}$	7	81	88
$\sum$ Column	12	88	100

Table 2.7: The modified table

same if only we fix the values of f_{11}, f_{01}, and f_{10}. Given the extreme example that we add a huge number of ice fishing documents into the set of computer science papers, the ice fishing documents start to dominate the corpus and the set of computer science papers becomes the background noise. Although the co-occurrence of the pair of words "data" and "mining" is not changed, intuitively, we don't want the correlation between "data" and "mining" to be as strong as that in the initial setting since it is the background noise now. We can also analyze this case in another way. The actual probability of A and B is $f_{11}/(f_{11}+f_{01}+f_{10}+f_{00})$ and the expected probability of A and B is $(f_{11}+f_{10})/(f_{11}+f_{01}+f_{10}+f_{00})\cdot(f_{11}+f_{01})/(f_{11}+f_{01}+f_{10}+f_{00})$. For the pair A and B, both the actual probability and the expected probability decreases when f_{00} increases. The decrease of the actual probability lowers the correlation according to Property 2, and the decrease of the expected probability increases the correlation according to Property 3. The final change of correlation is due to the tradeoff between the effect from the actual probability and that from the expected probability, and it is unnecessary for these to be the same. When we add a huge number of ice fishing documents into the set of computer science papers, the Support of the pair of words "data" and "mining" is close to 0. The correlation of the pair of words "data" and "mining" should also be close to the constant number C according to Property 4, which also contradicts the null addition property.

For the third scaling property, let's reconsider the gender-grade example shown

in Table 2.4 and 2.5. Though the ratio of male students who achieve a high grade to those who achieve a low grade in 1993 is still the same as that of 2004, the ratio of male students who achieve a high grade to those who achieve a low grade is different from that of females. Since the portion of the male students has changed from 1993 to 2004, we are expecting the high-grade students are less likely to be male in 2004. The correlation between grade and gender should be changed.

Though we doubt the three addition properties qualify as desirable properties and the experimental results in this chapter support our arguments, it is still up to users' choice. Besides the three properties proposed by Tan et al., Geng et al. [37] also proposed two properties for pair correlation measures as follows:

AP4: M should be an increasing function of Support if the margins in the contingency table are fixed.

AP5: M should be an increasing function of Confidence if the margins in the contingency table are fixed.

However, the increase of Support must cause the increase of Confidence if the margins in the contingency table are fixed. These two rules are talking about the same idea, and they are also rephrase Property 2 in a different way. Therefore, we combine these two properties into Property 2.

2.3 Formulas and Property Satisfaction

In this section, we study a collection of popular correlation measures for both the general and the pair-only types, and their property satisfaction.

2.3.1 General Correlation Measures

Given an itemset $S = \{I_1, I_2, ..., I_m\}$ with m items in the dataset with the sample size n, the true probability is $tp = P(S)$, the expected probability is $ep = \prod_{i=1}^{m} P(I_i)$, and the occurrence is $k = P(S) \cdot n$. The above parameters will be used in different general correlation measures, and we use "Support" and "true probability" interchangeably in this thesis.

2.3.1.1 Support

Support of the itemset S is the proportion of transactions that contain S. Using Support, the level of correlation is simply the fraction of times that the items co-occur. As a metric, Support facilitates fast search, but it has drawbacks [14, 15]. For example, the finding that A and B occur together in 81% of the transactions is not interesting if A and B both occur in 90% of the transactions. This would be expected since $P(A) = P(A|B)$ and $P(B) = P(B|A)$. Most simple statistical tests over A and B would reflect this lack of true correlation between A and B, even though A and B together have very high Support. Among all the seven properties mentioned above, Support only satisfies Property 2 and 6 [1]. Since even two mandatory properties are violated by Support, it is a poor correlation measure.

2.3.1.2 Any-confidence

Any-confidence [66] of the itemset S is the ratio of its probability to the probability of the item with the lowest probability in S: $AnyConfidence(S) = P(S)/min(P(I_1), P(I_2), ..., P(I_m))$. The value of Any-confidence is the upper bound of the confidence of all association rules which can be generated from the itemset S.

[1] The property satisfaction proofs related to each correlation measure are in the appendix.

It helps us to determine whether we can find at least one rule which has a confidence greater than the specified threshold. However, it is not designed as a correlation measure, and does not have a downward closure property [2] to facilitate search. A property ρ is downward-closed if for every set with property ρ, all its subsets also have property ρ.

2.3.1.3 All-confidence

All-confidence [66] of the itemset S is the ratio of its probability to the probability of the item with the highest probability in S: $AllConfidence(S) = P(S)/max(P(I_1), P(I_2), ..., P(I_m))$. The value of All-confidence is the lower bound of the confidence of all association rules which can be generated from the itemset S. Although All-confidence itself possesses the downward closure property to facilitate search, it is not designed as correlation measure and suffers the same problems as Support. Theoretically, All-confidence lacks comparison to the expected probability under the independence assumption. Like Support, it satisfies only the second and the sixth of the seven desired correlation measure properties. Practically, All-confidence shares three problems with Support. First, it is biased toward itemsets with high-Support items. If an itemset S consists of independent, high-Support items, $Support(S)$ will be high (despite the independence), and $AllConfidence(S)$ will also be high. This problem is exaggerated if we extend our search to include the presence of some items and the absence of others, since absence of a rare item is itself a high-Support item. This is typically not relevant in marketing but could be in, e.g., genetic data. Second, intuitively we want exact-length correlation patterns. However, All-confidence is biased to short itemsets as its value decreases monotonically with increasing itemset size. More maximal All-confidence sets are likely to be 2-itemsets

like maximal frequent itemsets. Third, the anti-monotone property makes it difficult to compare correlation among itemsets of different sizes.

2.3.1.4 Bond/Jaccard

Bond [66] of the itemset S is the ratio of its probability to the probability of the union of all the items in S: $Bond(S) = P(S)/P(I_1 \cup I_2 \cup ... \cup I_m)$. Normally, Jaccard is used for pairs and Bond is used for itemsets, but they share the same idea. Bond is similar to Support but with respect to a related subset of the data rather than the entire data set. Like Support and All-confidence, Bond possesses the downward closure property. Given a set of strongly related rare items, both Bond and All-confidence can assign a high score for this itemset which can relieve the disadvantage of Support. However, worse than All-confidence, Bond satisfies only the sixth of the seven desired correlation measure properties, and measures correlation in a sub-optimal way.

2.3.1.5 The Simplified χ^2-statistic

The χ^2 is calculated as $\chi^2 = \sum_i \sum_j (r_{ij} - E(r_{ij}))^2/E(r_{ij})$. If an itemset contains n items, 2^n cells in the contingency table must be considered for the above Pearson χ^2 statistic. The computation of the statistic itself is intractable for high-dimensional data. However, we can still use the basic idea behind the χ^2-statistic to create the Simplified χ^2-statistic: $\chi'^2 = (r - E(r))^2/E(r)$, i.e., $n \cdot (tp - ep)^2/ep$, where the cell r corresponds to the exact itemset S. Since the Simplified χ^2-statistic is more computationally desirable, in the rest of thesis we only discuss the properties and experimental results of the Simplified χ^2-statistic. The value of the Simplified χ^2-statistic is always larger than 0 and cannot differentiate positive from negative

correlation. Therefore, we take advantage of the comparison between tp and ep. If $tp > ep$, it is a positive correlation. Then the Simplified χ^2-statistic is equal to $n \cdot (tp - ep)^2/ep$. If $tp < ep$, it is a negative correlation. Then the Simplified χ^2-statistic is equal to $-n \cdot (tp - ep)^2/ep$. This transformed Simplified χ^2-statistic is mathematically favorable. Larger positive numbers indicate stronger positive correlation, 0 indicates no correlation, and larger (in magnitude) negative numbers indicate stronger negative correlation.

However, even if we can solve the computational problem for a given itemset, the performance of the χ^2-statistic (including the Simplified χ^2-statistic) for measuring correlation within the itemset framework is still very doubtful. The problem with this correlation measure stems from the fact that each possible event should be expected to occur at least five times for the χ^2 test of independence to be valid [36]. This requirement is unrealistic in many data mining domains. For example (from [48]), in market basket applications, it is practical to anticipate the order of 10^5 items (or more) that may be purchased. Suppose each basket contains 20 items on average, and we want to apply the χ^2 test to an arbitrary three-itemset. In the best case (without a skewed distribution of items purchased), to ensure that we would expect to find at least one such three-itemset in the database, we would need a database with at least $((10^5)^3)/20 \approx 10^{14}$ transactions. That is, we require that every person on Earth purchases tens of thousands of baskets. The situation deteriorates exponentially when we move from three-way correlations to four-way, five-way, and higher-degree correlations. Dunning [30] shows how the application of the χ^2 test to the domain of co-occurrence analysis in text can produce poor results. The domain is closely related to the market basket domain where we have a relatively large number of possible "items" (words in the English language) and a database which is not large enough to

validate the test.

2.3.1.6 Probability Ratio/Lift/Interest Factor

Probability Ratio [15] is the ratio of its actual probability to its expected probability under the assumption of independence. It is calculated as follows: $PR(S) = tp/ep$. This measure is straightforward and means how many times the itemset S happens more than expected. In some cases, we also use the log value of Probability Ratio. In that way, we make the constant number C in the first mandatory property 0, which is consistent with other measures. However, this measure might not still be a reasonable correlation measure to use. The problem is that it favors the itemsets containing a large number of items rather than significant trends in the data. For example, given a common transaction containing 30 items and each item in this transaction has 50% chance to be bought individually, the expected probability for this transaction is $9.31 * 10^{-10}$ if all the items are independent. Even if this transaction coincidentally happened once out of one million transactions, its Probability Ratio is 1073 which is still very high. However, a single transaction is hardly something that we are interested in.

2.3.1.7 Leverage

An itemset S with higher Support and low Probability Ratio may be more interesting than an alternative itemset S' with low Support and high Probability Ratio. Introduced by Piatesky-Shapiro [68], $Leverage(S) = tp - ep$. It measures the difference between the actual probability of an itemset S and its expected probability if all the items in S are independent from each other. Since $\prod_{i=1}^{m} P(I_i)$ is always no less than 0, $Leverage(S)$ can never be bigger than $P(S)$. Therefore, Leverage is

biased to high-Support itemsets.

2.3.1.8 Likelihood Ratio

Likelihood Ratio is similar to a statistical test based on the loglikelihood ratio described by Dunning [30]. The concept of a likelihood measure can be used to statistically test a given hypothesis, by applying the likelihood ratio test. Essentially, we take the ratio of the highest likelihood possible given our hypothesis to the likelihood of the best "explanation" overall. The greater the value of the ratio, the stronger our hypothesis will be.

To apply the likelihood ratio test as a correlation measure, it is useful to consider the binomial distribution. This is a function of three variables: $Pr(p, k, n) \rightarrow [0 : 1]$. Given our assumption of independence of all items in the itemset S, we predict that each trial has a probability of success ep. Then the binomial likelihood of observing k out of n transactions containing S is $Pr(ep, k, n)$. However, the best possible explanation of the probability of containing S is tp instead of ep. Therefore, we perform the Likelihood Ratio test, comparing the binomial likelihood of observing k transactions under the assumption of independence with the best possible binomial explanation. Formally, the Likelihood Ratio in this case is $LikelihoodRatio(S) = Pr(tp, k, n)/Pr(ep, k, n)$.

In the rest of the thesis, we use a transformed Likelihood Ratio to measure correlation for two reasons. First, since the actual Likelihood Ratio could be extremely large, we use the ln value instead of its original value. Second, the numerator of the Likelihood Ratio is the maximal likelihood of the real situation, so the Likelihood Ratio is always larger than 1 and cannot differentiate positive from negative correlation. When calculating the transformed Likelihood Ratio, we take advantage of the compar-

ison between tp and ep. If $tp > ep$, it is a positive correlation. Then the transformed Likelihood Ratio is equal to $ln(LikelihoodRatio(S))$. If $tp < ep$, it is a negative correlation. Then the transformed Likelihood Ratio is equal to $-ln(LikelihoodRatio(S))$. This transformed Likelihood Ratio is mathematically favorable. Larger positive numbers indicate stronger positive correlation, 0 indicates no correlation, and larger (in magnitude) negative numbers indicate stronger negative correlation.

The Likelihood Ratio strikes a balance between the Probability Ratio and the actual occurrence k. It favors itemsets with both high Probability Ratio and high occurrence. For the itemsets containing a small number of items, their occurrence tends to be high, but the Probability Ratio tends to be low, while, for the itemsets containing a large number of items, their Probability Ratio tends to be high, but the actual occurrence tends to be low. Likelihood Ratio favors the middle size itemsets which can strike a balance between the Probability Ratio and the actual occurrence.

2.3.1.9 BCPNN

Probability Ratio is straightforward and means how many times the combination happens more than expected. However, the Probability Ratio is very volatile when the expected value is small, which makes it favor coincidences rather than significant trends in the data. In order to solve the problem, we use shrinkage [7, 29, 65] to regularize and reduce the volatility of a measure by trading a bias to no correlation for decreased variance. For an itemset S, the calculated tp is 0 if S is not observed in the dataset. However, we might get some transactions containing S if we get more samples. In order to make the conservative estimation to the ground tp and ep, we add a continuity correction number here. Suppose the continuity correction is cc, the formula of BCPNN is $BCPNN = ln(tp + cc)/(ep + cc)$. Normally, we set $cc = 0.5/n$

when the data set is relatively clean; however, it could be any positive number theoretically. Especially when the data set contains a lot of noisy data, we might use larger number to make more conservative estimation. This shrinkage strength has been successfully applied to pattern discovery in the analysis of large collections of individual case safety reports. Noren et al. [65] claimed that it precludes highlighting any pattern based on less than three events but is still able to find strongly correlated rare patterns. In general, the strength and direction of the shrinkage can be adjusted by altering the magnitude and ratio of the constants added to the nominator and denominator, which will be fully discussed in Section 2.4. From a frequency perspective, BCPNN is a conservative version of Probability Ratio, tending towards 0 for rare events and with better variance properties. As *tp* and *ep* increase, the impact of the shrinkage diminishes.

Lemma 2.1 (Variance Convergence). *Given the true probability p for an itemset S in the dataset with n transactions, the variance of Probability Ratio approaches ∞ and the variance of BCPNN approaches 0 when $p \to 0$.*

Proof. Since the occurrence of S, denoted by X, follows the binomial distribution, we get $E(X) = n \cdot p$ and $Var(X) = n \cdot p \cdot (1-p)$.

(1) $ProbabilityRatio = X/(n{\cdot}p)$. Therefore, $Var(ProbabilityRatio) = Var(X)/(n^2{\cdot}p^2) = (1-p)/(n{\cdot}p)$. According to the formula, we can see that $Var(ProbabilityRatio) \to \infty$ when $p \to 0$.

(2) $BCPNN = (X + cc)/(n \cdot p + cc)$. Therefore, $Var(BCPNN) = Var(X + cc)/(n{\cdot}p{+}cc)^2 = (n{\cdot}p{-}n{\cdot}p^2)/(n^2{\cdot}p^2{+}2{\cdot}cc{\cdot}n{\cdot}p{+}cc^2)$. As $p \to 0$, $Var(BCPNN) \to 0/cc^2 = 0$

2.3.1.10 Simplified χ^2 with Continuity Correction

Inspired by the shrinkage technique applied to BCPNN, we propose a new correlation measure, Simplified χ^2 with Continuity Correction (SCWCC). Suppose the continuity correction is cc, we add cc additional occurrences to both the actual events and the expected events. The formula of SCWCC is $SCWCC = n \cdot (tp - ep)^2/(ep + cc)$. As tp gets closer to 0, the upper bounds of Likelihood Ratio, Leverage, BCPNN and SCWCC get closer to the constant number C. However, different measures have different biases towards different Support regions which will be discussed in Section 2.4.

2.3.1.11 Interest Factor with Support

Since interest factor (Probability Ratio) favors the rare combinations rather than significant trends in the data and Support favors frequent combinations rather than strong correlation, Tan et al. [64] proposed Interest factor with Support (IS) which is the square root of the product of Interest factor and Support, i.e. $IS(S) = \sqrt{ProbabilityRatio(S) \cdot Support(S)} = tp/\sqrt{ep}$. When measuring pairs for binary data, IS is exactly cosine similarity, which is $n \cdot P(A \cap B)/\sqrt{n \cdot P(A) \cdot n \cdot P(B)} = tp/\sqrt{ep}$. Therefore, we treat the cosine similarity as a special case of IS when measuring binary data. Intuitively, IS is large when both Probability Ratio and Support are large enough. In fact, the IS value of a rare large combination is still very large which is not that much better than Probability Ratio. In addition, when all the items in the itemset S are independent with each other, i.e. $tp = ep$, $IS = \sqrt{ep}$ which is not constant. It violates the first mandatory property.

2.3.1.12 Two-way Support/The Simplified Mutual Information

Sharing the same idea with IS, Zhong et al. [97] proposed Two-way Support measure which is the product of Support and the log value of Probability Ratio, i.e. $TwoWaySupport(S) = tp \cdot ln(tp/ep)$. It borrows the idea of mutual information [32] to measure the correlation for the target cell. Better than IS, Two-way Support satisfies the first mandatory property and uses the log value of Probability Ratio to suppress the increase of Probability Ratio. As Support approaches 0, the decrease from Support dominates the increase from the log value of Probability Ratio, i.e., its upper bound is close to 0. However, the side effect is that its lower bound also approaches 0 when Support is close to 0. In other words, there are no significant negatively correlated patterns for low Support itemsets, which is wrong.

2.3.1.13 Simplified χ^2 with Support

Both Simplified χ^2 and Probability Ratio favor rare combinations rather than significant trends in the data. Inspired by the IS measure, we propose a new correlation measure called Simplified χ^2 with Support (SCWS), which is the product of Simplified χ^2 and Support. The formula of SCWS is $SCWS(S) = tp \cdot (tp - ep)^2/ep$. Better than IS measure, SCWS satisfies the first mandatory property. However, the same as IS measure, the SCWS value of a rare large combination is still very large. In addition, the same as Two-way Support, the SCWS value of the negatively correlated itemset gets closer to the constant number C when the Support of this itemset gets closer to 0, which is not qualified for the negative correlation search.

2.3.2 Pair-only Correlation Measures

Given the typical 2×2 contingency table for a pair of binary variables in Table 2.3, the commonly-used pair-only type correlation measures are calculated as follows.

2.3.2.1 ϕ Correlation Coefficient

The ϕ Correlation Coefficient [73] is derived from Pearson's Correlation Coefficient for binary variables. The formula of the ϕ Correlation Coefficient is as follows: $(f_{00}f_{11} - f_{01}f_{10})/\sqrt{f_{0+}f_{1+}f_{+0}f_{+1}}$. It measures the linear relationship between two binary variables.

2.3.2.2 Relative Risk

Relative Risk [77] is the ratio of the probability of the event occurring in the exposed group versus a non-exposed group. It is often used to compare the risk of developing a side effect, in people receiving a drug versus people not receiving the treatment. Given Table 2.3, the Relative Risk for the event B within the two situations defined by A and $\bar{A}$ is $\frac{f_{11}/f_{1+}}{f_{01}/f_{0+}}$.

2.3.2.3 Odds Ratio

The Odds Ratio [59] is a measure of effect size, describing the strength of non-independence between two binary variables and comparing them symmetrically. It plays an important role in logistic regression. The Odds Ratio is the ratio of the odds of an event occurring in one group to the odds of it occurring in another group. In Table 2.3, the odds for B within the two subpopulations defined by A and $\bar{A}$ are defined in the terms of the conditional probabilities in Table 2.8. Thus the Odds Ratio is $(\frac{f_{11}/f_{1+}}{f_{10}/f_{1+}})/(\frac{f_{01}/f_{0+}}{f_{00}/f_{0+}}) = \frac{f_{11}*f_{00}}{f_{10}*f_{01}}$. The final expression is easy to remember as the product of the concordant cells ($A = B$) divided by the product of the discord

cells ($A \neq B$). Since Relative Risk is a more intuitive measure of effectiveness, the distinction is important especially in cases of medium to high probabilities. If action A carries a risk of 99.9% and action B a risk of 99.0% then the Relative Risk is just over 1, while the odds associated with action A are almost 10 times higher than the odds with B. In medical research, the Odds Ratio is favored for case-control studies and retrospective studies. Relative Risk is used in randomized controlled trials and cohort studies.

	B	$\bar{B}$
A	f_{11}/f_{1+}	f_{10}/f_{1+}
$\bar{A}$	f_{01}/f_{0+}	f_{00}/f_{0+}

Table 2.8: The conditional probability table for variables A and B

2.3.2.4 Conviction

Conviction [15] is calculated as $f_{1+} * f_{+0}/f_{10}$. Logically, $A \rightarrow B$ can be rewritten as $\neg(A \wedge \neg B)$. Similar to Lift, $f_{11}/(f_{1+} * f_{+1})$, which seeks the deviation from independence between A and B, Conviction exams how far $A \wedge \neg B$ deviates from independence. In other words, Conviction looks for the correlation underlying the rule $A \rightarrow B$ instead of the pair A and B.

2.3.2.5 Added Value

The same as Conviction, Added Value [98] is a measure for rules instead of pairs. Given the rule $A \rightarrow B$, Added Value measures the difference between $Support(B)$ in the whole population and that in the population A. Specifically,

$AddedValue(A \rightarrow B) = P(B|A) - P(B)$. If we transform the formula, we get $AddedValue(A \rightarrow B) = (P(A \wedge B) - P(A) \cdot P(B))/P(A) = Leverage(A, B)/P(A)$. It is Leverage tuned by $P(A)$. When $P(A)$ is small, $Leverage(A, B)$ will also be small. Added Value tries to divide Leverage by $P(A)$ to prompt the correlated pattern in a small population.

2.3.3 Summary of Correlation Measures

We have categorized both general and pair-only correlation measures. The general type can be further divided into three sub-categories: sub-optimal measures, basic measures, and adjusted measures. Support, Any-confidence, All-confidence, and Bond are the sub-optimal measures. All of them violate more than one mandatory correlation property. However, most of them have the downward closed property to facilitate the search. The basic correlation measures, derived from simple statistical theories, include Simplified χ^2, Probability Ratio, Leverage, and Likelihood Ratio. They satisfy all three mandatory properties, but don't possess the downward closed property. In addition, they might violate some desirable correlation properties. The measures adjusted by continuity correction are BCPNN and Simplified χ^2 with Continuity Correction. They use the shrinkage technique to reduce the volatility of a measure by trading a bias to no correlation for decreased variance. In this way, we modify the basic correlation measures to satisfy all the desirable properties. The measures adjusted by Support include IS, Two-way Support, and SCWS. They try to adjust the basic measures by multiplying Support to suppress the increase from correlation measures when Support is close to 0. Table 2.9 shows the original formulas of measures, and Table 2.10 is a summary of the original version measures with regard to all ten properties.

Correlation Measure	Formula
Support	tp
Any-confidence	$\frac{tp}{min(P(I_1),P(I_2),...,P(I_m))}$
All-confidence	$\frac{tp}{max(P(I_1),P(I_2),...,P(I_m))}$
Bond	$\frac{tp}{P(I_1 \cup I_2 \cup ... \cup I_m)}$
Simplified χ^2-statistic	$n \cdot \frac{(tp-ep)^2}{ep}$
Probability Ratio	$ln\ \frac{tp}{ep}$
Leverage	$tp - ep$
Likelihood Ratio	$n \cdot [tp \cdot ln\ \frac{tp}{ep} + (1-tp) \cdot ln\ \frac{1-tp}{1-ep}\ ]$
BCPNN	$ln\ \frac{tp+cc}{ep+cc}$
SCWCC	$n \cdot \frac{(tp-ep)^2}{ep+cc}$
IS	$\frac{tp}{\sqrt{ep}}$
Two-way Support	$tp \cdot ln\ \frac{tp}{ep}$
SCWS	$n \cdot tp \cdot \frac{(tp-ep)^2}{ep}$
ϕ-coefficient	$\frac{f_{00} \cdot f_{11} - f_{10} \cdot f_{01}}{\sqrt{f_{1+} \cdot f_{0+} \cdot f_{+1} \cdot f_{+0}}}$
Relative Risk	$\frac{f_{11}/f_{1+}}{f_{01}/f_{0+}}$
Odds Ratio	$\frac{f_{11} \cdot f_{00}}{f_{10} \cdot f_{01}}$
Conviction	$\frac{f_{1+} \cdot f_{+0}}{n \cdot f_{10}}$
Added Value	$\frac{n \cdot f_{11} - f_{1+} \cdot f_{+1}}{n \cdot f_{1+}}$

Table 2.9: Formulas of correlation measures

Correlation Measure	P1	P2	P3	P4	P5	P6	P7	AP1	AP2	AP3
Support		X				X				
Any-confidence		X				X			X	
All-confidence		X				X			X	
Bond						X			X	
Simplified χ^2-statistic	X	X	X		X	X	X			
Probability Ratio	X	X	X			X				
Leverage	X	X	X	X	X	X		X		
Likelihood Ratio	X	X	X	X	X	X	X			
BCPNN	X	X	X	X	X	X				
SCWCC	X	X	X	X	X	X	X			
IS		X	X			X			X	
Two-way Support	X	X	X	X	X					
SCWS	X	X	X		X		X			
ϕ-coefficient	X	X	X			X		X		
Relative Risk	X	X	X			X				
Odds Ratio	X	X	X			X		X		X
Conviction	X	X	X			X				
Added Value	X	X	X			X				

Table 2.10: Properties of correlation measures

2.4 The Upper and Lower Bounds of Measures

Among 18 correlation measures we study, the three mandatory properties proposed by Piatetsky-Shapiro screen out 5 measures. The two desired properties proposed by us together with the three mandatory properties can narrow down the candidate list to 5 measures: Leverage, Likelihood Ratio, BCPNN, Two-way Support, and SCWCC. Since the candidate list still has 5 measures, a natural question is whether they achieve the same results. In order to find the differences, we study the upper and lower bounds of different measures when tp is fixed, and discuss the trade-offs between Support and itemset size in this section.

2.4.1 Upper and Lower Bound Analysis

The following theorem is used to analyze the upper and lower bounds.

Theorem 2.2. *Given an itemset $S = \{I_1, I_2, ..., I_m\}$ with the actual probability tp, its expected probability ep is no less than tp^m and no more than $((m-1+tp)/m)^m$.*

Proof. (1) According to definition, $ep = \prod_{i=1}^{m} P(I_i = 1)$. For each item I_i in S, $tp \leq P(I_i = 1) \leq 1$. When the actual probability of each item I_i reaches the lower bound tp and all the items occur together, the expected probability ep reaches its lower bound tp^m.

(2) Given the itemset $\{I_1, I_2, ..., I_m\}$, we have

$$\sum_{I_1=0}^{I_1=1} \sum_{I_2=0}^{I_2=1} ... \sum_{I_m=0}^{I_m=1} P(I_1, I_2, ..., I_m) = 1. \tag{2.1}$$

and the Support for each item I_i is

$$P(I_i = 1) = \sum_{I_1=0}^{I_1=1} ... \sum_{I_{i-1}=0}^{I_{i-1}=1} \sum_{I_{i+1}=0}^{I_{i+1}=1} ... \sum_{I_m=0}^{I_m=1} P(I_1, ..., I_{i-1}, I_i = 1, I_{i+1}, ..., I_m).$$

Given the cell $\{I_i = 1, I_1, I_2, ..., I_p = 0, ..., I_q = 0, ..., I_m\}$ with more than two items having value 0, if its probability is greater than 0, we can decrease its probability to 0 and increase the probability of the cell $\{I_i = 1, I_1, I_2, ..., I_p = 1, ..., I_q = 0, ..., I_m\}$ (or $\{I_i = 1, I_1, I_2, ..., I_p = 0, ..., I_q = 1, ..., I_m\}$). By doing that, we keep $P(I_i = 1)$ the same but increase $P(I_p = 1)$ (or $P(I_q = 1)$).

Since $ep = \prod_{i=1}^{m} P(I_i = 1)$, ep can be increased by adjusting the probability of the cell with more than two absent items to 0. Therefore, in order to get the maximal ep, we can simplify Equation 2.1 to

$$P(I_1 = 1, I_2 = 1, ..., I_m = 1) + \sum_{i=1}^{m} P(I_1 = 1, ..., I_{i-1} = 1, I_i = 0, I_{i+1} = 1..., I_m = 1) = 1.$$

Since we know $P(I_1 = 1, I_2 = 1, ..., I_m = 1) = tp$, then

$$\sum_{i=1}^{m} P(I_1 = 1, ..., I_{i-1} = 1, I_i = 0, I_{i+1} = 1..., I_m = 1) = 1 - tp.$$

Therefore, we have

$$P(I_i = 1) = 1 - P(I_1 = 1, I_2 = 1, ..., I_{i-1} = 1, I_i = 0, I_{i+1} = 1..., I_m = 1)$$

and

$$\begin{aligned} \sum_{i=1}^{m} P(I_i = 1) &= m - \sum_{i=1}^{m} P(I_1 = 1, ..., I_{i-1} = 1, I_i = 0, I_{i+1} = 1..., I_m = 1) \\ &= m - 1 + tp. \end{aligned}$$

In order to get the maximal $ep = \prod_{i=1}^{m} P(I_i = 1)$ when $\sum_{i=1}^{m} P(I_i = 1) = m - 1 + tp$, we have $P(I_1 = 1) = P(I_2 = 1) = ... = P(I_m = 1) = (m - 1 + tp)/m$. Therefore, the upper bound of ep is $((m - 1 + tp)/m)^m$.

2.4.1.1 Support

Since $Support(S) = tp$, both the upper bound and the lower bound of $Support(S)$ is tp.

2.4.1.2 Any-confidence

Since $tp \leq P(I_i) \leq 1$ for each item I_i in S, the minimal value of $min(P(\{I_i | I_i \in S\}))$ is tp. Suppose the maximal value of $min(P(\{I_i | I_i \in S\}))$ is x, then we need to find the maximal x which satisfies $P(I_1) \geq x$, $P(I_2) \geq x$, ..., and $P(I_m) \geq x$. In order to maintain the value of tp, x is the maximal value that all the $P(I_i)$ can reach simultaneously. According to Theorem 2.2, we get $x = (m - 1 + tp)/m$. Therefore, the upper bound of Any-confidence(S) is $tp/tp = 1$ and the lower bound is $tp/((m - 1 + tp)/m) = m \cdot tp/(m - 1 + tp)$.

2.4.1.3 All-confidence

Since $tp \leq P(I_i) \leq 1$ for each item I_i in S and $P(I_1 \cap I_2 \cap ... \cap I_m) = tp$ holds when each $P(I_i) = tp$, the minimal value of $max(P(\{I_i | I_i \in S\}))$ is tp when each

$P(I_i) = tp$. When a certain $P(I_i) = 1$, it is still possible for $P(I_1 \cap I_2 \cap ... \cap I_m) = tp$. Therefore, the maximal value of $max(P(\{I_i | I_i \in S\}))$ is 1. Then, the upper bound of All-confidence(S) is $tp/tp = 1$ and the lower bound is $tp/1 = tp$.

2.4.1.4 Bond

Since $tp \leq P(I_i) \leq 1$ for each item I_i in S and $P(I_1 \cap I_2 \cap ... \cap I_m) = tp$ holds when each $P(I_i) = tp$, the minimal value of $P(I_1 \cup I_2 \cup ... \cup I_m)$ is tp when each $P(I_i) = tp$. When a certain $P(I_i) = 1$, it is still possible for $P(I_1 \cap I_2 \cap ... \cap I_m) = tp$. Therefore, the maximal value of $P(I_1 \cup I_2 \cup ... \cup I_m)$ is 1. Then, the upper bound of Bond(S) is $tp/tp = 1$ and the lower bound is $tp/1 = tp$.

2.4.1.5 Correlation Measures satisfying Property 3

In the following, we study the upper and lower bounds of the correlation measures satisfying Property 3.

Theorem 2.3. *The lower bound of ep, tp^m, is no more than tp and its upper bound $((m - 1 + tp)/m)^m$ is no less than tp.*

Proof. (a) Since $0 \leq tp \leq 1$ and m is a positive integer larger than 1, $tp^m \leq tp$.

(b) Let $f(tp) = ((m-1+tp)/m)^m - tp$ be a function of tp, then we have $f'(tp) = ((m-1+tp)/m)^{(m-1)} - 1$. Since $0 \leq tp \leq 1$, we have $(m-1)/m \leq (m-1+tp)/m \leq m/m$. Therefore, $((m - 1 + tp)/m)^{(m-1)} \leq 1$ and $f'(tp) \leq 0$. Thus, $f(tp) \geq f(1) = 0$. We get $((m - 1 + tp)/m)^m - tp \geq 0$, i.e., $((m - 1 + tp)/m)^m \geq tp$.

Theorem 2.4. *Given any correlation measure which satisfies Property 3 and an itemset $S = \{I_1, I_2, ..., I_m\}$ with fixed tp, the correlation measure reaches its upper bound when $ep = tp^m$, and reaches its lower bound when $ep = ((m - 1 + tp)/m)^m$.*

Proof. Given any correlation measure which satisfies Property 3, its correlation value should monotonically decrease with the increase of ep when tp is fixed. In other words, this measure reaches its upper bound when ep reaches the lower bound tp^m, given the itemset size m and the actual probability tp. Similarly, any correlation measure which satisfies Property 3 reaches its lower bound when ep reaches the upper bound.

2.4.1.6 Pair-only Correlation Measures

All the pair-only correlation measures in this chapter satisfy Property 3. Given tp and $m = 2$, $tp^2 \leq ep \leq ((1+tp)/2)^2$. They reach their upper bound when $ep = tp^2$. When $ep = tp^2$, we have $f_{10} = 0$, $f_{01} = 0$, and $f_{00} = n - f_{11}$. According to the formulas shown in Table 2.9, it is easy to get $\phi_{ub} = 1$, $RelativeRisk_{ub} = \infty$, $OddsRatio_{ub} = \infty$, $Conviction_{ub} = \infty$, and $AddedValue_{ub} = 1 - tp$. The lower bounds for the pair-only correlation measures are achieved in one or more of the following situations: (1) $f_{11} = tp$, $f_{10} = 1 - tp - \epsilon$, $f_{01} = \epsilon$, and $f_{00} = 0$ where ϵ is a very small positive number. (2) $f_{11} = tp$, $f_{10} = \epsilon$, $f_{01} = 1 - tp - \epsilon$, and $f_{00} = 0$ where ϵ is a very small positive number. (3) $f_{11} = tp$, $f_{10} = (1 - tp)/2$, $f_{01} = (1 - tp)/2$, and $f_{00} = 0$. ϕ reaches its lower bound $-\frac{1-tp}{1+tp}$ in Case 3. Relative Risk reaches its lower bound tp in Case 1. Odds Ratio reaches its lower bound 0 when $f_{00} = 0$. Conviction reaches its lower bound tp in Case 2. Added Value reaches its lower bound $-\frac{(1-tp)^2}{2\cdot(1+tp)}$ in Case 3.

2.4.2 Upper and Lower Bound Summary

Table 2.11 shows the upper and lower bounds of all the measures given tp. Figures 2.1, 2.2, 2.3, 2.4, and 2.5 show the upper and lower bounds of the various measures with respect to different Support and itemset sizes. It is easy to see that different measures favor itemsets within different Support ranges.

Correlation Measure	Upper Bound	Lower Bound
Support	tp	tp
Any-confidence	1	$\frac{m \cdot tp}{m-1+tp}$
All-confidence	1	tp
Bond	1	tp
Simplified χ^2-statistic	$\frac{(tp-tp^m)^2}{tp^m}$	$-(tp-(\ \frac{m-1+tp}{m}\ )^m)^2 \cdot (\ \frac{m}{m-1+tp}\ )^m$
Probability Ratio	$\frac{tp}{tp^m}$	$tp \cdot \left(\frac{m}{m-1+tp}\right)^m$
Leverage	$tp - tp^m$	$tp - \left(\frac{m-1+tp}{m}\right)^m$
Likelihood Ratio	$tp \cdot ln\ \frac{tp}{tp^m}\ + (1-tp) \cdot ln\ \frac{1-tp}{1-tp^m}$	$-tp \cdot ln\ \frac{tp \cdot m^m}{(m-1+tp)^m}\ - (1-tp) \cdot ln\ \frac{(1-tp) \cdot m^m}{m^m-(m-1+tp)^m}$
BCPNN	$\frac{tp+cc}{tp^m+cc}$	$\frac{(tp+cc) \cdot m}{m-1+tp+cc \cdot m}$
SCWCC	$\frac{(tp-tp^m)^2}{tp^m+cc}$	$-\frac{(tp \cdot m^m-(m-1+tp)^m)^2}{m^m \cdot (m-1+tp)^m+cc \cdot m^{2m}}$
IS	$tp^{(1-m/2)}$	$tp \cdot (\ \frac{m}{m-1+tp}\ )^{m/2}$
Two-way Support	$(1-m) \cdot tp \cdot ln(tp)$	$tp \cdot ln(tp) - m \cdot tp \cdot ln\ \frac{m-1+tp}{m}$
SCWS	$\frac{tp \cdot (tp-tp^m)^2}{tp^m}$	$-\frac{tp \cdot (tp-((m-1+tp)/m)^m)^2}{((m-1+tp)/m)^m}$
ϕ-coefficient	1	$-\frac{1-tp}{1+tp}$
Relative Risk	∞	tp
Odds Ratio	∞	0
Conviction	∞	tp
Added Value	$1-tp$	$-\frac{(1-tp)^2}{2(1+tp)}$

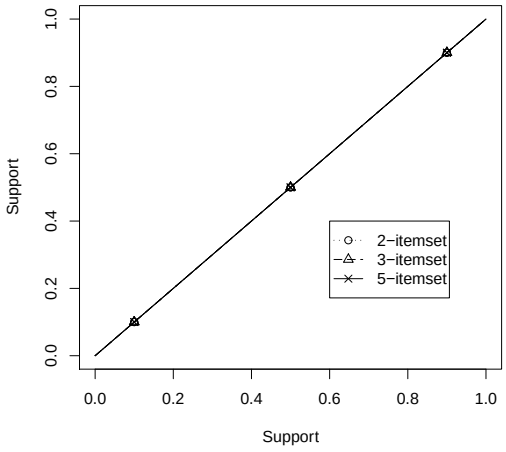
Table 2.11: Bounds of correlation measures

2.4.2.1 Sub-optimal measures

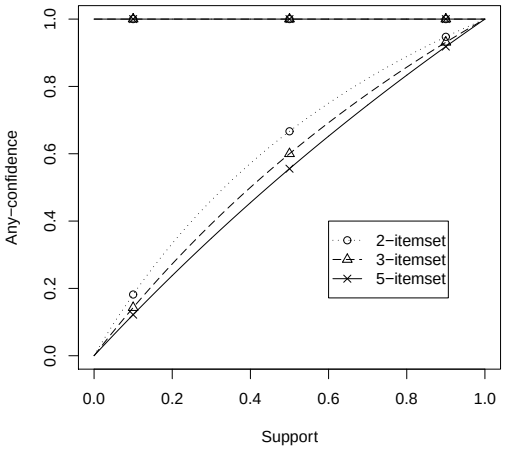
Since both the upper bound and lower bound of Support are itself, Support strictly favors high Support itemset. In Figure 2.1, Any-confidence has the fixed upper bound 1, but the lower bound of Any-confidence increases with the increase of tp. Given an itemset with the fixed Support tp and the fixed itemset size m, we assume its Any-confidence follows a certain distribution between its upper bound 1 and the lower bound $m \cdot tp/(m - 1 + tp)$. The expected Any-confidence increases with the increase of tp when m is fixed, and the expected Any-confidence decreases with the increase of m when tp is fixed. Any-confidence favors high Support and small size itemsets. Similar to Any-confidence, All-confidence and Bond favor high Support itemsets. Though the lower bounds of All-confidence and Bond have nothing to do with the itemset size m, Support favors small size itemsets by its nature. Therefore, All-confidence and Bond favor small size itemsets indirectly. In addition, the lower bounds of All-confidence and Bond are lower than that of Any-confidence. Therefore, All-confidence and Bond favor higher Support itemsets than Any-confidence.

2.4.2.2 Other general measures

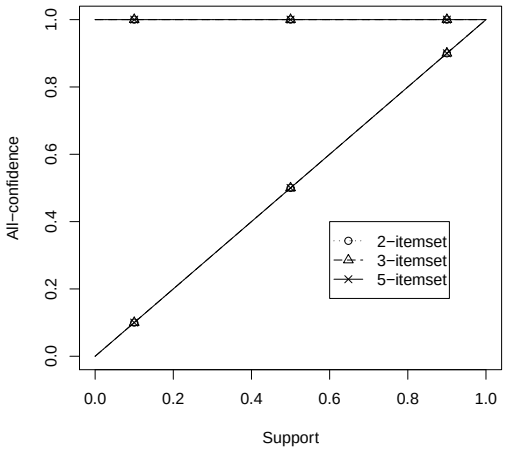
The upper bounds of the Simplified χ^2-statistic and Probability Ratio increase to infinity when Support is close to 0, which means they favor coincidences rather than significant patterns. The only special situation is that the upper bound of the Simplified χ^2-statistic is equal to 1 instead of ∞ when the itemset size is 2. That explains why χ^2 works well for pairs but poorly for larger itemsets. As we increase the itemset size, the upper bounds of the Simplified χ^2-statistic and Probability Ratio get higher as Support approaches 0. Compared to the Simplified χ^2-statistic, Probability Ratio is more biased to low Support itemset with large size.



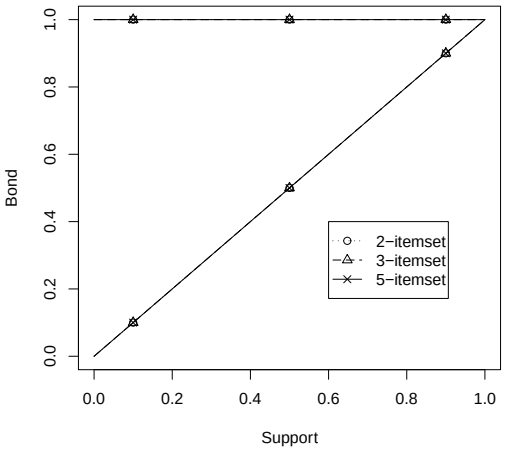
(a) Support



(b) Any-confidence



(c) All-confidence

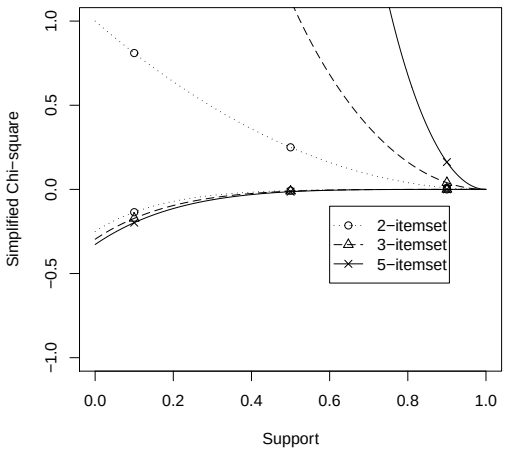


(d) Bond

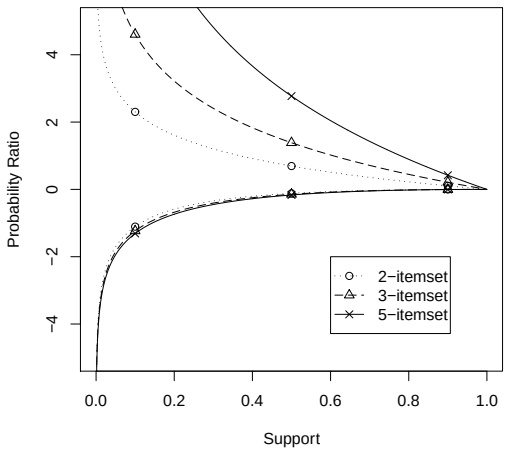
Figure 2.1: Upper and lower bounds of sub-category 1

Leverage, Likelihood Ratio, BCPNN, Two-way Support, and SCWCC reach their highest upper bound when tp is between 0 and 1. For Leverage, the maximal value is reached when tp is between 0.5 and 0.8. In other words, the itemset with tp between 0.5 and 0.8 has better chance to get higher value. As we can see, different measures favor different tp regions. According to Figures 2.2 and 2.3, BCNNP favors lowest-Support itemsets, followed by SCWCC, Likelihood Ratio, Two-way Support, and Leverage. The favored Support range of BCPNN and SCWCC can be adjusted by different continuity correction numbers according to Figure 2.3. Normally, we recommend $cc = 0.5/n$ for clear datasets. If the dataset is dirty, we might use a larger value to favor relatively high Support region to suppress the false positive correlations from the noise data in the low Support region, but the large value will also suppress the true positive correlations in the low Support region at the same time. Therefore, improper cc adjustment will degrade the effectiveness of BCPNN and SCWCC.

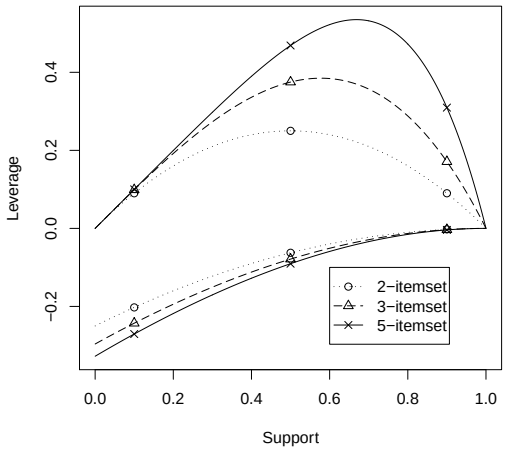
Tan et al. [64] purposed IS by hoping the additional Support can suppress the increase of Probability Ratio when Support is close to 0. It works for 2-itemsets, but fails for large size itemsets. Better than IS, Two-way Support successfully decreases the upper bound when Support is close to 0. However, the lower bound trend is also reversed when the Support is close to 0, which is not what we want. We expect the highest negatively-correlated itemsets to come from the low Support region. The Simplified χ^2 with Support has both the disadvantage of IS and the disadvantage of Two-way Support. It successfully suppresses the upper bound of 2-itemsets and 3-itemsets, but not for itemsets with the size greater than 3. In addition, the trend of lower bound is reversed when the Support is close to 0.



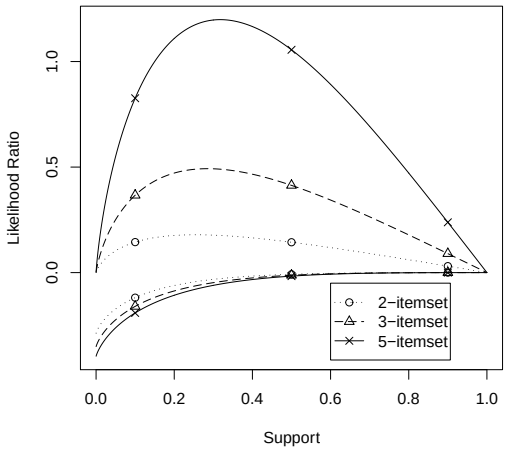
(a) Simplified χ^2



(b) Probability Ratio

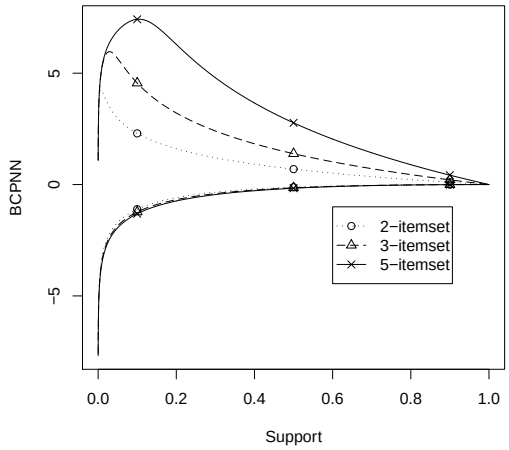


(c) Leverage

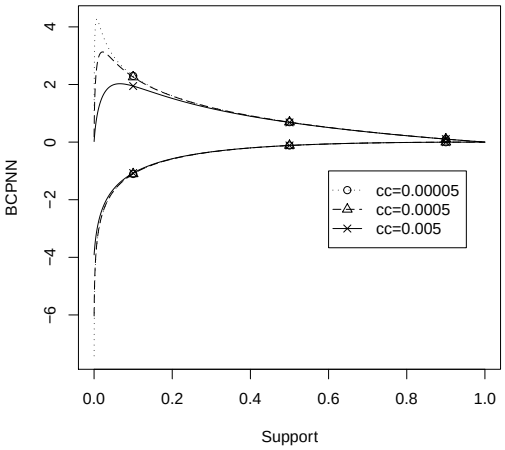


(d) Likelihood Ratio

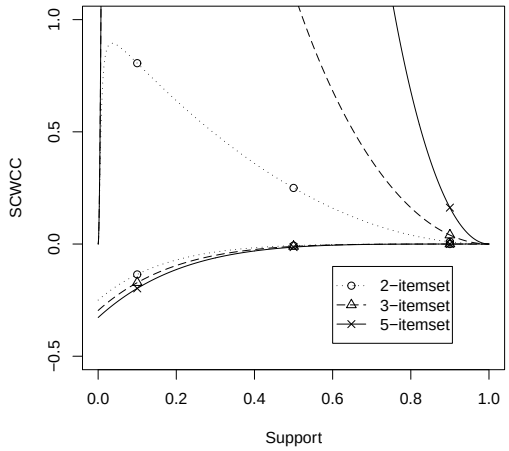
Figure 2.2: Upper and lower bounds of sub-category 2



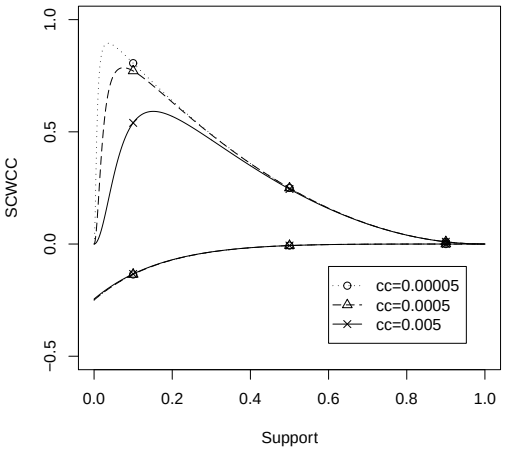
(a) BCPNN for different itemset size when $cc = 0.00005$



(b) BCPNN for different cc when itemset size is 2

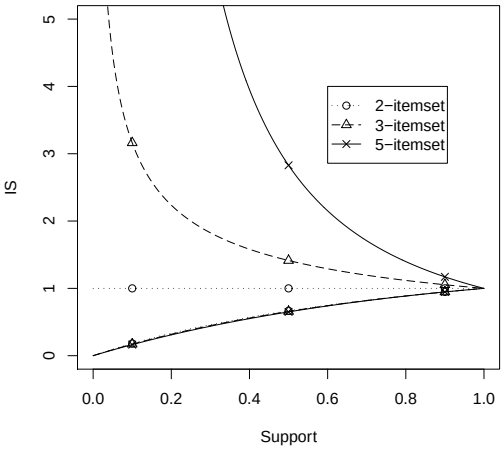


(c) SCWCC for different itemset size when $cc = 0.00005$

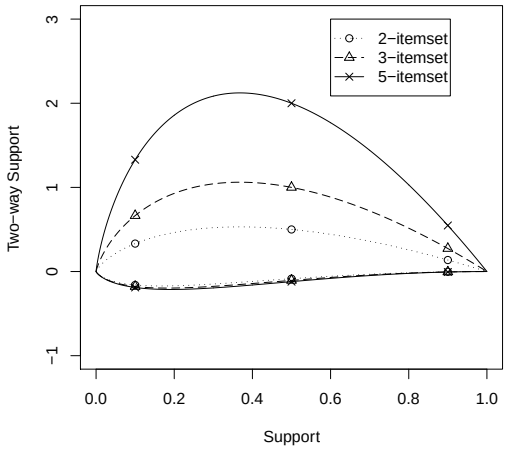


(d) SCWCC for different cc when itemset size is 2

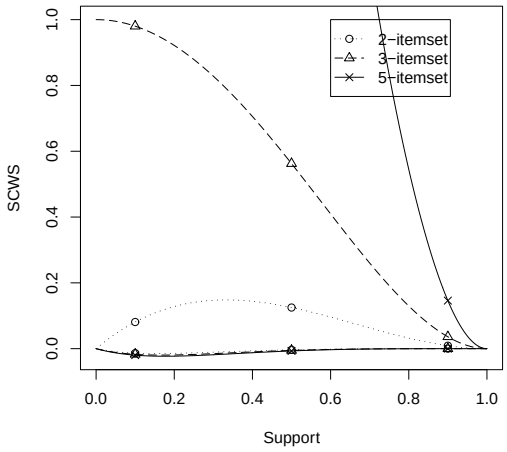
Figure 2.3: Upper and lower bounds of sub-category 3



(a) IS

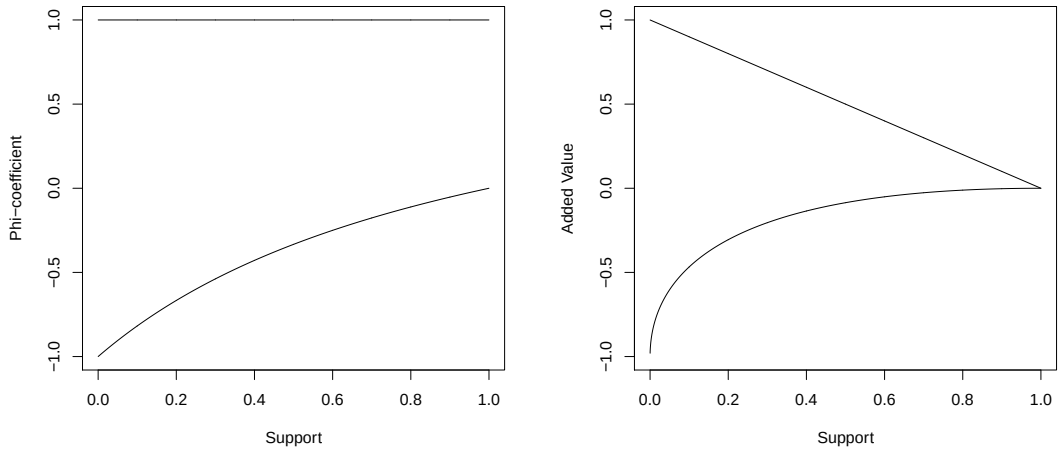


(b) Two-way Support



(c) Simplified χ^2 with Support

Figure 2.4: Upper and lower bounds of sub-category 4



(a) ϕ-coefficient (b) Added Value

Figure 2.5: Upper and lower bounds of pair-only measures

2.4.2.3 Pair-only measures

The upper bound of ϕ-coefficient, Relative Risk, Odds Ratio, and Conviction is a fixed number, they don't favor any region. The highest value can come from anywhere. The upper bound of Added Value decreases with the increase of Support, which violates Property 4.

2.5 Experiments

There is no ground truth for us to compare with for real life datasets as correlation search is an unsupervised learning procedure. Therefore, we will make use of the characteristics of different datasets to evaluate the performance of different measures.

2.5.1 OMOP dataset

The Observational Medical Outcomes Partnership (OMOP[2]) is a public-private partnership to help monitor drug safety. For observational analysis, we want to find correlation between drugs and conditions from a population. To facilitate the methodological research, it typically requires some 'gold standard' to measure performance. However, the ground truth may not be absolutely determined because most observational data sources are poorly characterized, and clinical observations may be insufficiently recorded or poorly validated. Because of these issues, OMOP developed a simulation procedure to supplement the methods evaluation.

The simulated dataset has the predefined associations between drugs and conditions. For each condition, each synthetic patient has a prevalent probability of having it. When the patient takes the related drugs for a certain condition, the probability of having it will increase. The dataset contains ten million persons, more than 90 million drug exposures from 5000 unique drugs and more than 300 million condition occurrences from 4500 unique conditions over a span of 10 years. In order to simulate the reality, most drugs and conditions only happen a few times. Therefore, only 1/3 of the predefined associations are observed in the simulated dataset. In addition, among those predefined associations being observed, most of them only happen a few times.

A key step in the application of our correlation measure is the mapping of the data into drug-condition two-by-two tables, and then we can calculate the correlation between drugs and conditions. Among different ways of constructing two-by-two tables for longitudinal data (such as claims databases or electronic health records),

[2]http://omop.fnih.org/

we only use the "Modified SRS" method [67] to construct the two-by-two contingency table which is the benchmark proposed by OMOP.

We check the bias and performance for each measure. First, we calculated the average Support of the top-K pairs retrieved by each measure. If the value is large, the measure favors frequent correlation patterns. Second, the mean average precision (MAP), a commonly-used metric in the field of information retrieval, is used to evaluate each method. It measures how well a system ranks items, and emphasizes ranking true positive items higher. Let y_{dc} be equal to 1 if the dth drug causes the cth condition, and 0 otherwise. Let $M = \sum_{d,c} y_{dc}$ denote the number of causal combinations and $N = D \times C$ the total number of combinations. Let z_{dc} denote the correlation value for the dth drug and cth condition. For a given set of correlation values $\overrightarrow{z} = (z_{11}, ..., z_{DC})$, we define "precision-at-K" denoted $P^K(\overrightarrow{z})$ as the fraction of causal combinations among the K largest predicted values in $\overrightarrow{z}$. Specifically, let $z_1 > ... > z_N$ denote the ordered value of $\overrightarrow{z}$. Then, $P^K(\overrightarrow{z}) = \frac{1}{K}\sum_{i=1}^{K} y_i$, where y_i is the true status of the combination corresponding to z_i. The MAP is calculated as $\frac{1}{M}\sum_{K=1}^{N}(P^K(\overrightarrow{z}) \cdot y_K)$. The MAP is very similar to the area under the precision-recall curve, which penalizes both types of misclassification: identifying a correlation when no relationship exists (false positive) and failing to identify true correlations (false negative). Table 2.12 shows the average Support of the top 1000 pairs and the MAP for each measure.

Since only 1/3 of the predefined associations are observed, no methods can achieve MAP beyond 0.33 unless it can infer unobserved drug-condition correlations. In Section 2.2, correlation properties are categorized into four groups: mandatory, desired, optional, and pair-only. Here, we study the effectiveness of these properties in terms of selecting good correlation measures.

Type	Measures	Average Support of the Top 1000 pairs	Mean Average Precision
Sub-optimal Measures	Support	55849.62	0.0344
	Any-confidence	9377.08	0.0925
	All-confidence	32425.03	0.0668
	Bond	33330.55	0.0694
Basic Measures	Simplified χ^2-statistic	31838.57	0.2258
	Probability Ratio	71.19	0.1102
	Leverage	44955.40	0.1472
	Likelihood Ratio	42298.72	0.2505
Adjusted Measures	BCPNN	2984.61	0.2440
	SCWCC	35370.45	0.2415
	IS	32531.93	0.0961
	Two-way Support	43695.10	0.1876
	SCWS	41345.57	0.1983
Pair-only Measures	ϕ-coefficient	31855.09	0.2256
	Relative Risk	89.62	0.1070
	Odds Ratio	6928.90	0.0482
	Conviction	4344.26	0.1020
	Added Value	4344.00	0.1016

Table 2.12: Evaluation Result for OMOP data

First, Support, Any-confidence, All-confidence, Bond, and IS violate some mandatory properties, and all their MAPs are below 0.1. If the correlation measures satisfy all the mandatory properties, all their MAPs are above 0.1 except Odds Ratio which is frequently used for case-control studies and retrospective studies.

Second, among all the measures satisfying all three mandatory properties, if they satisfy two desired properties proposed by us, their MAPs are generally better. In order to simulate the reality, most drugs and conditions only happen a few times; therefore, the Support of most predefined associations is small. However, Leverage favors the high Support region, and that is why Leverage doesn't work well in this dataset. According to the average Support of the top 1000 pairs, Leverage favors the highest Support pairs, followed by Two-way Support, Likelihood Ratio, SCWCC, and BCPNN, which is consistent with Figures 2.2, 2.3, and 2.4. Here, we are measuring the

performance of pair search. If we only consider the upper bound for pairs, SCWS also satisfies two desired properties and its MAP is good. If we further relax Property 4 to "The upper bound of M won't increase to infinity when $P(S)$ is close to 0", Simplified χ^2-statistic satisfies the relaxed Property 4 since its upper bound is equal to 1 instead of ∞ when the itemset size is 2. Therefore, when searching correlated pairs, Simplified χ^2-statistic works well. This also explains why statisticians usually use χ^2-statistic for pairs but doubt the performance of χ^2-statistic for large-size itemsets. The upper bound of Probability Ratio increases to infinity when $P(S)$ is close to 0. It favors coincidences rather than significant correlations in the data, which is verified by its small average Support of the top 1000 pairs.

Third, Simplified χ^2-statistic, Likelihood Ratio, and SCWCC satisfy all the optional properties. Here, we are not interested in the negative correlation and search positive correlation in one dataset. Therefore, the optional properties won't help us to identify good correlation measures in this experiment.

At last, satisfying the first and third additional properties cannot help us to identity good correlation measures. In addition, only bad correlation measures satisfy the second additional property. Therefore, we doubt the three additional properties qualify as desired properties.

2.5.2 Facebook dataset

We crawled Facebook for the University of Iowa community. The resulting dataset contains 41,073 people and their friend information within this community. If we consider each friend list as a transaction and people as items in the transaction, we can calculate the correlation between any two people in this community and get a ranking list of how people are correlated with each other. However, we don't have the

Measures	Mean Average Precision	Mean Personal MAP
Support	0.4415	0.7084
Any-confidence	0.3657	0.6106
All-confidence	0.4865	0.5920
Bond	0.5062	0.6302
Simplified χ^2-statistic	0.5029	0.6563
Probability Ratio	0.1800	0.4312
Leverage	0.4579	0.7278
Likelihood Ratio	0.5287	0.7363
BCPNN	0.5168	0.7342
SCWCC	0.4970	0.7327
IS	0.5067	0.6582
Two-way Support	0.5177	0.7360
SCWS	0.5540	0.7104
ϕ-coefficient	0.5033	0.6564
Relative Risk	0.1275	0.1977
Odds Ratio	0.1609	0.3278
Conviction	0.2420	0.5874
Added Value	0.3224	0.7278

Table 2.13: Evaluation result for Facebook data

ground truth whether two given people are correlated or not in this real life dataset. Therefore, we assume two given people are correlated with each other if they are friends of each other. Since the ground truth is not perfect, the experimental result is only complementary to the OMOP result. By using the friend relationships within this community, we can calculate the MAP to evaluate the friend ranking list as we do in the OMOP dataset. Another interesting question related to this dataset is how other people are correlated to a particular person. The ranking list of this type is useful for friend recommendation in Facebook. Similarly, we can calculate the MAP for each ranking list related to a particular person, and then average all the personal MAP values. All the evaluation values are shown in Table 2.13.

Surprisingly, sub-optimal measures work pretty well in this application. The result supports why the Facebook friend recommender system recommends the per-

son with the most common friends. The reason why sub-optimal measures work well in this application is as follows. If two people know 90% of the people in this community and have a lot of common friends, the chance for them to know each other is high. Knowing each other doesn't mean the high correlation with each other. For example, we might make friends with a stranger who doesn't know any of our friends. We use friendship as a indicator for correlation because there is no better indicator for this dataset. Although friendship biases to Support, for Simplified χ^2-statistic, Leverage, Likelihood Ratio, BCPNN, SCWCC, Two-way Support, SCWS which satisfy all the mandatory and relaxed desired properties, they can still do slightly better than Support. Another interesting measure that works well for mean personal MAP in this application is Added Value. The formula, $P(B|A) - P(B)$, indicates that B can achieve a high score if B knows most of the people A knows. In social activity, B usually has a tight connection with A first in order to know most of the people A knows.

2.5.3 Netflix Dataset

Since different correlated itemsets might have an overlapping part, the simulation and validation procedure is controversial. Therefore, we will make use of the characteristics of the Netflix dataset [3] to evaluate the effectiveness of different correlated itemset search methods. Since the Netflix dataset contains 17,770 movies and has around 480 thousand transactions, it is impossible to find the top-k correlated itemsets due to the computational cost. Therefore, we create a subset of Netflix which only contains the first 100 movies (according to the code sequence in the Netflix dataset), and use brute-force search method to find top-k correlated itemsets in this

[3] http://www.netflixprize.com/

subset.

We show the top-3 correlated itemsets of typical measures and Support for each itemset in Table 2.14. First, all three patterns retrieved by Support contain the movie "Something's Gotta Give." It is the most popular movie in the subset. Considering the probability of liking "Dragonheart" conditioned on liking "Something's Gotta Give" is lower than the probability of liking "Dragonheart," Support is a bad measure for correlation search. Second, Simplified χ^2-statistic, Probability Ratio, IS, and the SCWS violate the desired Property 4, and they retrieve the same three long patterns that only happen once. The upper bounds of these four measures become steeper when the itemset size increases from 2 to 3 to 5, and their upper bounds increase to infinity when Support is close to 0. In other words, they favor rare itemsets with large size. The upper bound graphs are consistent with the experimental result. If the measure violates the desired properties proposed by us, the performance might be good for pair search, but they are all bad for itemset search. Third, for the measures satisfying all the mandatory and desired properties, Leverage favors frequent correlation patterns followed by Two-way Support, Likelihood Ratio, SCWCC, and BCPNN according to the Support of retrieved itemsets.

In the Netflix data, we can assume movies in the same series are strongly correlated. Since this subset contains few movies in the same series, most retrieved patterns are not movies in the same series which is hard to justify. It would be better if we can find the top-k patterns in the whole Netflix dataset. Therefore, we make use of the maximal fully-correlated itemset framework [25] in Section 3.2.4 to find the top-5 patterns in the whole Netflix dataset for four typical measures in Table 2.15. Only one of the five patterns retrieved by Support is movies in the same series. All the five patterns retrieved by Leverage, Likelihood Ratio, and BCPNN are movies

Measures	Top-3 correlated itemsets	Support
Support	Lilo and Stitch (2002), Something's Gotta Give (2003)	9574
	Something's Gotta Give (2003), Silkwood (1983)	5248
	Something's Gotta Give (2003), Dragonheart (1996)	3736
Simplified χ^2-statistic, Probability Ratio, IS, and SCWS	a set with 18 movies	1
	a set with 17 movies	1
	a set with 17 movies	1
Leverage	Lilo and Stitch (2002), Dragonheart (1996)	3108
	Dragonheart (1996), Congo (1995)	1091
	Spitfire Grill (1996), Silkwood (1983)	1207
Two-way Support	Lilo and Stitch (2002), Dragonheart (1996)	3108
	Dragonheart (1996), Congo (1995)	1091
	Spitfire Grill (1996), Silkwood (1983)	1207
Likelihood Ratio	Dragonheart (1996), Congo (1995)	1091
	Lilo and Stitch (2002), Dragonheart (1996), Congo (1995)	501
	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005)	153
SCWCC	My Favorite Brunette (1947), The Lemon Drop Kid (1951)	103
	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005)	153
	Screamers (1996), Dragonheart (1996), Congo (1995)	120
BCPNN	My Favorite Brunette (1947), The Lemon Drop Kid (1951)	103
	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005), ECW: Cyberslam '99 (2002)	41
	WWE: Armageddon (2003), WWE: Royal Rumble (2005)	47

Table 2.14: Top-3 correlated itemsets for Netflix data

in the same series. Leverage and Likelihood Ratio find popular movie series, while BCPNN retrieves unpopular movie series, which is consistent with our correlation analysis.

2.6 Conclusion

In the chapter, we did a comprehensive study on effective correlation search for binary data. First, we studied 18 different correlation measures and proposed 2 desirable properties to help us select good correlation measures. Second, we studied different techniques to adjust original correlation measures, and use them to propose two new correlation measures: the Simplified χ^2 with Continuity Correction and the Simplified χ^2 with Support. Third, we studied the upper and lower bounds of different measures to find their different favorable Support region. Last, we made use of the characteristics of different data sets to validate our conclusions. In Section 1, we men-

Measure	Maximal fully-correlated itemsets
Support	The Lord of the Rings: The Fellowship of the Ring (2001), The Lord of the Rings: The Two Towers (2002), The Lord of the Rings: The Return of the King (2003)
	Forrest Gump (1994), The Green Mile (1999)
	The Lord of the Rings: The Two Towers (2002), Pirates of the Caribbean: The Curse of the Black Pearl (2003)
	The Lord of the Rings: The Fellowship of the Ring (2001), Pirates of the Caribbean: The Curse of the Black Pearl (2003)
	Forrest Gump(1994), The Shawshank Redemption: Special Edition (1994)
Leverage	The Lord of the Rings: The Fellowship of the Ring (2001), The Lord of the Rings: The Two Towers (2002), The Lord of the Rings: The Return of the King (2003)
	Raiders of the Lost Ark (1981), Indiana Jones and the Last Crusade (1989)
	Star Wars: Episode V: The Empire Strikes Back (1980), Star Wars: Episode VI: Return of the Jedi (1983)
	The Lord of the Rings: The Fellowship of the Ring: Extended Edition (2001), The Lord of the Rings: The Two Towers: Extended Edition (2002)
	Star Wars: Episode IV: A New Hope (1977), Star Wars: Episode V: The Empire Strikes Back (1980)
Likelihood Ratio	The Lord of the Rings: The Fellowship of the Ring: Extended Edition (2001), The Lord of the Rings: The Two Towers: Extended Edition (2002), The Lord of the Rings: The Return of the King: Extended Edition (2003)
	Star Wars: Episode IV: A New Hope (1977), Star Wars: Episode V: The Empire Strikes Back (1980), Star Wars: Episode VI: Return of the Jedi (1983)
	The Lord of the Rings: The Fellowship of the Ring (2001), The Lord of the Rings: The Two Towers (2002), The Lord of the Rings: The Return of the King (2003)
	Harry Potter and the Sorcerer's Stone (2001), Harry Potter and the Chamber of Secrets (2002)
	Kill Bill: Vol. 1 (2003), Kill Bill: Vol. 2 (2004)
BCPNN	Roughnecks: The Starship Troopers Chronicles: The Homefront Campaign (2000), Roughnecks: The Starship Troopers Chronicles: The Klendathu Campaign (2000)
	Now and Then, Here and There: Vol. 1: Discord and Doom (1999), Now and Then, Here and There: Vol. 2: Flight and Fall (2002), Now and Then, Here and There: Vol. 3: Conflict and Chaos (1999)
	Dragon Ball: King Piccolo Saga: Part 1 (1986), Dragon Ball: King Piccolo Saga: Part 2 (1986) Dragon Ball: Piccolo Jr. Saga: Part 1 (1995), Dragon Ball: Piccolo Jr. Saga: Part 2 (1995)
	Dragon Ball: Red Ribbon Army Saga (2002), Dragon Ball: Commander Red Saga (2002)
	Dragon Ball: Piccolo Jr. Saga: Part 1 (1995), Dragon Ball: Piccolo Jr. Saga: Part 2 (1995) Dragon Ball: Red Ribbon Army Saga (2002)

Table 2.15: Top-3 correlated itemsets for Netflix data

tioned different correlation measures are favored in different domains by studying the literature related to correlation, which can be explained by our correlation analysis. In text mining, too frequent or rare words don't have too much discriminative power for classification. Therefore, they favor Likelihood Ratio to find the pattern which is not too rare or too frequent [30]. In medical domain, the associations between frequent diseases and frequent symptoms have already been observed in the clinical trials. The big issue is how to find the correlation between rare diseases and rare symptoms. That is why they use BCPNN [7]. In social networks, people are more interested the patterns affecting a larger population. Therefore, they favor Leverage [20]. We recommend Leverage for searching obvious patterns in the dataset that we know nothing about. We suggest Likelihood Ratio, Simplified χ^2 with Continuity Correction, and Two-way Support for searching typical patterns in the dataset that we know something of. We refer BCPNN for searching rare patterns in the dataset that we know well.

CHAPTER 3
CORRELATED ITEMSET MINING

3.1 Introduction and Related Work

Much previous research focuses on finding correlated pairs instead of correlated itemsets in which all items are correlated with each other. However, there are some applications in which we are specifically interested in correlated itemsets rather than correlated pairs. For example, we are interested in finding sets of correlated stocks in a market, or sets of correlated gene sequences in a microarray experiment. But finding correlated itemsets is much harder than finding correlated pairs because of three major problems. First, computing correlation for each possible itemset is an NP-complete problem [48]. Second, if there are some highly correlated items within an itemset and the rest are totally independent items, most correlation measures still indicate that the itemset is highly correlated. No existing measure provides information to identify the itemsets with independent items. Third, there is no guarantee that the itemset has high correlation if any of its strict subsets are highly correlated. Since the correlated pair search can be considered a special case of the correlated itemset search for 2-itemsets, we focus on the correlated itemset search in the rest of the chapter.

The first related technique for correlated itemset search is frequent itemset mining. By using support, the search is fast. However, co-occurrence is related to two factors: the correlation and the single item support within the itemset. In other words, co-occurrence is related but not equal to correlation. The second technique is to find the top-k correlated itemsets. Tan et al. [81] compared 21 different measures for correlation. Only six of the 21 measures can be used to measure the correlation within a given k-itemset.

The above correlation measures can search for more meaningful correlated patterns than support, but do not have the downward-closed property to reduce the computational expense. Once we can find a set which does not satisfy a given downward-closed property, we can prune the exponential superset search space immediately. Since existing correlation measures satisfying the three primary correlation properties [25] do not possess the downward-closed property to facilitate the search, finding the top-k correlated itemsets is very computationally expensive. To sum up, frequent itemset mining is efficient, but not effective; top-k correlated itemset mining is effective, but not efficient. In order to solve the problem, others proposed efficient correlation measure like all-confidence [66]. All-confidence is as fast as support; however, the correlation is still measured in a sub-optimal way. Requiring the correlation measure to be both effective and efficient is too demanding. Instead of proposing another efficient correlation measure, we propose the framework of fully-correlated itemsets (FCI) [25], in which any two subsets are correlated. This framework can not only decouple the correlation measure from the need for efficient search, but also rules out the itemsets with irrelevant items. With it, we only need to focus on effectiveness when selecting correlation measures.

Even though the FCI framework can impose the downward-closed property for any given correlation measure, there are still two computational issues to find the desired maximal fully-correlated itemsets (MFCIs). First, unlike finding maximal frequent itemsets which can start pruning from 1-itemsets, finding MFCIs must start pruning from 2-itemsets. However, as the number of items and transactions in the dataset increases, calculating the correlation values for all the possible pairs is computationally expensive. Since there is no monotone property for correlation measures which can help prune, the brute-force method is straightforward. When a database

contains 10^5 items, a brute-force approach requires computing the correlation value of $0.5 * 10^{10}$ pairs. Even worse, when the support of all the pairs cannot be loaded into the memory, the IO cost for retrieving supports is much more expensive than the computational cost for calculating correlations. Therefore, an efficient correlated pair search algorithm can speed up the MFCI search.

The most significant progress on correlated pair search was made by Xiong et al. [89, 90], who made use of the upper bound of the Pearson correlation coefficient (ϕ-coefficient) for binary variables. The computation of this upper bound is much cheaper than the computation of the exact correlation because this upper bound is a function of single item supports. In addition, the upper bound has special 1-dimensional and 2-dimensional properties that prune many pairs from the search space without the need to compute their upper bounds. The algorithm TAPER [90] makes use of the 1-dimensional and 2-dimensional properties to retrieve correlated pairs above a given threshold. The algorithm TOP-COP [89] uses the 1-dimensional property and a diagonal traversal method, combined with a refine-and-filter strategy, to efficiently find the top-k pairs. However, this work is only related to the ϕ-coefficient, which is not the only or the best correlation measure. Second, users usually need to try different correlation thresholds for different desirable MFCIs. For example, when we set the likelihood ratio threshold to $15,000$ using the Netflix dataset, we successfully retrieve the series of "Lord of the Rings 1, 2, and 3" in one MFCI; however, we only retrieve several pairs of the TV show "Sex and the City" like {1, 2}, {2, 3}, {3, 4}, {4, 5}, {5, 6}. In order to get the whole series of "Sex and the City 1, 2, 3, 4, 5, 6" in one MFCI, we have to lower the threshold to 8000. However, the cost of processing the Apriori procedure each time for a different correlation threshold is very high. Since the framework of FCI is relatively new, there is

no related work on improving its efficiency.

Since itemset mining has a long history from 1993 when frequent itemset mining [2] was proposed, a lot of related concepts are proposed such as closed itemset [92], contrast itemset [1], and discriminative itemset [19]. These concepts are related to each other in a certain degree; however, in this chapter, we only focus on the performance issue and the general framework of finding correlated itemsets where the correlation measure must explicitly use the measures true probability tp and expected probability ep. In addition, there are several other issues related to correlation that we also don't address here, such as statistical type-1 and type-2 error reduction, error-tolerant methods using sampling, and the existing optimization methods on itemset mining. In Chapter 2, we carefully studied 18 correlation measures and provided four extra properties for correlation measures from the statistical point of view which can help users to choose the correlation measure retrieving results closer to human intuition. Webb [87] proposed a framework of reducing the type-1 and type-2 error of itemset mining which can be applied to our pattern search framework. Zhang and Feigenbaum [94] studied the distribution of the ϕ-coefficient and relaxed the upper bound in TAPER in order to speed up search. Guns et al. [43] formulated the traditional itemset mining problem, such as frequent, closed, and discriminative itemset mining, as constraint programming problems, and then applied an existing solver for constraint programming to speed up the search. However, traditional itemset mining can easily retrieve the value of a given itemset S, while the fully-correlated itemset mining cannot retrieve the value of a given itemset S without calculating the value of all the subsets of S. Therefore, there is no obvious way of formulating our problem as a constraint programming problem.

The rest of this chapter is organized as follows. Section 2 presents basic notions

of correlation upper bound, 1-dimensional and 2-dimensional properties, and the fully-correlated itemset framework. We propose several methods to speed up correlated pair and correlated itemset search in Section 3. Section 4 shows the experimental results. Finally, we draw a conclusion in Section 5.

3.2 Basic Properties

In this section, some basic properties of correlation are introduced to better explain the improved performance of correlation search. We only address the performance issue related to the correlation measures satisfying the first three mandatory properties in Section 2.2.1. Without loss of generality, in this chapter we do experiments by using the typical correlation measure [25], likelihood ratio, which measures the ratio of the likelihood of k out of n transactions containing the itemset S when the single trial probability is the true probability to that when the single trial probability is the expected probability if all the items in S are independent from each other.

3.2.1 Correlation Upper Bound for Pairs

Theorem 3.1. *Given any pair* $\{I_i, I_j\}$ *and support values* $P(I_i)$ *for item* I_i *and* $P(I_j)$ *for item* I_j*, the correlation upper bound* $CUB(I_i, I_j)$*, i.e. the highest possible correlation value of the pair* $\{I_i, I_j\}$*, is the correlation value for* $\{I_i, I_j\}$ *when* $P(I_i \cap I_j) = min\{P(I_i), P(I_j)\}$.

Proof. The upper bound of the support value $P(I_i \cap I_j)$ for the 2-itemset $\{I_i, I_j\}$ is $min\{P(I_i), P(I_j)\}$ and the lower bound is $max\{0, P(I_i) + P(I_j) - 1\}$. For the given $P(I_i)$ and $P(I_j)$, any correlation measure reaches its upper bound when $P(I_i \cap I_j) = min\{P(I_i), P(I_j)\}$ and its lower bound when $P(I_i \cap I_j) = max\{0, P(I_i) + P(I_j) - 1\}$ according to correlation property 2.

The calculation of correlation upper bound (CUB) for pairs only needs the support of each item which can be saved in memory even for large datasets; however, the calculation of correlation for pairs needs the support of pairs, incurring a high IO cost for large datasets. Given a 2-itemset $\{I_i, I_j\}$, if its correlation upper bound is lower than the correlation threshold we specify, there is no need to retrieve the support of the 2-itemset $\{I_i, I_j\}$, because the correlation value for this pair is definitely lower than the threshold no matter what the support is. If we only retrieve the support of a given pair in order to calculate its correlation when its upper bound is greater than the threshold, we will save a lot of unnecessary IO cost when the threshold is high.

3.2.2 1-Dimensional Property

Although the correlation upper bound calculation can save a lot of unnecessary IO cost, it still requires the correlation upper bound calculation for all the possible pairs. Therefore, we make use of the 1-dimensional property to eliminate unnecessary upper bound checks. It is motivated by the search algorithm TAPER [90] for the ϕ-coefficient. In order to fit the situation for any good correlation measure, we sort the items according to their supports in increasing order instead of decreasing order as in TAPER.

Theorem 3.2. *Given a user-specified threshold θ and an item list $\{I_1, I_2, ..., I_m\}$ sorted by support in increasing order, the correlation upper bound of $\{I_i, I_k\}$ is less than θ if the correlation upper bound of $\{I_i, I_j\}$ is less than θ and $i < j < k$.*

Proof. Since $i < j < k$ and the item list $\{I_1, I_2, ..., I_m\}$ is sorted by support in increasing order, $P(I_i) \leq P(I_j) \leq P(I_k)$. Then, the support upper bound of both $\{I_i, I_j\}$ and $\{I_i, I_k\}$ is equal to $P(I_i)$. For the pair $\{I_i, I_j\}$, $P_{upper}(I_i \cap I_j) = P(I_i)$ and $P_{expected}(I_i \cap I_j) = P(I_i)P(I_j)$. For the pair $\{I_i, I_k\}$, $P_{upper}(I_i \cap I_k) = P(I_i)$

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		X	X	X	X	X
I_2			X	X	X	X
I_3				X	X	X
I_4					X	X
I_5						X
I_6						

Table 3.1: Pair Matrix

and $P_{expected}(I_i \cap I_k) = P(I_i)P(I_k)$. Therefore, $P_{upper}(I_i \cap I_k) = P_{upper}(I_i \cap I_j)$, and $P_{expected}(I_i \cap I_k) \geq P_{expected}(I_i \cap I_j)$ because $P(I_k) \geq P(I_j)$. According to correlation property 3, we get $CUB(I_i, I_k) \leq CUB(I_i, I_j)$. Since $CUB(I_i, I_j) < \theta$, $CUB(I_i, I_k) < \theta$.

To get all the pair correlation upper bounds, we need to calculate an $n \times n$ matrix for an item list sorted by support as shown in Table 3.1. Since this matrix is symmetrical, we only need to calculate the upper part above the diagonal. If the data set contains n items, $(n-1)$ branches (rows) need to be calculated. The pairs in branch i are $\{I_i, I_j\}$ where $i+1 \leq j \leq n$. The reference item I_i is fixed in each branch i and it has the minimum support value due to the way we construct the branch. Since items in each branch are also sorted based on their support in increasing order, the correlation upper bound of $\{I_i, I_j\}$ monotonically decreases with the increase of j by Theorem 3.2. In other words, $CUB(I_i, I_k) < \theta$ when $CUB(I_i, I_j) < \theta$ and $j+1 \leq k \leq n$.

3.2.3 2-Dimensional Property

When the threshold is low, we might still calculate a lot of correlation upper bounds by using the 1-dimensional property. In order to avoid too many correlation

upper bound checks, a 2-dimensional property similar to TAPER [90] for the ϕ-coefficient is used. However, we present different 2-dimensional properties and use different search sequences for three different types of correlation measures.

Given three items I_i, I_j, and I_k with $P(I_i) \geq P(I_j) \geq P(I_k)$, the correlation measure M is

- Type 1 if $CUB(I_i, I_j) \geq CUB(I_i, I_k)$.
- Type 2 if $CUB(I_i, I_j) \leq CUB(I_i, I_k)$.
- Type 3 if $CUB(I_i, I_j) = CUB(I_i, I_k)$.

Given an itemset $S = \{I_1, I_2, ..., I_m\}$ with m items, the actual probability is $tp = P(S)$, the expected probability is $ep = \prod_{i=1}^{m} P(I_i)$. The simplified χ^2-statistic, leverage, likelihood ratio, and ϕ-coefficient are all type 1 correlation measures. Although we know of no existing type 2 correlation measure, we can construct a type 2 correlation measure which satisfies all the three mandatory correlation properties like

$$Correlation_{type2} = \begin{cases} \frac{\sqrt{tp-ep}}{ep}, & \text{when } tp \geq ep \\ -\frac{\sqrt{ep-tp}}{ep}, & \text{when } tp < ep \end{cases} \tag{3.1}$$

Probability ratio is a type 3 correlation measure.

If we sort items according to their support values in increasing order and calculate the upper bound for each pair, Tables 3.2, 3.3, and 3.4 show the typical patterns of different types of correlation measures. For different types of correlation measures, we get different 2-dimensional properties. For type 1 correlation measures, the upper bound value decreases from left to right and from bottom to top. If the upper bound of the current cell $\{I_i, I_j\}$ is below θ, then $CUB(I_p, I_q) < \theta$ when $1 \leq p \leq i$ and $j \leq q \leq n$. In the example, if the correlation upper bound of the randomly selected cell $\{I_2, I_5\}$ is below $\theta = 40$, the cells in the gray area are all below

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		43	39	35	26	13
I_2			56	41	33	25
I_3				42	37	29
I_4					59	37
I_5						46
I_6						

Table 3.2: Type 1 Correlation Upper Bound Pattern

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		92	84	76	69	51
I_2			75	67	58	41
I_3				55	42	38
I_4					36	23
I_5						17
I_6						

Table 3.3: Type 2 Correlation Upper Bound Pattern

θ. For type 2 correlation measures, the upper bound value decreases from left to right and from top to bottom. If the upper bound of the current cell $\{I_i, I_j\}$ is below θ, then $CUB(I_p, I_q) < \theta$ when $i \leq p \leq n-1$ and $max(p+1, j) \leq q \leq n$. In the example, if the correlation upper bound of the randomly selected cell $\{I_2, I_5\}$ is below $\theta = 60$, the cells in the gray area are all below θ. For type 3 correlation measures, the rightmost column has the lowest upper bound value. If the upper bound of the current cell $\{I_i, I_j\}$ is below θ, then the upper bounds of any cell to its right is below θ. In the example, if the correlation upper bound of the randomly selected cell $\{I_2, I_5\}$ is below $\theta = 40$, the cells in the gray area are all below θ.

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		53	41	37	23	17
I_2			41	37	23	17
I_3				37	23	17
I_4					23	17
I_5						17
I_6						

Table 3.4: Type 3 Correlation Upper Bound Pattern

3.2.4 Fully-correlated Itemset Framework

Correlation search for arbitrary size itemsets has two more problems than that for pairs. First, we need a way to rule out independent items from a given itemset. Second, we need a framework or a measure with downward-closed property to speed up the search. Since no existing correlation measures satisfying the three correlation properties can solve either of the above problems, we proposed the fully-correlated itemset framework [25].

Definition 1. *Given an itemset S and a correlation measure, if the correlation value of any subset containing no less than two items of S is higher than a given threshold θ, this itemset S is a fully-correlated itemset.*

This definition has two very important properties. First, it can be incorporated with any correlation measure for itemsets to impose the downward-closed property which can help us to prune unsatisfied itemsets quickly like Apriori [2]. If a given itemset S is not a fully-correlated itemset, it must contain a subset S' whose correlation is below the threshold. Since the subset S' is a subset of any superset of S, any superset of S is not a fully-correlated itemset either. With this framework, we only need to focus on effectiveness side when selecting correlation measures. Second, it helps to rule out an itemset with independent items. For example in Table 3.5,

	C		not C	
	B	not B	B	not B
A	25	25	25	425
not A	25	25	25	425

Table 3.5: An independent case

B is correlated with C, and A is independent from B and C. Suppose we use the likelihood ratio and set the correlation threshold to be 2. The likelihood ratio of the set $\{A, B, C\}$ is 8.88 which is higher than the threshold. But the likelihood ratio of its subset $\{A, B\}$ which is 0 doesn't exceed the threshold. According to our definition, the set $\{A, B, C\}$ is not a fully-correlated itemset. The only fully-correlated itemset in this example is $\{B, C\}$ whose likelihood ratio is 17.93.

3.3 Correlation Search

As the number of items and transactions in the data set increases, calculating the correlation values for all the possible pairs is already computationally expensive, and it is much harder to calculate correlation for all the possible itemsets. We propose several methods to handle the efficiency problem.

3.3.1 Correlated Pair Search

Correlated pair search is a special case of correlated itemset search. Unlike frequent itemset search which can start pruning for 1-itemsets, correlated itemset search even with the downward-closed property has to start pruning with 2-itemsets. Therefore, correlated pair search is the cornerstone of correlated itemset search. In general, there are two types of correlated pair searches: correlated pairs above a certain threshold and top-k correlated pairs. Although the correlation itself doesn't have a monotone property to help prune, the correlation upper bound does. Consequently, we make use of the 1-dimensional monotone property of the upper bound of

any good correlation measure, and different 2-dimensional monotone properties for different types of correlation measures. We can either use the 2-dimensional search algorithm to retrieve correlated pairs above a certain threshold, or our new Token-Ring algorithm to find the top-k correlated pairs, to prune many pairs without the need to compute their correlations. Although the 2-dimensional search algorithm can efficiently find correlated pairs above a given threshold, it is difficult for users to provide an appropriate correlation threshold in real-world applications since different data sets have different characteristics. Instead, we try to find the top-k correlated pairs. However, when using the Token-Ring algorithm to find the top-k correlated pairs, we could spend a lot of time on calculation and end up with trivial results in a data set that doesn't contain strong positive correlations. Therefore, we propose a user procedure to combine the 2-dimensional search algorithm and the Token-Ring algorithm to reconcile the two tasks.

3.3.1.1 Correlated Pairs above a Certain Threshold

In this subsection, we focus on the task of finding correlated pairs above a certain threshold. The easiest way of speeding up search is calculating the correlation upper bound before the actual correlation measure. If the upper bound is already below the threshold, there is no need to retrieve the pair support to calculate the correlation. Therefore, we greatly reduce IO cost for high thresholds. Upper bound checking needs to calculate the upper bound of all the possible $n(n-1)/2$ pairs which is still computationally expensive. The 1-dimensional property can be used to save a lot of unnecessary upper bound checks for very high thresholds. We specify the reference item A and start a search within each branch. The reference item A is fixed in each branch and it has the minimum support value due to the way we construct the branch.

Items in each branch are also sorted based on their support in increasing order. By Theorem 3.2, the correlation upper bound of $\{A, B\}$ monotonically decreases with the increase of the support of item B. Therefore, if we find the first item B, the turning point, which results in an correlation upper bound less than the user-specified threshold, we can stop the search for the current branch. If the upper bound is greater than the user-specified threshold, we calculate the exact correlation and check whether this pair is really satisfied. Furthermore, the 2-dimensional property can be used to prune cells faster. The key difference between 1-dimensional search and 2-dimensional search is that the 2-dimensional search algorithm records the turning point in the previous branch and starts computing from that point instead of the beginning of the current branch. But we will use different search sequences for three different types of correlation measures. For type 1 correlation measures, we start the search from the upper-left corner. If the upper bound of the current cell is above the threshold θ, we will check the cell to the right of it; else we will instead check the cell under it. Take the threshold 36 in Table 3.2 for example, the search sequence is (I_1, I_2), (I_1, I_3), (I_1, I_4), (I_2, I_4), (I_2, I_5), (I_3, I_5), (I_3, I_6), and (I_4, I_6). The upper bound of any cell below this boundary is greater than the threshold θ. For type 2 correlation measures, we start the search from the upper-right corner. If the upper bound of the current cell is greater than θ, we check the cell below the current cell; else, we check the cell to the left of the current cell. Take the threshold 45 in Table 3.3 for example, the search sequence is (I_1, I_6), (I_2, I_6), (I_2, I_5), (I_3, I_5), and (I_3, I_4). The upper bound of any cell above this boundary is greater than θ. For type 3 correlation measures, the rightmost column has the lowest upper bound value. We only need to search the first branch. If the current column is above the threshold, we continue the search of the right-side column until the current column is below the threshold.

3.3.1.1.1 Performance Analysis

Theorem 3.3. *The number of upper bound calculations in the 2-dimensional search is between* $n-1$ *and* $2n-3$ *for the first type,* $n-1$ *for the second type, and between* 1 *and* $n-1$ *for the third type.*

Proof. For the 2-dimensional search in type 1, the number of calculations is determined by the cell where we stop on the right most column. If the algorithm stops at the upper-right corner, the whole search moves from left to right $n-1$ times. If the algorithm stops at the lower-right corner, the whole search moves from left to right $n-1$ times and from up to down $n-2$ times, so there are $n-1+n-2=2n-3$ movements. Since the search might stop at any cell on the most right column, the calculation for type 1 is between $n-1$ and $2n-3$.

For the second type, the search starts at the upper-right corner and stops at one of the border cells along the diagonal, that is, cell $C_{i,i+1}$ where $i=1,2,...,n-1$. From the upper-right corner cell $C_{1,n}$ to the cell $C_{i,i+1}$, we have to make i movements from up to down and $n-i-1$ movements from right to left. In all, we need to make $i+n-i-1=n-1$ movements no matter in which cell we stop the search.

For the third type, we stop the search between the calculation for the first column and that of the last column, so the number of calculation is between 1 and $n-1$.

Different methods of facilitating correlated pair search above a certain threshold are discussed in this section. If we don't make use of correlation upper bound at all, we need to calculate the correlation for all the possible pairs to find correlated pairs above a threshold. This brute-force method will have $n(n-1)/2$ support IO cost and $n(n-1)/2$ correlation calculations. Here, we call a pair a candidate if its corre-

lation upper bound is greater than the user-specified threshold θ. For a given dataset and θ, the number of candidates, α, is fixed. For each candidate, we have to calculate its true correlation to determine whether its correlation is above the threshold or not. No matter which method we use, the IO cost of retrieving support and the computing cost of correlation for α candidates are inevitable. The only difference is the number of correlation upper bound checks. If it is just upper bound calculation, $n(n-1)/2$ upper bounds need to be calculated. If it is 1-dimensional search, α plus an extra β upper bounds need to be calculated where β is between 0 and $n-1$ depending on in how many branches we check all the cells. If it is 2-dimensional search, γ upper bounds need to be calculated which is between $n-1$ and $2n-3$ for type 1, $n-1$ for type 2, and between 1 and $n-1$ for type 3.

Here, we further study the difference between 1-dimensional and 2-dimensional search. If we want to find correlated pairs above a given threshold, α IO cost and correlation calculation is inevitable because each candidate must be checked. The difference is that 1-dimensional has $\alpha+\beta$ upper bound calculations and 2-dimensional has γ upper bound calculations. Since α is much greater than β and γ, and IO is much more expensive than calculation, the α IO cost dominates the computational cost. There is no significant difference between 1-dimensional and 2-dimensional search with regard to finding pairs above a threshold. However, it is hard to determine which threshold is proper for a given dataset. Normally, this threshold is determined by intuition. But we can use the number of candidates for the given threshold to do the evaluation. For example, we may know how much time we spend on retrieving support and calculating correlation for a pair and how long we are allowed to search. We can estimate how many candidates we can afford to search. When just finding the number of candidates, we can avoid the α support IO cost and α correlation

calculation. In this case, 2-dimensional search can find the number of candidates in linear time which is much better than 1-dimensional search.

3.3.1.2 Top-k Correlated Pairs

Although 2-dimensional search can efficiently find correlated pairs above the threshold θ, it is hard for users to provide a proper correlation threshold in many applications. Instead, finding top-k correlated pairs is more meaningful for users.

3.3.1.2.1 TOP-COP Search

The original TOP-COP algorithm [89] introduced a diagonal traversal method for efficiently searching the top-k correlated pairs of ϕ-coefficient. While it exploits the 2-dimensional monotone property of the upper bound of the ϕ-coefficient, it needs $O(n^2)$ space to save pair status indicating whether or not the pair needs to be checked. Saving pair status only takes 3% of the space of saving support, but it is still not feasible for large datasets. If items are sorted by their support in increasing order, the upper bound of pairs for any good correlation measure decreases from left to right for each row in Table 3.1. The search starts from the diagonal consisting of $\{I_i, I_{i+1}\}$ which is the closest to the main diagonal, then goes to the diagonal consisting of $\{I_i, I_{i+2}\}$ which is the second closest to the main diagonal, and so on. During the iterative search process, this method maintains a top-k list and a pair is pushed into this list if its correlation coefficient is greater than the current minimum correlation coefficient in the top-k list. The search stops if the maximal upper bound of all the pairs in a diagonal is less than the current minimum correlation coefficient in the top-k list.

3.3.1.2.2 Token-Ring Search

TOP-COP calculates all the pairs in a diagonal to find the maximal upper bound. In fact, the upper bound calculation of some pairs in the diagonal can be avoided if the upper bound of their left pair is already less than the current minimum correlation coefficient τ in the top-k list. In order to achieve that, we propose a token-ring search algorithm. We treat all the $n-1$ branches as nodes in the token-ring. Only the branch that gets the token can calculate the upper bound of the leftmost uncalculated pair in this branch and compare its upper bound against τ. If the upper bound is above τ, we will check the correlation value of this pair. This pair will be pushed into this list if its correlation coefficient is greater than τ. If the upper bound is below τ, this branch will be removed from the token-ring because the upper bounds of the uncalculated pairs in this branch must be less than τ according to the 1-dimensional property. In addition, a branch will also be removed from the token-ring if all the pairs in this branch are calculated. The algorithm will stop when there is no branch in the token-ring. According to the way token-ring search works, the upper bound of all the pruned pairs must be less than the current minimum correlation coefficient in the top-k list, so the current top-k list contains the pairs with the k highest correlation values in the data set.

We coined a data set with 6 items. The correlation upper bounds and correlations of all the pairs are shown in Tables 3.6 and 3.7 respectively. An example of the Token-Ring search for the top-5 pairs in this data set is shown as follows. After traveling the first diagonal from $\{I_1, I_2\}$ to $\{I_5, I_6\}$, all the five pairs are pushed into the top-5 list, and the current minimum correlation coefficient in the top-5 list is 15. Since the current $\tau = 15$ is less than the maximal upper bound in the current diagonal

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		15	14	13	8	5
I_2			36	23	14	9
I_3				48	31	19
I_4					49	31
I_5						45
I_6						

Table 3.6: Correlation upper bound of the coined data

	I_1	I_2	I_3	I_4	I_5	I_6
I_1		15	14	3	-2	1
I_2			36	3	12	0
I_3				48	2	10
I_4					36	29
I_5						20
I_6						

Table 3.7: Correlation of the coined data

49, we continue the search. When checking $\{I_1, I_3\}$, the upper bound 14 is less than the current minimum correlation coefficient in the top-5 list 15, so the first branch quits the Token-Ring. The upper bound of $\{I_2, I_4\}$ is 23 which is greater than 15, but its correlation is less than 15. Therefore, the second branch stays in the Token-Ring and the current top-5 list is not changed. A similar thing happens to $\{I_3, I_5\}$. After checking $\{I_4, I_6\}$, the current minimum correlation coefficient in the top-5 list is 20 and the maximal upper bound in the current diagonal is 31. Therefore, we continue the search in the third diagonal. Branch 1 is already pruned, so $\{I_1, I_4\}$ won't be checked. We only check $\{I_2, I_5\}$ and $\{I_3, I_6\}$. After traveling the third diagonal, the current minimum correlation coefficient in the top-5 list is 20 and the maximal upper bound in the current diagonal is 19. Finally, we stop the search at cell $\{I_3, I_6\}$.

Theorem 3.4. *Both TOP-COP and Token-Ring calculate the same number of correlations. The only performance difference between TOP-COP and Token-Ring is that Token-Ring reduces the number of correlation upper bound computations.*

Proof. The correlation upper bound check for the cell $\{I_i, I_{i+j}\}$ in the i-th branch and the j-th diagonal line can be avoided if the correlation upper bound of the cell $\{I_i, I_{i+j-1}\}$ in the i-th branch and the $(j-1)$-th diagonal line is below the minimum correlation coefficient in the top-k list at that time. Therefore, Token-Ring avoids the upper bound checks of TOP-COP for those pairs that cannot affect the current top-k list, but the current top-k list changes at the same cells for both Token-Ring and TOP-COP. In all, both TOP-COP and Token-Ring calculate the same number of correlations, but they check a different number of correlation upper bounds.

Theorem 3.5. *For the Token-Ring algorithm, the difference between the number of computed correlation upper bounds and the number of computed correlations is no more than* $n-1$.

Proof. For the Token-Ring algorithm, there are $n-1$ branches at the beginning. If the i-th branch is pruned because all the pairs in this branch are calculated, we calculate the same number of upper bounds as correlations. If the i-th branch is pruned because the upper bound is lower than the current minimum correlation coefficient in the top-k list, we calculate one more upper bound than correlations for this branch. Therefore, the difference between the number of computed correlation upper bounds and the number of computed correlations is no more than $n-1$.

3.3.1.3 Combination of Two Tasks

The 2-dimensional search can efficiently find correlated pairs above θ, but it is difficult for users to provide an appropriate correlation threshold. When using the Token-Ring algorithm to find the top-k correlated pairs, we could spend a lot of time on calculation and end up with trivial results in a data set that doesn't contain strong positive correlations. In the extreme case, any item in a transaction appears alone, which means there is no positive correlation for any pair. Both TOP-COP and Token-Ring will calculate the correlations of all the pairs and end up with the trivial top-k correlated pairs. In order to solve this problem, we propose a user procedure to combine 2-dimensional search and Token-Ring search which can stop searching wisely on its way finding the top-k correlated pairs when there are few strong positive correlations in the data set.

Before we try to find the correlated pairs, we can calculate the number of candidates for a given threshold and use this number to estimate the time we will spend. For example, if we hope the algorithm is completed in one day and the time for processing one candidate is 20ms, then we should choose the threshold under which the number of candidates is less than 4 million. Due to the existence of 2-dimensional search, we can get the number of candidates for any threshold in the linear time $O(n)$. We can choose the lowest threshold θ_{min} under which the number of candidates is less than 4 million, and then search the correlated pairs by using 2-dimensional search. The advantage is that the algorithm will stop on time if the data set contains few strong positive correlations. However, there are two disadvantages. First, we might retrieve too many pairs and we are only interested in the top-k correlated pairs. Second, we will definitely spend one day to search the correlated pairs, while we

might only need to spend one hour to find the top-k correlated pairs by using the Token-Ring algorithm. Instead of searching the correlated pairs above θ_{min} by the 2-dimensional search algorithm, we integrate the threshold θ_{min} into the Token-Ring algorithm. We treat all the $n-1$ branches as nodes in the token-ring. Only the branch which gets the token can calculate the upper bound of the leftmost uncalculated pair in this branch and compare its upper bound against the value $\psi = max\{\theta_{min}, \tau\}$ where τ is the current minimum correlation coefficient in the top-k list. If the upper bound is above ψ, we will check the correlation value of this pair. This pair will be pushed into this list if its correlation coefficient is greater than ψ. If the upper bound is below ψ, this branch will be removed from the token-ring. The algorithm will stop when there is no node in the token-ring. In that way, even if the current minimum correlation coefficient τ in the top-k list never exceeds the threshold θ_{min}, the algorithm will process all the candidates under the threshold θ_{min} which won't cost more than the maximal time we allow. It stops searching wisely on its way to finding the top-k correlated pairs when there are few strong positive correlations in the data set. If there are a lot of strong positive correlations in the data set, the algorithm will find the top-k list and stop. For example, if we try to find the top-5 pairs and set up the threshold 40 in the coined data set shown in Tables 3.6 and 3.7, the search sequence is as follows: $\{I_1, I_2\}$, $\{I_2, I_3\}$, $\{I_3, I_4\}$, $\{I_4, I_5\}$, $\{I_5, I_6\}$, $\{I_3, I_5\}$, and $\{I_4, I_6\}$. We end up with the only strong pair $\{I_3, I_4\}$.

3.3.2 Correlated Itemset Search

In a large dataset, it is impossible to calculate correlation for all the possible itemsets in order to retrieve the top-k correlated itemsets or correlated itemsets above a threshold. Even the best measures, such as leverage and likelihood ratio, only

penalize itemsets with independent items, but cannot detect whether a given itemset contains items that are independent of the others. Therefore, we make use of the fully-correlated itemset framework [25] for correlated itemset search. Given a fully-correlated itemset S, if no other item can be added to generate a new fully-correlated itemset, then S is a *maximal fully-correlated itemset*. MFCIs provide more compact information for FCI as maximal frequent itemset for frequent itemset.

When going through the Apriori procedure, we will discard the $(k-1)$-level information after we generate the k-level information. However, if we generate a fully-correlated k-itemset given a correlation threshold θ and keep that information, we know that this itemset satisfies any correlation threshold lower than θ. To achieve that, we build an enumeration tree to save the fully-correlated value for all the MFCIs under a given initial correlation threshold. We can either efficiently retrieve the desired MFCIs for any given threshold above the initial threshold or incrementally grow the tree if the given threshold is below the initial threshold.

Definition 2. *Given an itemset S, the fully-correlated value $\rho(S)$ for this itemset is the minimal correlation value of all its subsets.*

Given an itemset S and the correlation threshold θ, if the current correlation threshold $\theta \leq \rho(S)$, the current itemset is a fully-correlated itemset because the correlation of any subset is no less than θ, i.e., any subset is highly correlated. If the current correlation threshold θ is greater than $\rho(S)$, the current itemset is not a fully-correlated itemset because the correlation of at least one subset is lower than θ, i.e., at least one subset is uncorrelated.

Theorem 3.6. *Given a k-itemset S ($k \geq 3$), the fully-correlated value (FCV) $\rho(S)$ for this itemset is the minimal value α among its correlation value and the fully-*

correlated values for all its $(k-1)$*-subsets.*

Proof. For any true subset SS of the current k-itemset S, SS must be the subset of at least one of all the $(k-1)$-subset of the current itemset S. Among all the subsets of S, either S itself or one true subset SS has the minimal correlation value β.

(i) If the S itself has the minimal correlation value β, then the correlation of any true subset SS is no less than β. According to the definition of fully-correlated values, $\rho(S) = \beta$ and $\rho((k-1)\text{-}Subset_i) \geq \beta$. According to the definition of α in this theorem, $\alpha = \beta$. Therefore, $\rho(S) = \alpha$.

(ii) If one true subset SS has the minimal correlation value β, then the correlation of S and any true subset is no less than β. According to the definition of fully-correlated values, $\rho(S) = \beta$ and $\rho((k-1)\text{-}Subset_i) \geq \beta$. Since SS must be the subset of at least one of all the $(k-1)$-subset of the current itemset S, $\rho((k-1)\text{-}Subset_j) = \beta$ at least for one j. According to the definition of α in this theorem, $\alpha = \beta$. Therefore, $\rho(S) = \alpha$.

By making use of the above theorems, we can calculate the fully-correlated value for each itemset by Algorithm 3.1.

An enumeration tree will be used to store the fully-correlated value. This enumeration tree is commonly used for itemset search, and more details can be found in [11]. If the fully-correlated value of the current node is less than a given threshold θ, the fully-correlated value of any descendant of the current node is also lower than θ which can be avoided when traversing the tree. Then we can easily traverse this tree to retrieve all possible fully-correlated itemsets given a threshold. Since finding the maximal fully-correlated itemsets from the enumeration tree saving fully-correlated value information is exactly the same as finding the maximal frequent itemsets from

Algorithm 3.1 Find the Fully-Correlated Value

Main: CalculateFullyCorrelatedValue()

$\rho(pairs) = CalCorrelation(pairs)$;
for $k = 3; k \leq n; k++$ **do**
  for each possible k-itemset C **do**
    for $i = 1; i \leq k; i++$ **do**
      θ_i is the fully-correlated value of the i-th (k-1)-subset of C
    end for
    $\rho(C) = min\{CalCorrelation(C), \theta_1, \theta_2, ..., \theta_k\}$
  end for
end for

the enumeration tree saving support information, we can apply any technique of searching maximal frequent itemsets to MFCI search, such as MAFIA [16], Max-Miner [8], and GenMax [40]. Here, we use MAFIA to find the MFCIs given a threshold.

3.3.2.1 The incremental enumeration tree generation

Given threshold θ, we can save all the fully-correlated itemsets with FCV greater than θ in this enumeration tree structure. Therefore, we can modify the original MFCI algorithm as follows. Given the threshold θ, instead of only getting the fully-correlated itemsets for each level, we keep all the candidates generated from lower levels and their fully-correlated values in the enumeration tree T. Although we keep all the candidates generated from low levels, we only use the fully-correlated itemsets instead of the candidates in the current level to generate the next-level candidates. For any threshold θ_1 greater than θ, we can easily get the corresponding fully-correlated itemsets and the corresponding enumeration tree T_1 by traversing the current enumeration tree T. For a threshold $\theta_2 < \theta$, the current enumeration tree T is a subtree of the target enumeration tree T_2. In this case, we only need to increment the current enumeration tree T instead of building the target enumeration tree T_2

from scratch. To generate k-itemset candidates from fully-correlated $(k-1)$-itemsets, the candidates generated by the fully-correlated itemsets whose fully-correlated value is greater than θ have already been saved in the enumeration tree T. In order to generate the remaining candidates, we generate the candidates involving at least one $(k-1)$-itemset whose fully-correlated value is between θ and θ_2 to increment the enumeration tree T. In this way, we can generate the target enumeration tree T_2 and avoid repeating calculations that were made when building the enumeration tree T.

3.3.2.2 User Interaction Procedure

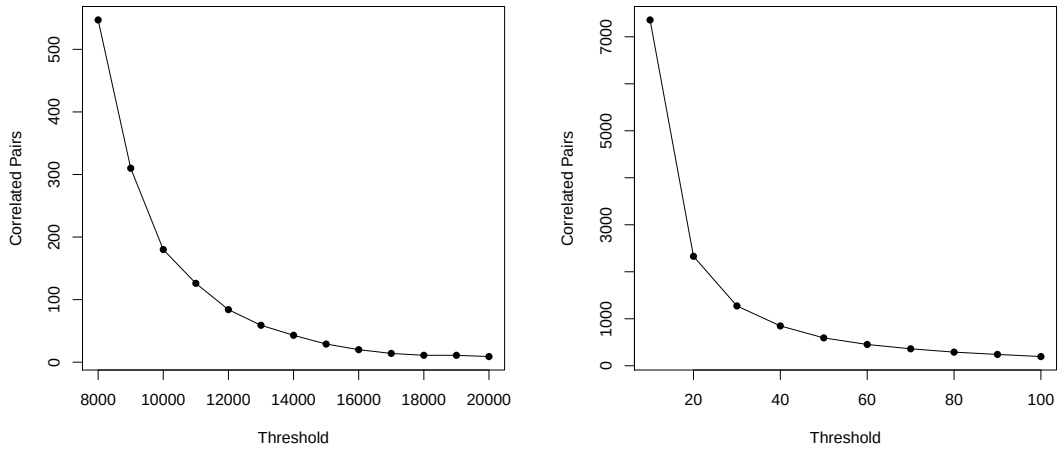
The best way to get the desired MFCIs is to select a relatively low threshold θ to build a corresponding enumeration tree to keep the fully-correlated values first. Then for any threshold above θ, we can easily generate the corresponding MFCIs from this enumeration tree. The core problem is how to determine this relative low threshold θ. If the threshold is too high, there might be some MFCIs which we are interested in but are not contained in the enumeration tree. We have to do extra work to increment the current enumeration tree. If the threshold is too low, we will keep much unnecessary information and might not have enough memory to generate or save the enumeration tree. Therefore, several proper user interaction procedures are needed in order to select the proper threshold.

We split the enumeration tree construction into two steps. First, we keep relevant pair correlation information for a relatively low threshold θ. By doing that, we can get the satisfied pairs for any threshold higher than θ by a simple query. Second, we construct the enumeration tree by trying different thresholds. Using the 2-dimensional search algorithm, we can easily count the number of pair candidates whose correlation upper bound is above a tentative threshold. Since we need to re-

trieve the support of all the pair candidates to calculate their correlation, we can estimate the time we will spend in the first step by the count for any given threshold. We will choose the threshold under which the computational time is affordable to us. For the second step, it is hard to estimate the time we will spend for any given threshold. The best way to finish the second step is to choose a relatively high threshold first, and then gradually grow the enumeration tree by lowering the threshold until we run out of time to update the tree for a threshold.

3.4 Experiments

The efficiency performance of our methods was tested on several real-life datasets. Since the results were similar for all the datasets, we only present results on two typical datasets. The first is the Netflix data set which contains 17,770 movies and 480,000 transactions. The second is a retail data set from the FIMI repository containing 16,470 items and 88,000 transactions. Netflix contains many correlated patterns because of movies in the same series and TV shows of many episodes, while the retail data set contains fewer correlated patterns because those highly correlated items might be already offered by manufactures as a package. The number of correlated pairs under different likelihood ratio thresholds for these two data sets is shown in Figure 3.1. We implemented our algorithm using Java 1.6.0 on a Dell workstation with 2.4GHz Dual CPU and 4G memory running on the Vista operating system. In the following, we will check the performance improvement from each algorithm.



(a) Netflix (b) Retail

Figure 3.1: The number of correlated pairs under different likelihood ratio thresholds

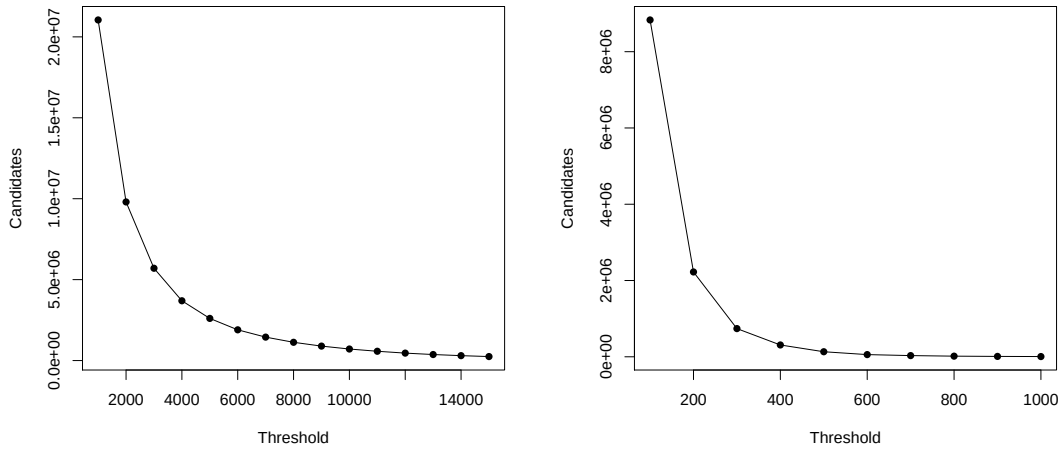
3.4.1 Correlated pair search

3.4.1.1 Finding correlated pairs above a certain threshold

Here, we focus on the task of finding correlated pairs above a given threshold θ.

3.4.1.1.1 Count the number of pair candidates

Before we determine the threshold to search the satisfied pairs, we can count how many pairs' CUBs are above a tentative threshold. This number can help us to estimate the time we will spend on the satisfied pair search because the support of the pair needs to be retrieved if its CUB is above the threshold. The number of candidates and the time spent on counting candidates under different thresholds are shown in Figure 3.2 and 3.3 respectively. To get the number of pair candidates, 2-dimensional search is linear and 1-dimensional search is exponential with the threshold. When the threshold is 0, the 1-dimensional search will check all the pairs. Therefore, the



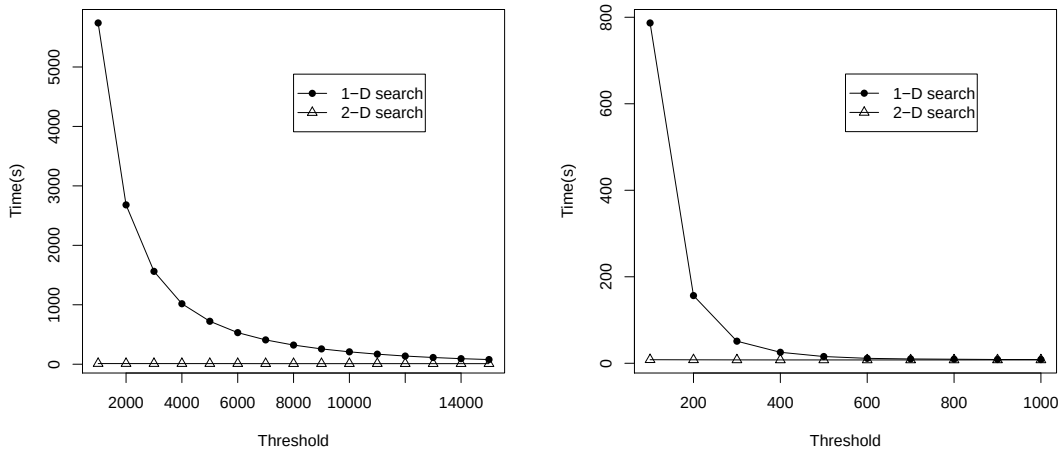
(a) Netflix (b) Retail

Figure 3.2: The number of candidates under different thresholds

runtime of 1-dimensional search with threshold 0 is equal to the runtime of the upper bound calculation method. The 1-dimensional search takes less time than the upper bound calculation method when the threshold is high, but the 2-dimensional search is always an order of magnitude faster than the just upper bound calculation method.

3.4.1.1.2 Search the satisfied pairs

Since the IO cost for calculating correlation of pairs is much higher than the time cost for calculating CUB, the CUB calculation can save a lot of unnecessary IO cost when the threshold is high. The runtime of retrieving the satisfied pairs by calculating CUB first given different correlation thresholds is shown in Figure 3.4. The 1-dimensional and 2-dimensional search are almost identical. The runtime of the CUB, 1-dimensional, and 2-dimensional search all decreases drastically as we increase the threshold. Both 1-dimensional and 2-dimensional search take much less time than the CUB search when the threshold is high because 1-dimensional and



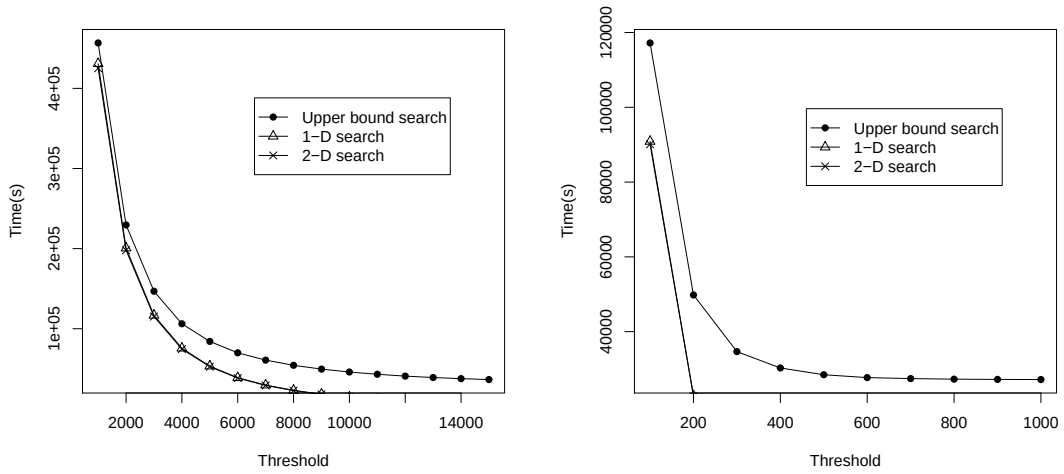
(a) Netflix (b) Retail

Figure 3.3: The runtime of determining the number of candidates under different thresholds

2-dimensional search take less time to find the pairs whose CUB is higher than the threshold. However, the CUB, 1-dimensional, and 2-dimensional search do not make too much difference when the threshold is low.

3.4.1.2 Finding top-k correlated pairs

For finding the top-k correlated pairs, the runtime of the brute-force method is determined by the number of correlations we need to compute, while the runtime of TOP-COP or Token-Ring is determined by the number of correlations and the number of CUBs we need to compute. The brute-force method will calculate the correlation of all the possible pairs which is not feasible for large datasets. According to Theorem 3.4, Token-Ring computes fewer CUBs than TOP-COP, but computes the same number of correlations. Therefore, the runtime difference between Token-Ring and TOP-COP is determined by the difference between the number of CUB



(a) Netflix (b) Retail

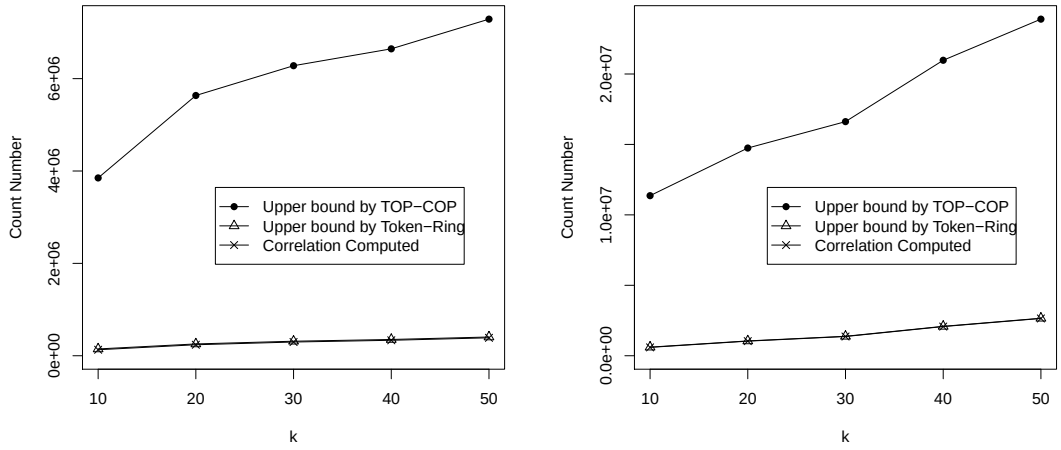
Figure 3.4: The runtime for retrieving the satisfied pairs

calculations. The number of correlation calculations and CUB calculations under different k is shown in Figure 3.5. According to Theorem 3.5, the number of correlation calculations is almost equal to that of CUB calculations in the Token-Ring algorithm which is verified in Figure 3.5. Token-Ring can save a huge number of unnecessary CUB checks compared to TOP-COP. The runtime of TOP-COP and Token-Ring for top-k correlated pairs is shown in Figure 3.6. When the IO cost of retrieving pair support is much more expensive than the CUB calculation for a dense dataset like Netflix, the total runtime is dominated by the time spent on correlation calculation and there is little difference between Token-Ring and TOP-COP. When the IO cost is not that expensive compared to the CUB calculation for a sparse dataset like retail, we can see Token-Ring took significantly less time than TOP-COP. The advantage of Token-Ring over TOP-COP is obvious only for datasets where the gap between the true correlation and the correlation upper bound is large.

The top-k search method is more flexible for finding the desired patterns compared to the 2-dimensional method. Assuming we can get the k-top pair correlation ahead of time and use it to search the top-k pairs with the 2-dimensional method, Figure 3.7 shows the number of correlation calculations using 2-dimensional search and those using our top-k search method. The gap between the 2-dimensional method and the top-k search method is not huge and the gap gradually increases with the k. The top-k search method is more flexible and does not require much more time than the threshold search method.

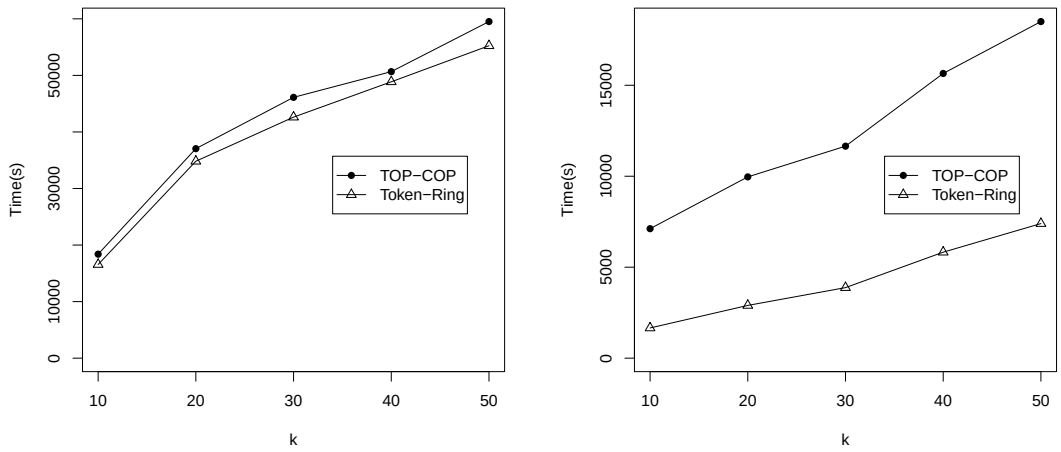
3.4.1.3 Combination of Two Tasks

We proposed a user procedure to combine the 2-dimensional search algorithm and the Token-Ring algorithm to reconcile two tasks of searching for pairs above a certain threshold and for top-k pairs. The performance of the user procedure really depends on the parameters we set. If the threshold θ is so high that the k-th correlation in the dataset is less than θ, the runtime is equal to the 1-dimensional search under the threshold θ because the modified Token-Ring checks the same number of correlations and the same number of CUBs as the 1-dimensional search algorithm. If k is small and the data set contains many strong positive correlations, the runtime is close to the original Token-Ring algorithm. From Figures 3.4 and 3.6, we can easily estimate the runtime of the modified Token-Ring for different parameters. For example, when the threshold θ is 1000 and k is 10 in Netflix, the runtime will be close to 18,000 seconds.



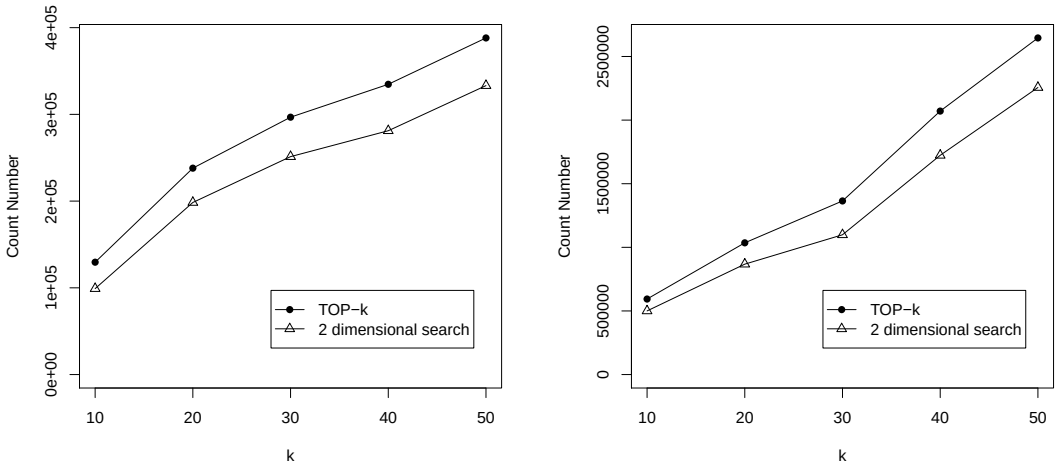
(a) Netflix (b) Retail

Figure 3.5: The number of correlation and correlation upper bound checks



(a) Netflix (b) Retail

Figure 3.6: The runtime for top-k algorithms



(a) Netflix (b) Retail

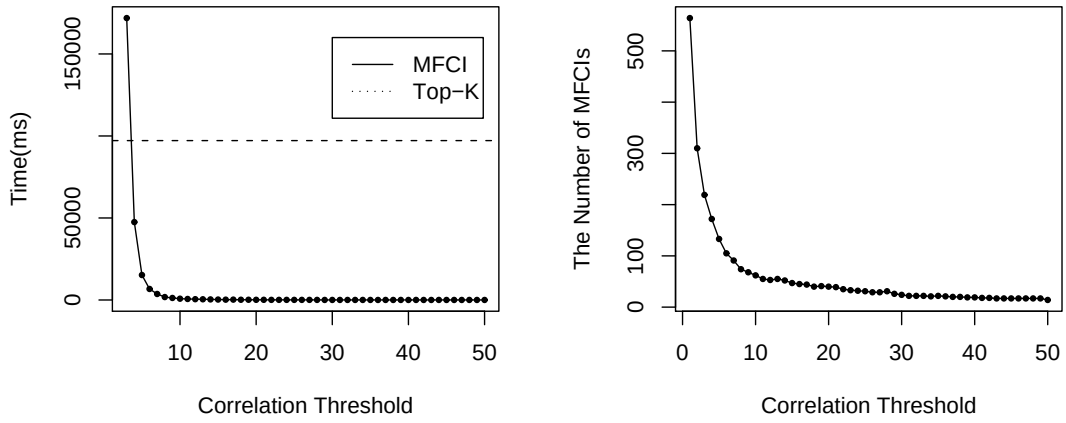
Figure 3.7: Correlation checks for top-k search and threshold search

3.4.2 Correlated itemset search

3.4.2.1 Maximal Fully-correlated Itemset Search

Since it is impossible to find the top-k correlated itemsets on both datasets and we want to compare the top-k correlated itemsets with MFCIs, we use a subset of the Netflix dataset which only contains the first 100 movies according to the code sequence in the Netflix dataset and test on it. The performance of our algorithm depends on the characteristic of the data set. In the extreme case, among all the n items in the data set, if the first $(n-1)$ items always show up together and the remaining item appears alone, our algorithm will start with $\binom{n-1}{2}$ 2-itemsets and end with the only $(n-1)$-maximal fully-correlated itemset. In other words, all the $2^{(n-1)}$ possible combinations will be checked. It is still an NP-hard problem. But in reality, most data sets are sparse, so most of the search space can be pruned.

The runtimes of our algorithm on the Netflix subset data given different cor-



(a) Runtime for each threshold

(b) MFCIs for each threshold

Figure 3.8: Performance results for Netflix

relation thresholds are shown in Figure 3.8(a). The runtime decreases drastically as we increase the threshold. The runtime of the top-k method is also shown. When running the top-k method, we only checked the itemsets which occurred at least once in the dataset, so about 2 million itemsets were checked instead of all 2^{100} possible itemsets. In the worst case, our algorithm checks all the itemsets that occur at least once. Therefore, the runtime of our algorithm is at least as good as the top-k method if both use Apriori. Even though we use the prefix tree structure [11] to find the top-k correlation itemsets, the runtime of our method is less than the top-k method when the threshold is larger than 3. In addition, the number of maximal fully-correlated itemsets corresponding to each correlation threshold is shown in Figure 3.8(b). It shows even if we use a small threshold like 1, we still get a relatively small number of compact itemsets.

Besides the gain from the efficiency side, we can also make use of the character-

istics of the Netflix dataset to evaluate the effectiveness of the MFCI framework. Like maximal frequent itemsets, there is no ranking of maximal fully-correlated itemsets. Rather, we only get a different number of them under different thresholds. Therefore, we try to find the threshold under which only 15 MFCIs are retrieved to compare with top-20 correlation itemsets which are retrieved by calculating every possible combination. The 15 maximal fully-correlated itemsets are shown in Table 3.8, and the top-20 correlated sets are listed in Table 3.9. Our maximal fully-correlated itemsets have several advantages over the top-20 correlated sets. First, some top-k correlated sets are redundant since they are subsets of other top-k correlated sets. For example, the first correlated set of Netflix is a subset of the second one. There is no need to show redundant information. Second, some top-k correlated itemsets contain irrelevant information. Among all the top-20 correlated itemsets, 16 of them are subsets of the 15 maximal fully-correlated itemsets. Four remaining itemsets all contain one more movie "Something's Gotta Give (2003)" than the corresponding maximal correlation sets. In fact, "Something's Gotta Give (2003)" is the most favored movie among all the 100 films. It almost has no correlation with any other movie. If we remove "Something's Gotta Give (2003)" from these 4 itemsets, we can get higher correlation values. For each of these four itemsets, if we treat "Something's Gotta Give (2003)" as set A and the remaining movies as set B, all the correlation values between A and B are close to 0 which means there is no correlation. Especially in the 17th correlation set, {Dragonheart (1996), Congo (1995)} and {Something's Gotta Give (2003)} are negatively correlated. The probability of favoring "Something's Gotta Give (2003)" is 55.28%. The conditional probability of favoring "Something's Gotta Give (2003)" given {Dragonheart (1996), Congo (1995)} is 47.02%. Since favoring "Dragonheart (1996)" and "Congo (1995)" will decrease the probability of favoring

"Something's Gotta Give (2003)", putting them together is not a wise choice. As mentioned above, the top-k method contains some redundant itemsets. One way to solve this problem is to only keep the itemsets that do not have any superset in the top-k list. However, because of the existence of irrelevant items like "Something's Gotta Give (2003)", the itemsets that have no superset within the top-k itemsets might contain irrelevant items. The maximal fully-correlated itemsets are more reasonable than the top-k correlated itemsets. We get similar patterns by testing other correlation measures. Some of the movies in MFCIs do not seem to go together due to the weak connection among a small set of items. However, if MFCIs are considered bad, the top-k correlated itemsets are even worse. Since it is impossible to compare MFCI with the top-k correlated itemsets on the whole dataset, we instead chose the traditional maximal frequent itemsets to compare with. We tried different thresholds and stopped when only five patterns are retrieved. Table 2.15 shows the 5 maximal frequent itemsets and the 5 MFCIs on the whole Netflix dataset. We naively assume that movies and TV shows in the same series are more correlated. MFCIs get better results than maximal frequent itemsets.

3.4.2.2 Improvement from the enumeration tree structure

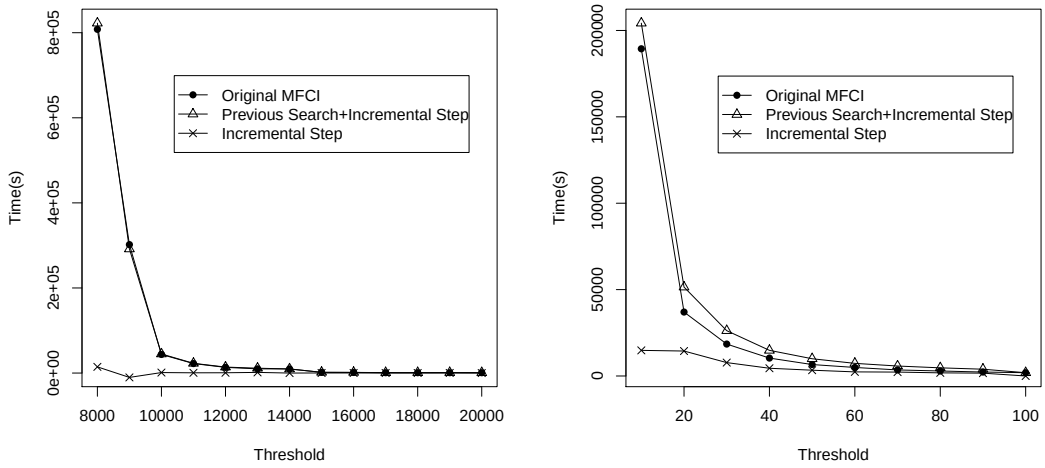
The performance improvement for MFCI comes from the improved pair search and the enumeration tree structure. We have checked the improvement from the improved pair search. In the following, we check the improvement from the enumeration tree structure. The fully-correlated values saved in the enumeration tree benefit the next search round. We need to build the tree first, and then handily retrieve MFCIs according to fully-correlated values saved in the enumeration tree.

ID	Maximal Fully-Correlated Itemsets
1	Character (1997), Mostly Martha (2002)
2	My Favorite Brunette (1947), The Lemon Drop Kid (1951)
3	7 Seconds (2005), Never Die Alone (2004)
4	Immortal Beloved (1994), Richard III (1995)
5	Aqua Teen Hunger Force: Vol. 1 (2000), Invader Zim (2004)
6	Rudolph the Red-Nosed Reindeer (1964), Jingle All the Way (1996)
7	The Bad and the Beautiful (1952), The Killing (1956)
8	Richard III (1995), The Killing (1956)
9	Dragonheart (1996), Jingle All the Way (1996)
10	The Rise and Fall of ECW (2004), WWE: Armageddon (2003), WWE: Royal Rumble (2005)
11	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005), ECW: Cyberslam '99 (2002)
12	Screamers (1996), Dragonheart (1996), Congo (1995)
13	Immortal Beloved (1994), Spitfire Grill (1996), Silkwood (1983)
14	Lilo and Stitch (2002), Justice League (2001), The Powerpuff Girls Movie (2002)
15	Lilo and Stitch (2002), Dragonheart (1996), Congo (1995)

Table 3.8: Maximal fully-correlated itemsets from Netflix subset using likelihood ratio

Ranking	Correlated Itemset
1	Dragonheart (1996), Congo (1995)
2	Lilo and Stitch (2002), Dragonheart (1996), Congo (1995)
3	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005)
4	Spitfire Grill (1996), Silkwood (1983)
5	My Favorite Brunette (1947), The Lemon Drop Kid (1951)
6	Immortal Beloved (1994), Spitfire Grill (1996), Silkwood (1983)
7	Screamers (1996), Dragonheart (1996), Congo (1995)
8	Lilo and Stitch (2002), Dragonheart (1996)
9	Lilo and Stitch (2002), Something's Gotta Give (2003), Dragonheart (1996), Congo (1995)
10	The Rise and Fall of ECW (2004), WWE: Royal Rumble (2005), ECW: Cyberslam '99 (2002)
11	Aqua Teen Hunger Force: Vol. 1 (2000), Invader Zim (2004)
12	Something's Gotta Give (2003), Spitfire Grill (1996), Silkwood (1983)
13	The Bad and the Beautiful (1952), The Killing (1956)
14	Rudolph the Red-Nosed Reindeer (1964), Jingle All the Way (1996)
15	Immortal Beloved (1994), Richard III (1995)
16	Immortal Beloved (1994), Something's Gotta Give (2003), Spitfire Grill (1996), Silkwood (1983)
17	Something's Gotta Give (2003), Dragonheart (1996), Congo (1995)
18	The Rise and Fall of ECW (2004), WWE: Armageddon (2003), WWE: Royal Rumble (2005)
19	The Rise and Fall of ECW (2004), ECW: Cyberslam '99 (2002)
20	Screamers (1996), Dragonheart (1996)

Table 3.9: Top-20 correlated itemsets for Netflix subset using likelihood ratio

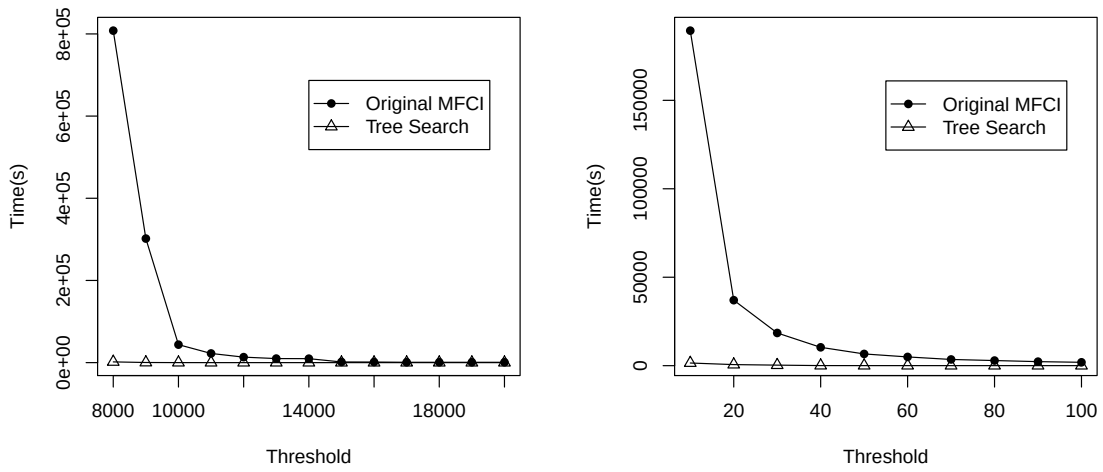


(a) Netflix when the threshold gap is 1000 (b) Retail when the threshold gap is 10

Figure 3.9: The runtime of building the enumeration tree

3.4.2.3 Build the enumeration tree

Since it is hard to estimate the time we will spend to build the enumeration tree for any given threshold, the best way to build the enumeration tree is to choose a relative high threshold first, and then gradually increment the enumeration tree by lowering the threshold. Given the threshold θ and the threshold gap g, the time to build the enumeration tree T for the threshold θ is a, the time to build the enumeration tree T_1 for the threshold $(\theta + g)$ is b, and the time for incrementing the tree from T_1 to T is c. Therefore, the runtime of the original MFCI is a, and the total runtime of the incremental algorithm is $(b+c)$. The runtime of the original MFCI and the incremental algorithm given the threshold gap g and different threshold is shown in Figure 3.9. The total runtime of these two algorithms is very close, but the incremental algorithm is more flexible for user interactions.



(a) Netflix (b) Retail

Figure 3.10: The runtime for generating MFCIs

3.4.2.4 Generate MFCIs from the enumeration tree

The enumeration tree saving fully-correlated values has exactly the same downward-closed property of the enumeration tree saving support, so we can apply any search technique for maximal frequent itemsets. Here, we only show how much improvement we get if we use MAFIA to generate MFCIs from the enumeration tree. Figure 3.10 shows the runtime of the original MFCI algorithm and the MFCI generation from the enumeration tree. The MFCI generation from the enumeration tree is much faster than the original MFCI algorithm.

3.4.2.5 User Interaction Procedure

If the goal is to build a relatively large enumeration tree which can facilitate the search of MFCIs under different thresholds, we use the following procedure. First, 2-dimensional search is used to evaluate the threshold θ under which time allows to

retrieve all the correlated pairs. Second, CUB search is used to find the correlated pairs and save their correlation value in the database. Third, we choose a relatively high threshold, and then gradually update the enumeration tree by lowering the threshold until we run out of time to grow the tree or lowering the threshold. After that, we can easily retrieve MFCIs under any threshold above the threshold of the enumeration tree.

If the goal is to identify the highly correlated patterns, we can combine 2-dimensional search with the incremental algorithm for the enumeration tree. From Figure 3.4 and 3.9, we expect the runtime saving mainly comes from the 2-dimensional search.

3.5 Conclusion

In the chapter, we propose an FCI framework to decouple the correlation measure from the need for efficient search. By wrapping the desired measure in our FCI framework, we take advantage of the desired measure's superiority in evaluating itemsets, make use of the property of FCI to eliminate itemsets with irrelevant items, and still achieve good computational performance. Further, we facilitated search for both pairs and itemsets. For pairs, we can either use the 2-dimensional search to retrieve correlated pairs above a certain threshold, or our new Token-Ring algorithm to find top-k correlated pairs. For itemsets, we build an enumeration tree to save the fully-correlated value. In that way, we can either efficiently retrieve the desired FCIs for any given threshold above the initial threshold or incrementally grow the tree if the given threshold is below the initial threshold.

CHAPTER 4
CORRELATION-BASED NETWORK COMMUNITY DETECTION

4.1 Introduction

The modern science of graphs (alternatively networks) has significantly helped us understand complex systems. One important feature of graphs is community structure where many edges join vertices of the same community and comparatively few edges join vertices of different communities. Such communities can be considered as fairly independent components of a graph and play a role like the organs in the human body. Community detection has a long history with sociology, biology, and computer science where systems are often represented as graphs. The original study on graphs is Euler's solution [31] of the puzzle of Konigsberg's bridges in 1736. A lot of studies on graph mathematical properties [10] have been made since then. Social network analysis started in the 1930's and remains one of the most important topics in sociology [75]. With the development of information techniques, we are able to capture the large scale of actual individual activities at the micro-level. The size of real graphs has grown to millions or even billions of vertices, which requires new methods to handling large-scale graphs [61].

Community structure has many different forms in graphs from sociology, biology, computer science, engineering, politics, etc. For social networks, we have group organizations such as families, working and friendship, towns, and nations. The diffusion of Internet leads to the creation of online communities [35]. In protein-protein interaction networks, communities are likely to be a group of proteins with the same function in the cell [17]. In the World Wide Web, they might be related to groups of pages of the same topics [24]. In metabolic networks, they might correspond to

functional modules such as cycles and pathways [41].

Community detection is the process of identifying the modules and, possibly, the hierarchical or overlapping structure by using only the graph topology. The first research on community detection was made by Weiss and Jacobson [88]. They studied the matrix of working relationships between members of a government agency. Work groups were found by removing the members working with people in different departments. The cutting bridge idea is the basis of several modern community detection methods. In 2002, Girvan and Newman proposed a new algorithm [39] aiming at the identification of edges lying between communities. Two years later, they proposed a measurement called modularity [62] to measure the quality of detected communities. However, some heuristic methods [18] aiming at maximizing modularity also find good results.

We are also concerned with two issues related to evaluating community detection results. First, testing an algorithm means applying it to a specific problem which has a known solution and comparing the solution with that delivered by the algorithm. Since community detection is unsupervised learning and we don't have the ground truth to test the performance of different methods for real datasets, we need to simulate a network with a realistic community structure. Second, we need a reliable testing measure to judge how close the detected results are to the setting that we use to simulate the network data.

The rest of this chapter is organized as follows. Section 2 introduces the classical community detection methods and explores the opportunity for us to improve their results through correlation. The simulation of network data is discussed in Section 3. We investigate the way of evaluating the detected results according to the ground truth we have for the simulated data in Section 4. Finally, we talk about the

future research plan in Section 5.

4.2 Community Detection Methods

Community detection is intuitive but not well defined. However, the widely accepted idea of communities is that there must be more edges inside the community than edges linking vertices of the community with the rest of the graph. Because of the ambiguous concept of community, people have proposed many different objectives to optimize. Given a subgraph S of a graph G with $n_s = |S|$ and $n = |G|$ vertices, k_v^{int} is the internal degree of vertex $v \in S$ as the number of edges connecting v to other vertices of S and k_v^{ext} is the external degree. If $k_v^{ext} = 0$, the vertex only has neighbors within S, which is likely to be a member of community S. If $k_v^{int} = 0$, the vertex is disjoint from S and it should be assigned to another community.

Currently, there are four categories of methods: cut-based, spectral-based, density-based, and correlation-based community detection.

1. Cut-based methods identify communities in a graph by detecting the edges that connect different communities. The communities get disconnected from each other by removing these edges. They just perform hierarchical clustering on the graph data. Being hierarchical clustering techniques, the results are represented by dendrograms. The early cut-based methods use conductance [10]. The conductance $\phi(S)$ of the subgraph S is $\frac{c(S,G\backslash S)}{min(k_S,k_{G\backslash S})}$ where $c(S, G\backslash S)$ is the cut size of S, and k_S, $k_{G\backslash S}$ are the total degree of S and of the rest of the graph $G\backslash S$ respectively. The low value of the cut size and the large value of the denominator are required to minimize conductance. The most popular algorithm on this cut-based category was proposed by Girvan and Newman [39] which started a new era of community detection. They raised the concept of between-

ness which is related to the edge frequency of a certain process. Historically, edge betweenness was introduced by Anthonisse [3]. Although Girvan and Newman considered three alternative definitions: geodesic edge betweenness, random-walk edge betweenness, and current-flow betweenness, these definitions share the same idea, and we only discuss geodesic edge betweenness without the loss of generality. They calculate the number of shortest paths between all vertex pairs that run through the edge. It is intuitive that intercommunity edges have a large value of the edge betweenness because many shortest paths connecting vertices of different communities will pass through them. This method iteratively removes the highest betweenness edge that has not been removed. Then, we get the dendrogram of disconnected subgraphs. Tyler et al. [84] proposed a faster version of the Girvan-Newman algorithm. Instead of starting from each vertex and computing the contribution to betweenness from all paths starting at that vertex, Tyler et al. calculate the contribution to edge betweenness only from a limited number of randomly chosen nodes. Empirical tests indicate that to pick the log number of total vertices is enough. Another fast version of the Girvan-Newman algorithm proposed by Rattigan et al. [72] uses a network structure index [71] to approximate the edge betweenness values. Counting all possible shortest path in the calculation of betweenness may lead to unbalanced partitions. Chen and Yuan [17] proposed to count only non-redundant paths whose endpoints are all different from each other.

2. Spectral-based methods use the eigenvectors of the adjacency matrix for community detection. Donath and Hoffman [23] proposed the first spectral-based methods. In the same year, Fiedler [33] found the eigenvector of the Laplacian

matrix can obtain a bipartition of the graph with low cut. The Laplacian is by far the most used matrix in spectral-based methods. Though there is no unique convention which matrix is exactly called the graph Laplacian [54], one commonly used Laplacian is calculated as follows. Given the adjacency matrix W of the graph G, we calculate the matrix D where the diagonal element d_{ii} is equal to $\sum_{j=1}^{n}(w_{ij})$ and non-diagonal elements are 0. The Laplacian matrix L [63,76] is equal to $D - W$. If the graph G has k mutually disconnected components, the Laplacian matrix has k zero eigenvalues. Usually the graph we are dealing with is one component and the Laplacian matrix only has one zero eigenvalue. Therefore, we choose k eigenvectors corresponding to the k smallest eigenvalues to transform the original adjacency matrix, and then apply any clustering method like k-means on the transformed matrix. The spectral-based methods are popular because the change of representation induced by eigenvectors makes the community structure more obvious.

3. Density-based methods try to find the communities within which vertices are tightly connected with each other. We define the internal-community density $\delta_{int}(S)$ of the subgraph S as the ratio of the number of internal edges of S to the number of all possible internal edges, i.e. $\delta_{int}(S) = \frac{k_{int}(S)}{n_s*(n_s-1)/2}$ where $k_{int}(S)$ is the number of internal edges of S. Similarly, the external-community density $\delta_{ext}(S) = \frac{k_{ext}(S)}{n_s*(n-n_s)}$. For S to be a community, we expect large $\delta_{int}(S)$ and small $\delta_{ext}(S)$. Searching for the best tradeoff between $\delta_{int}(S)$ and $\delta_{ext}(S)$ is the goal of density-based methods. A simple way of doing that is to maximize the sum of the difference $\delta_{int}(S) - \delta_{ext}(S)$ over all the communities [57].

4. The most famous correlation-based community detection method is related to

maximazing modularity [62] which is originally introduced as a stopping criterion for the cut-based method [39]. However, it has rapidly become an essential element of many clustering methods. The modularity function has several variants, but these variants share the same idea. Without the loss of generality, we introduce the original modularity-based method [20]. Given a graph with n nodes and m links represented by the adjacency matrix W, the expected number of edges falling between two nodes i and j is $k_i \cdot k_j/(2m)$ under the assumption of independence where k_i is the degree of node i. The modularity Q is calculated as $\frac{1}{2m}\sum_{ij}(w_{ij} - \frac{k_i \cdot k_j}{2m}) \cdot \delta(v_i, v_j)$. It is the sum of the difference between the actual number of edges and the expected number of edges over all the pairs of nodes in the same community. $\delta(v_i, v_j)$ is the Kronecker delta function whose value is equal to 1 if v_i and v_j are in the same community and 0 otherwise. Initially, each node is the only member of its own community. The original algorithm iteratively joins the two communities that increase the modularity most in the current round. The original algorithm will stop if the best merge cannot further increase modularity. Modularity is by far the most used and the best community detection method. It has been proved that modularity optimization is an NP-complete problem [12]. The commonly used optimization techniques for modularity are greedy search [20, 60, 85], simulated annealing [42, 58], extremal optimization [9, 28], and genetic algorithms [44, 69]. In addition, modularity can be easily extended to different forms. For networks with weighted edges, we only need to replace degrees with the sum of weights. In fact, weights can also be assigned to the edges of an undirected network by using any measure of correlation between vertices. In this way, we can use weighted modularity to detect communities with a potentially better exploitation of the network struc-

ture [38,91]. Modularity can also be extended to directed networks [4,51]. If an edge is directed, the probability oriented in in(out)-directions depends on the in(out)-degrees of the end vertices.

According to empirical study [34], modularity-based methods yield better results in the general setting. The modularity function calculates the difference between the actual number of edges inside communities and the expected number of edges inside communities if communities are randomly partitioned. Higher modularity means the given partition is less likely to be a random partition. Since correlation analysis also searches patterns which deviate from expectation under the assumption of independence, we can see the connection between the modularity function and correlation analysis. All the existing modularity-based methods use different optimization techniques to maximize the modularity function. Instead of searching other optimization techniques for the modularity function, we investigate the opportunity to improve the modularity function from the correlation analysis perspective in this chapter.

4.3 Network Simulation

When a community detection algorithm is designed, we need to compare it with other methods. Since the ground truth for real datasets is impossible to retrieve completely, we investigate the classical simulation procedures here.

The first work on this problem is called the planted L-partition model [21]. The model partitions a graph with $n = g * l$ vertices in l groups with g vertices each. Vertices of the same group are linked with a probability p_{in}, whereas vertices of different groups are linked with a probability p_{out}. If $p_{in} > p_{out}$, the intra-cluster edge density exceeds the inter-cluster edge density. Then the graph has a community structure which is quite intuitive. A seemingly different benchmark is the relaxed

caveman graph [86] to simulate the clustering properties of social networks. The starting point is a set of disconnected cliques. With the probability p, edges are rewired to link different cliques. Such models are smooth variations of the graph with perfect communities. To some extent, the model is equivalent to the planted L-partition model, where $p_{in} = 1 - p$ and $p_{out} = p$.

However, by using the planted L-partition model, all vertices have approximately the same degree and all communities have exactly the same size. In real social network data, degree distributions are usually skewed, with many vertices with low degree and a few vertices with high degree. A similar heterogeneity is also observed in the distribution of cluster size. Bagrow [5] introduced a model with power-law degree distributions. It starts with the Barabasi-Albert scale free graphs [6], and then vertices are randomly assigned to one of four equal size communities. At last, pairs of edges between two communities are rewired so that either edge ends up with the same community, without changing the degree of each vertex. Suppose we have edges a_1-b_1 and a_2-b_2, where a_1, a_2 belong to community A and b_1, b_2 belong to community B. The edges are replaced by a_1-a_2 and b_1-b_2. With this rewiring procedure, we can arbitrarily vary the edge density within and between clusters, and keep the degree of each vertex.

Recently, Lancichinetti et al. [50] introduced a very popular LFR model. They assume that the distributions of degree and community size are power laws with τ_1 and τ_2 respectively. Each vertex shares a fraction of $1-u$ of its edges with the vertices in the same community and a fraction of u with the vertices in the other communities, where u is the mixing parameter. The simulation procedure is as follows:

1 A set of community sizes s_j following the predefined power law parameter τ_2 is

generated.

2 A set of node degrees k_i following the predefined power law parameter τ_2 is generated. The internal degree of each node is $(1-u)k_i$ where u is the mixing parameter.

3 In the beginning, nodes are not assigned to any community. For each node, it will be assigned to a randomly chosen community which has empty spots to accept a new node. If the community size exceeds the internal degree of the node, the node enters the community; otherwise, it enters a waiting list.

4 For each node in the waiting list, we let the node enter a random community whose size exceeds the node's internal degree and randomly kick one node in the selected community out to the waiting list. We do this step iteratively until the waiting list is empty.

5 We enforce the condition on the fraction of internal degree and external degree. The rewiring procedure in [5] is performed when needed.

As we have seen, nearly all existing benchmark graphs are inspired by the planted L-partition model, to some extent. However, the model needs to be redefined to provide a good description of real graphs with community structure. The hypothesis that the linking probabilities of each vertex with the vertices of its community or of the other communities are constant is not realistic. It is more plausible that each pair of vertices i and j has its own linking probability p_{ij}, and that such probabilities are correlated for vertices in the same cluster. Since the LFR model is the closest model to reality by far, we will use the networks generated by this model to test our algorithms. However, the constraint used in the LFR model to assign the internal degree of each node in the second step is problematic because the condition imposed

by a fixed u cannot guarantee $p_{internal} > p_{external}$ which must be satisfied for a community structure. For a node A in a community with n' nodes in a graph with n nodes, u must be smaller than $1 - n'/n$ to guarantee $p_{internal} > p_{external}$. Therefore, we use the following constraint to assign the internal degree of each node in the second step: $p_{internal} = \beta \cdot p_{external}$, where β is the ratio to control the community structure and must be greater than 1.

4.4 Community Detection Evaluation

Evaluating a community detection algorithm involves a criterion to measure how similar the partition result of the algorithm is to the partition we hope to find. For two paritions $X = (X_1, X_2, ..., X_{n_x})$ and $Y = (Y_1, Y_2, ..., Y_{n_y})$ of a graph, X is determined by the algorithm with n_x communities and Y is the ground truth with n_y communities. Most evaluation measures can be divided into three categories: pair counting, community matching, and information theory.

Measures based on pair counting depend on the number of pairs which are successfully (unsuccessfully) classified in the same (different) communities according to the ground truth. Let SS be the number of pairs in the same community in both X and Y partitions, SD be the number of pairs in the same community in X but different communities in Y, DS be the number of pairs in different communities in X but the same community in Y, and DD be the number of pairs in different communities in both X and Y partitions. The Rand Index [70] is the ratio of the number of pairs correctly classified to the total number of pairs, $RI = (SS + DD)/(SS + SD + DS + DD)$. Since DD is usually greater than SS and putting nodes not in the same community in different partitions is not very challenging, Jaccard proposed a measure focusing on SS. The Jaccard Index [46] is the ratio of the number of vertex pairs in

the same community in both partitions to the number of pairs in the same community in at least one partition, $JI = SS/(SS + SD + DS)$.

Measures based on community matching aim at the largest overlaps between pairs of communities of different partitions. The most popular measures for this category are Purity, Inverse Purity, and their harmonic mean (F measure). Purity [96] focuses on the frequency of the most common Y_j into each X_i, $Purity = \sum_{i=1}^{n_x} \frac{|X_i|}{n} max_j \frac{X_i \cap Y_j}{X_i}$. Purity penalizes the noise in X partitions, but it does not reward grouping objects in the same community in Y partitions. If we simply make each node a community, we can trivially get a maximum purity value. Similarly, we can calculate Inverse Purity as: $InversePurity = \sum_{j=1}^{n_y} \frac{|Y_j|}{n} max_i \frac{X_i \cap Y_j}{Y_j}$. Inverse Purity rewards grouping objects in the same community in Y partitions, but doesn't penalizes the noise in X partitions. A more robust metric [78] combines the concepts of Purity and Inverse Purity using F measure [74]: $F = \sum_{i=1}^{n_x} \frac{|X_i|}{n} max_j \frac{2 * \frac{X_i \cap Y_j}{X_i} * \frac{X_i \cap Y_j}{Y_j}}{\frac{X_i \cap Y_j}{X_i} + \frac{X_i \cap Y_j}{Y_j}}$.

The third class of measures is based on the framework of information theory [56]. The idea is that we need little information to infer one partition given the other if two partitions are similar. Mutual Information MI measures the amount of information by which our knowledge about the communities in one partition increases when we are told what the communities are in the other partition. It is calculated as follows: $MI = \sum_i \sum_j P(X_i \cap Y_j) log \frac{P(X_i \cap Y_j)}{P(X_i)P(Y_j)}$. The minimum of MI is 0 if the X partition is random with respect to the Y partition. However, given a partion Y, all partitions derived from Y by further partitioning have the same mutual information with Y, even though they are different from each other. In this case, the mutual information is equal to the entropy $H(Y) = -\sum_j P(Y_j) log(P(Y_j))$. To avoid that, Danon et al. [22] proposed the normalized mutual information: $NMI = \frac{2 * MI}{H(X) + H(Y)}$. It is currently often used and reaches its maximal value 1 if the X partition is identical

to the Y partition.

4.5 Community Detection from Correlation Perspective

In this section, we introduce an improvement to modularity-based community detection from the correlation perspective.

4.5.1 Modularity-based Community Detection

The modularity function has several variants, but these variants share the same idea. Without the loss of generality, we introduce the original modularity-based method [20]. Given a graph with n nodes and m links represented by the adjacency matrix W, the expected number of edges falling between two nodes i and j is $k_i \cdot k_j/(2m)$ under the assumption of independence where k_i is the degree of node i. The modularity Q is calculated as $\frac{1}{2m}\sum_{ij}(w_{ij} - \frac{k_i \cdot k_j}{2m}) \cdot \delta(v_i, v_j)$. It is the sum of the difference between the actual number of edges and the expected number of edges over all the pairs of nodes in the same community. $\delta(v_i, v_j)$ is the Kronecker delta function whose value is equal to 1 if v_i and v_j are in the same community and 0 otherwise. Initially, each node is the only member of its own community. The original algorithm iteratively joins the two communities that increase the modularity most in the current round. The original algorithm will stop if the best merge cannot further increase modularity.

4.5.2 Connecting Modularity-based Community Detection with Correlation Analysis

In this section, we transform the modularity function and connect it with correlation measures. Given a partition with l groups $\{G_1, G_2, ..., G_l\}$ for the graph G with n nodes and m links, the modularity Q is $\frac{1}{2m}\sum_{ij}(w_{ij} - \frac{k_i \cdot k_j}{2m}) \cdot \delta(v_i, v_j)$. For the

node v_q in the group G_p, k_q^{int} is the number of the nodes in the group G_p that connect to v_q. The partial modularity Q_p, which all the nodes in the group G_p contribute to the overall modularity function, is $\frac{1}{2m}\sum_{i\in G_p, j\in G}(w_{ij} - \frac{k_i \cdot k_j}{2m}) \cdot \delta(v_i, v_j)$.

Therefore,

$$\begin{aligned} Q_p &= \sum_{i\in G_p, j\in G} \frac{w_{ij}\cdot\delta(v_i,v_j)}{2m} - \sum_{i\in G_p, j\in G} \frac{k_i\cdot k_j\cdot\delta(v_i,v_j)}{(2m)^2} \\ &= \sum_{i\in G_p} \frac{\sum_{j\in G} w_{ij}\cdot\delta(v_i,v_j)}{2m} - \sum_{i\in G_p} \frac{k_i\cdot\sum_{j\in G} k_j\cdot\delta(v_i,v_j)}{(2m)^2} \\ &= \sum_{i\in G_p} \frac{k_i^{int}}{2m} - \sum_{i\in G_p} \frac{k_i\cdot\sum_{j\in G_p} k_j}{(2m)^2} \\ &= \frac{\sum_{i\in G_p} k_i^{int}}{2m} - \frac{\sum_{i\in G_p} k_i}{2m}\cdot\frac{\sum_{j\in G_p} k_j}{2m}. \end{aligned}$$

It is easy to calculate that the total number of links inside G_p is $\sum_{i\in G_p} k_i^{int}/2$ and the total number of links in the graph G is m. If we randomly select a link from the graph G, the probability of the link inside G_p is $\frac{\sum_{i\in G_p} k_i^{int}/2}{m}$. Similarly, the probability of the link with at least one end inside G_p is $\frac{\sum_{i\in G_p} k_i}{2m}$ when we randomly select a link from the graph G. If the partition with l groups $\{G_1, G_2, ..., G_l\}$ for the graph G is totally random, the probability of the link with the other end inside G_p from the links with one end already inside G_p is $\frac{\sum_{i\in G_p} k_i}{2m}\cdot\frac{\sum_{j\in G_p} k_j}{2m}$. Therefore, given a partition with l groups $\{G_1, G_2, ..., G_l\}$ for the graph G, if we randomly select a link from the graph G, the true probability of the link being inside G_p, tp, is $\frac{\sum_{i\in G_p} k_i^{int}}{2m}$, and the expected probability of the link being inside G_p under the assumption of independent partition, ep, is $\frac{\sum_{i\in G_p} k_i}{2m}\cdot\frac{\sum_{j\in G_p} k_j}{2m}$. Therefore, the partial modularity function Q_p can be rewritten as: $Q_p = tp - ep$. By comparing the correlation measure $Leverage(S) = tp - ep$, we can see the modularity function shares the same idea with the correlation measure Leverage. Since the other correlation measures are also functions of tp and ep, we can change the partial modularity function Q_p by using the

formula of other correlation measures. In the rest of the chapter, instead of using the term modularity, we use Simplified χ^2, Probability Ratio, Leverage, and Likelihood Ratio referring to the corresponding changed partial modularity function Q_p, and Leverage is the original modularity community detection method.

4.5.3 Upper Bound Analysis

In this section, we analyze the upper bound of different partial modularity functions Q_p inferred from four typical correlation measures: Simplified χ^2, Probability Ratio, Leverage, and Likelihood Ratio. According to the above correlation property, the measures reach their upper bound when tp is fixed and ep reaches its lower bound.

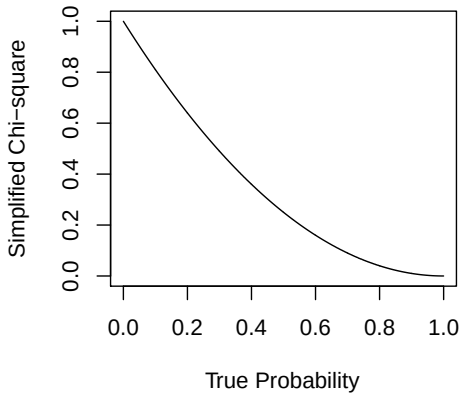
Given a partition with l groups $\{G_1, G_2, ..., G_l\}$ for the graph G, the true probability of a link being inside G_p, tp, is $\frac{\sum_{i \in G_p} k_i^{int}}{2m}$, and the expected probability, ep, is $\frac{\sum_{i \in G_p} k_i}{2m} \cdot \frac{\sum_{j \in G_p} k_j}{2m}$. If $\sum_{i \in G_p} k_i^{int}$ is fixed, the lowest possible value for $\sum_{i \in G_p} k_i$ is $\sum_{i \in G_p} k_i^{int}$ because $k_i^{int} \leq k_i$. In other words, when tp for the group G_p is fixed, the lowest possible value for ep is tp^2. When $ep = tp^2$, the measures reach their upper bound. Figure 4.1 shows the upper bounds of the various measures with respect to different tp for a single community. It is easy to see that different measures favor groups within different tp ranges. The upper bound of Simplified χ^2 increases to 1 and that of Probability Ratio increases to infinity when tp is close to 0, which means they favor extremely small groups rather than large groups. Leverage and Likelihood Ratio reach their highest upper bound when tp is between 0 and 1. According to the graph, Leverage does not favor the group which contains more than half of the edges in the graph since its upper bound starts to decrease even when the group size increases. Similarly, Likelihood Ratio does not favor the group which contains more

than roughly a quarter of the edges in the graph. In all, Probability Ratio favors the smallest groups, followed by Simplified χ^2, Likelihood Ratio, and Leverage.

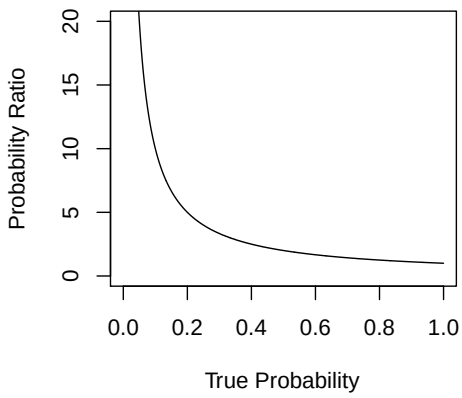
4.5.4 Ensembling Different Methods

Since different methods have different biases according to the analysis in Section 4.5.3, we can take advantage of the difference to ensemble them for better results. The raw value of different measures might be in the different scales. If we ensemble the raw value, the measure with the largest scale might dominate the final result. Instead, we try different ways to ensemble the ranking lists.

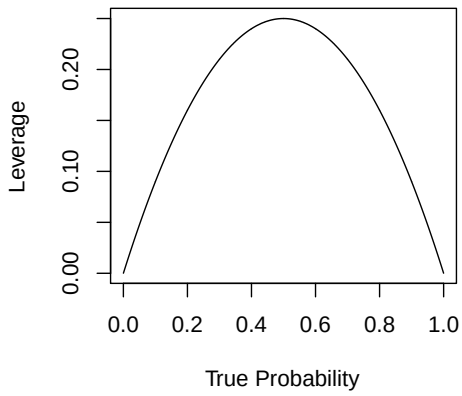
The algorithm uses the agglomerative clustering procedure. Two important steps affect the agglomerative clustering performance. First, we need to choose the most promising community pair to merge in each iteration. Since different measures rank the same pair differently, in order to choose the most promising pair, we can use three different functions to ensemble the ranking list: product, sum, and min. For example, given the community pair CP_i, the ranking given by method M_j is $R_{i,j}$. If we use the product function, the final value for CP_i is $\prod_{j=1}^{n} R_{i,j}$. Then, we choose the pair with the minimal final value as the most promising pair at each iteration. If there is a tie between different pairs, we use $min(R_{i,1}, R_{i,2}, ..., R_{i,n})$ to break the tie. Second, we need to decide when to stop the iteration. For the modularity-based algorithm, the algorithm will stop if the most promising merge cannot increase the modularity function any more. However, for ensemble algorithms, one objective function might increase but the other objective function might decrease for the most promising merge. Due to this conflict, we use two different stopping criteria. The first strategy is to stop the algorithm when any objective function starts to decrease. The second strategy is to stop the algorithm when all objective functions start to



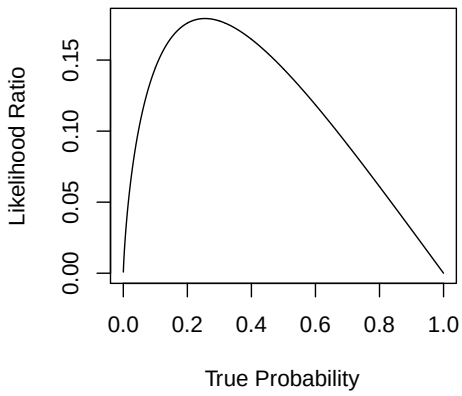
(a) Simplified χ^2



(b) Probability Ratio



(c) Leverage



(d) Likelihood Ratio

Figure 4.1: Upper bounds of different measures for a single community

decrease.

4.6 Experiments

Past research on modularity-based methods set modularity as their objective function and use different optimization techniques to search for the partition that generates the highest value. In this chapter, instead of exploring better optimization techniques for the same objective function, we change the objective function and study the impact the different objective functions make. In order to conduct a fair comparison for different objective functions, we choose greedy search, the simplest optimization technique, which is also used by the original modularity method and generates reasonably good results [20]. Initially, each node is the only member of its own community. The algorithm iteratively joins the two communities that increase the objective function most in the current round. The algorithm will stop if the best merge cannot further increase the objective function. Here, we conduct our experiments on both simulated and real life data sets.

4.6.1 Evaluation on Individual Methods

4.6.1.1 Results on Simulated Graphs

Here, we use the improved LFR model to simulate graphs and the normalized mutual information to measure the performance. There are 8 parameters related to the LFR simulation model: the total number of nodes, the minimal node degree, the maximal node degree, the power law parameter for node degree, the minimal community size, the maximal community size, the power law parameter for community size, and the ratio β for community structure. We conduct 9 sets of experiments and the parameter values are shown in Table 4.1. We only change the minimal community

Parameter	Value
The total number of nodes	2000
The minimal node degree	5
The maximal node degree	300
The power law parameter for node degree	2.5
The minimal community size	5, 50, or 100
The maximal community size	300
The power law parameter for community size	1.5
The ratio β for community structure	5, 10, or 20

Table 4.1: Parameter Setting for Simulated Graphs

size and the ratio β for community structure to generate different graphs. When the minimal community size is 5, there are many small communities, some mid-size communities, and a few large communities, while the graph only contains large communities when the minimal community size is 100. The community structure is fuzzy when β is 5, while it is clear when β is 20.

For each parameter setting, we generate the graph 10 times to test each method. We calculate the average value for each measure shown in Table 4.2[1]. Different evaluation measures provide different information. Since NMI is the most widely-accepted measure, we focus our discussion on NMI in the following. The average NMI and the average number of partitioned communities generated by each method for each parameter setting are shown in Figures 4.2 - 4.5.

Figure 4.2 shows the NMI achieved by each method and Figure 4.3 shows the number of partitioned communities when fixing the minimal community size and changing the ratio β. No matter what the minimal community size and the ratio β are, the NMI and the number of partitioned communities for both Simplified χ^2 and Probability Ratio are almost the same. They always detect more than 500 commu-

[1]DNC: the detected number of communities, ANC: the actual number of communities

Data Set	Measure	NMI	Jaccard	Rand Index	F-measure	DNC [2]	ANC [3]
MCS=5 β=5	Simplified χ^2	**0.5868**	0.0122	**0.9391**	0.0240	629.5	50.8
	Probability Ratio	0.5856	0.0062	0.9390	0.0124	903.3	50.8
	Leverage	0.1222	**0.0809**	0.7749	**0.1481**	9.4	50.8
	Likelihood Ratio	0.5515	0.0272	0.9388	0.0530	300.5	50.8
MCS=5 β=10	Simplified χ^2	**0.6023**	0.0146	0.9397	0.0289	604.8	51.2
	Probability Ratio	0.5937	0.0068	0.9394	0.0136	905	51.2
	Leverage	0.2741	**0.1523**	0.7900	**0.2605**	7.5	51.2
	Likelihood Ratio	0.5992	0.0462	**0.9406**	0.0883	263.4	51.2
MCS=5 β=20	Simplified χ^2	0.6212	0.0196	0.9436	0.0385	564.6	51.8
	Probability Ratio	0.6035	0.0089	0.9432	0.0177	855.7	51.8
	Leverage	0.5349	0.2265	0.8215	0.3658	6.8	51.8
	Likelihood Ratio	**0.7545**	**0.5136**	**0.9714**	**0.6699**	139.5	51.8
MCS=50 β=5	Simplified χ^2	**0.4775**	0.0091	**0.9177**	0.0181	586.2	16
	Probability Ratio	0.4765	0.0054	0.9176	0.0107	777.6	16
	Leverage	0.1172	**0.1016**	0.7754	**0.1820**	9.2	16
	Likelihood Ratio	0.4314	0.0194	0.9172	0.0381	283.7	16
MCS=50 β=10	Simplified χ^2	**0.5075**	0.0125	0.9240	0.0246	554.3	16.5
	Probability Ratio	0.4969	0.0069	0.9237	0.0137	747.9	16.5
	Leverage	0.4318	**0.2523**	0.8302	**0.3983**	6.2	16.5
	Likelihood Ratio	0.5040	0.0507	**0.9259**	0.0958	243.3	16.5
MCS=50 β=20	Simplified χ^2	0.5280	0.0154	0.9211	0.0303	533	15.8
	Probability Ratio	0.5065	0.0076	0.9205	0.0151	758.4	15.8
	Leverage	0.7375	0.4098	0.8886	0.5773	6.5	15.8
	Likelihood Ratio	**0.7663**	**0.6430**	**0.9703**	**0.7778**	67.9	15.8
MCS=100 β=5	Simplified χ^2	0.4210	0.0073	**0.8978**	0.0145	568.9	10.7
	Probability Ratio	**0.4243**	0.0044	0.8977	0.0088	750	10.7
	Leverage	0.1471	**0.1330**	0.7710	**0.2336**	8.5	10.7
	Likelihood Ratio	0.3727	0.0156	0.8972	0.0306	280.2	10.7
MCS=100 β=10	Simplified χ^2	0.4538	0.0092	0.9038	0.0183	571.9	11.3
	Probability Ratio	0.4514	0.0053	0.9036	0.0106	774.2	11.3
	Leverage	**0.5587**	**0.3423**	0.8532	**0.5038**	6.2	11.3
	Likelihood Ratio	0.4458	0.0287	**0.9046**	0.0558	250.8	11.3
MCS=100 β=20	Simplified χ^2	0.4844	0.0128	0.9016	0.0253	522.6	11.2
	Probability Ratio	0.4657	0.0069	0.9010	0.0138	721.9	11.2
	Leverage	**0.8318**	0.5755	0.9291	0.7300	6.8	11.2
	Likelihood Ratio	0.7614	**0.6550**	**0.9638**	**0.7851**	52.2	12

Table 4.2: Results on simulated datasets

nities. Since the total number of nodes is 2000, most of the communities they detect contain 2 or 3 nodes. That supports our observation in Section 4.5.3 that Simplified χ^2 and Probability Ratio favor small size communities. No matter what the minimal community size is, both Leverage and Likelihood Ratio achieve better NMI when the community structure becomes clearer. Only when the community structure is clear and the whole graph only contains large communities, the NMI of Leverage is better than that of Likelihood Ratio. Leverage has more bias towards large communities than Likelihood Ratio according to our upper bound analysis; therefore, we expect Leverage is better than Likelihood Ratio when the graph only contains large communities. In practice, social networks contain a lot of small communities; therefore, Likelihood Ratio is better in the common case. The number of partitioned communities by Leverage are almost the same no matter how we change the minimal community size and the ratio β, while the number of partitioned communities by Likelihood Ratio get closer to the ground truth when the community structure becomes clearer. Another interesting observation related to Leverage is that its NMI is very low when the minimal community size is large under the fuzzy community structure. Even under the fuzzy community structure, Leverage detects the same number of large communities. Such a partition assigns many nodes in different real communities to the same partitions, which results in the low NMI. Generally speaking, the partition generated by Likelihood Ratio is better and more adaptive to the different types of graphs than that by Leverage.

Figure 4.4 shows the NMI achieved by each method and Figure 4.5 shows the number of partitioned communities when fixing the ratio β and changing the minimal community size. Since both Simplified χ^2 and Probability Ratio favor small-size communities, their NMI decreases with the increase of the minimal community

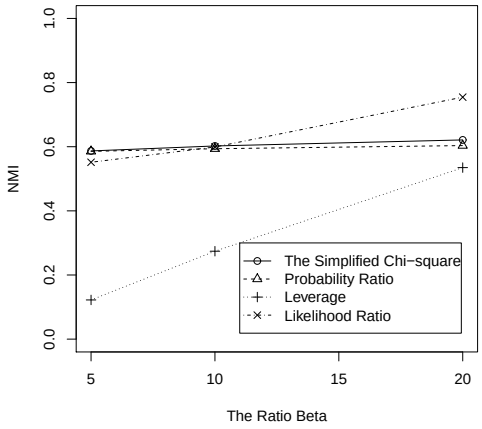
size whether the community structure is fuzzy or clear. The NMI of Likelihood Ratio decreases with the increase of the minimal community size when the community structure is not clear. However, when the community structure is clear, Likelihood Ratio achieves almost the same performance with the increase of the minimal community size. The NMI of Leverage always increases with the increase of the minimal community size since it has the bias to large communities.

4.6.2 Graph Parameter Estimation

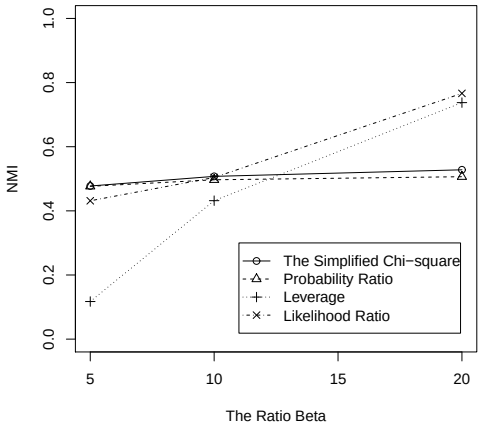
In the previous section, we checked the performance of different measures under different parameters. Likelihood Ratio is more robust. However, Leverage is better when there are only large communities and the community structure is clear. Given a real life dataset, we need a way to assess whether there are only large communities and the community structure is clear. And then, it helps us to choose the right function for community detection.

We calculate the number of k-cliques from each simulated graph (when k=2,3, and 4). If y is the number of k-cliques in the graph, we can find a following linear regression model: $y \sim k + \beta + minimal\_community\_size$. The regression function is $y = 21648.569 - 5851.667 * k + 91.23 * \beta + 1.867 * minimal\_community\_size$. The p-value of the coefficient for β is $2.48e-08$ and the p-value of the coefficient for $minimal\_community\_size$ is 0.465. Since the coefficient for β is significant, we can estimate the β by calculating the number of k-cliques from the graph (when k=2,3, and 4).

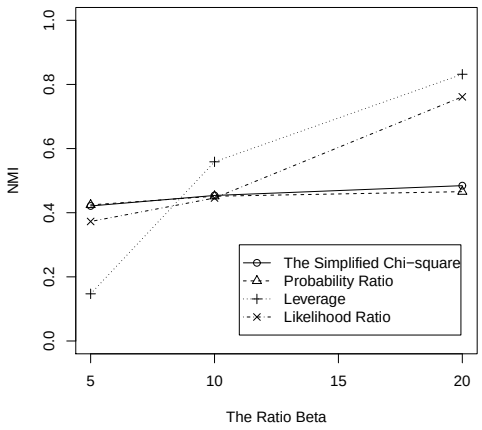
By checking the number of communities detected by Likelihood Ratio and Leverage, we find the number of communities detected by Likelihood Ratio is close to that of Leverage when there are only large communities and the community structure



(a) The minimal community size is 5

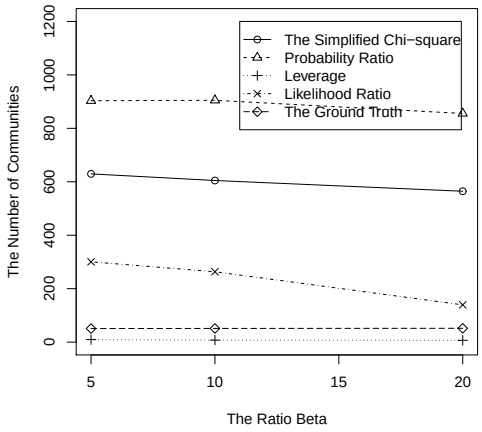


(b) The minimal community size is 50

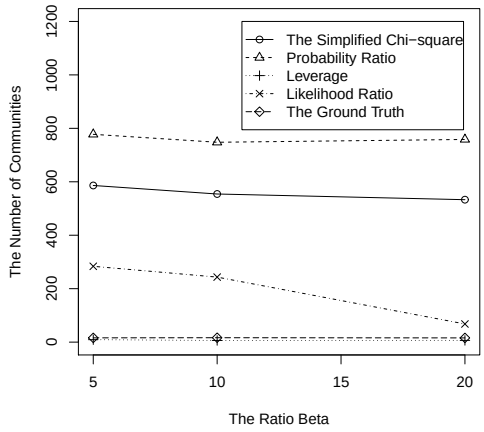


(c) The minimal community size is 100

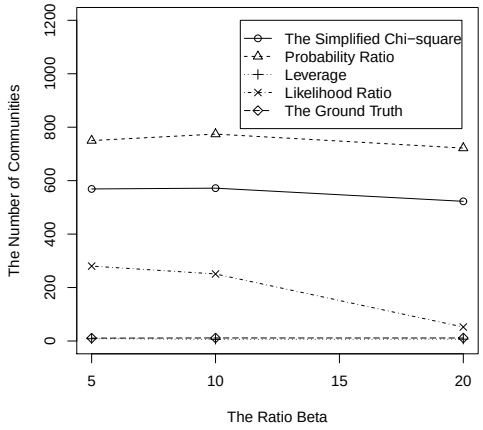
Figure 4.2: NMI when fixing the minimal community size



(a) The minimal community size is 5

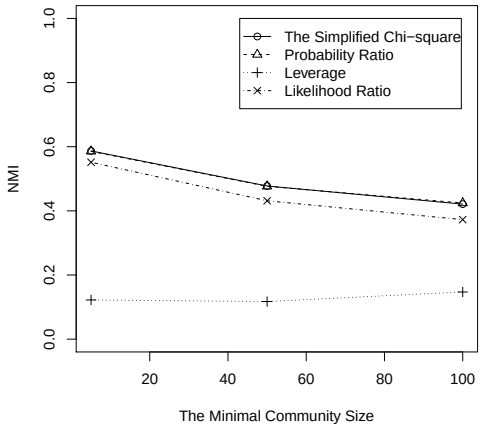


(b) The minimal community size is 50

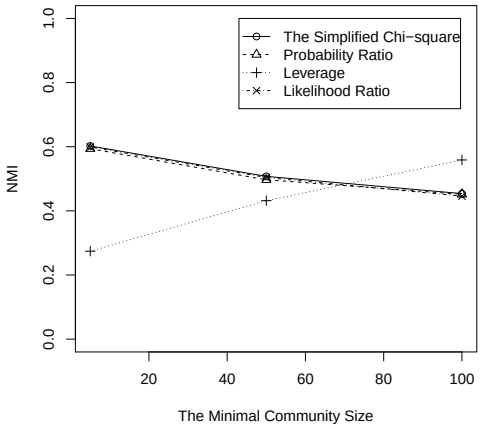


(c) The minimal community size is 100

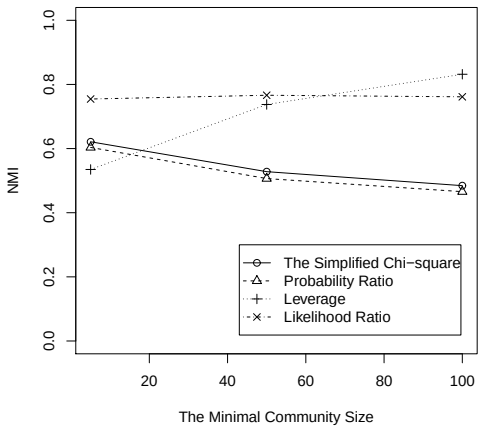
Figure 4.3: The number of communities when fixing the minimal community size



(a) The ratio β is 5

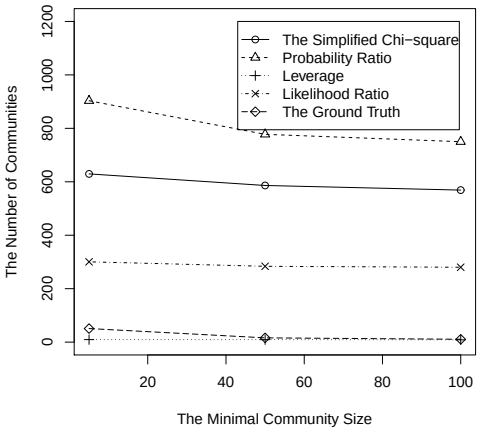


(b) The ratio β is 10

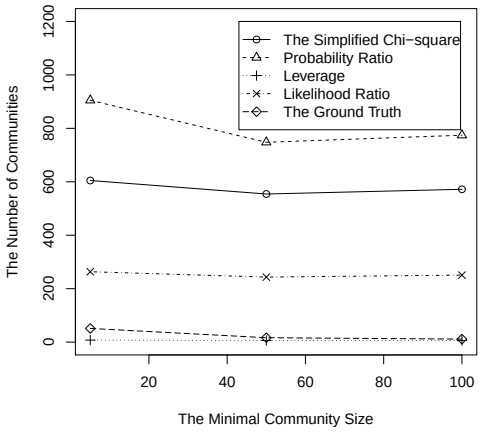


(c) The ratio β is 20

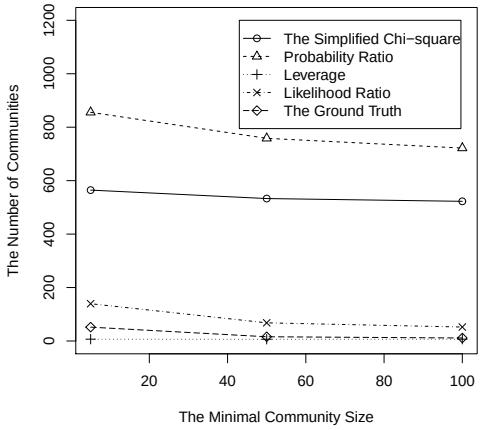
Figure 4.4: NMI when fixing the ratio β



(a) The ratio β is 5



(b) The ratio β is 10



(c) The ratio β is 20

Figure 4.5: The number of communities when fixing the ratio β

is clear. If the difference between the number of communities detected by Likelihood Ratio and Leverage is small and the estimated β is large, then $minimal\_community\_size$ must be large.

4.6.2.1 Results on Real Life Graphs

In this section, we conduct experiments on two real life datasets with manually identified community information: Karate [93], and Football [39]. The karate dataset contains friendships between 34 members of a karate club at a US university in the 1970s. There was a disagreement between the administrator and the instructor in the club, which resulted in two communities in this graph. The football dataset records games between Division IA colleges during regular season Fall 2000. There were 115 teams in 12 different conferences. We use different algorithms to partition graphs and evaluate how similar the partition is to the ground truth by different measures. The final result is shown in Table 4.3. The best method for Karate dataset is Leverage. The result is consistent with our theoretical analysis. The karate dataset only contains two large communities, and Leverage is the method having the most bias to large communities. Both Simplified χ^2 and Likelihood Ratio are good for the football dataset. This dataset contains 12 almost equal size communities. Likelihood Ratio has the bias to middle-size communities. According to its performance on the simulated data, it is not surprising that Likelihood Ratio works very well for the football dataset. However, Simplified χ^2 also works surprisingly well in the football dataset. According to Figure 4.1, it has the bias, but not the extreme bias, to small communities. When the dataset contains many clear middle-size communities, Simplified χ^2 can work well, but is not recommended because it is not as robust as Likelihood Ratio.

Data Set	Measure	NMI	Jaccard	Rand Index	F-measure	DNC	ANC
Karate	Simplified χ^2	0.4852	0.2842	0.6453	0.4426	7	2
	Probability Ratio	0.3868	0.0945	0.5561	0.1728	14	2
	Leverage	**0.6925**	**0.6833**	**0.8414**	**0.8118**	3	2
	Likelihood Ratio	0.5385	0.3958	0.6952	0.5671	5	2
Football	Simplified χ^2	**0.9141**	0.7571	0.9793	0.8618	14	12
	Probability Ratio	0.6864	0.0829	0.9240	0.1531	55	12
	Leverage	0.6977	0.3622	0.8807	0.5317	6	12
	Likelihood Ratio	0.9086	**0.7897**	**0.9812**	**0.8825**	12	12

Table 4.3: Results on real life datasets

4.6.3 Evaluation on Ensemble Methods

After checking the performance for each individual method, we examined the performance for ensemble methods. Among the four individual methods, Leverage and Likelihood Ratio are robust and have different biases. Therefore, we only check the ensemble of Leverage and Likelihood Ratio in the following. There are three ensemble functions and two stopping criteria. The results of different ensemble methods on the simulated data are shown in Table 4.4. Since Leverage tends to find large communities, Likelihood Ratio will stop the search before Leverage. Therefore, the stopping criterion is determined by Likelihood Ratio if the stopping criterion is "AtLeastOneNegative", or Leverage if the stopping criterion is "AllNegative". If the stopping criterion is "AtLeastOneNegative", the final performance is similar to Likelihood Ratio no matter which ensemble function we use. If the stopping criterion is "AllNegative", there are significant differences among three ensemble functions. The result of Min function is very similar to Likelihood Ratio, while the result of Sum function is similar to Leverage. The Product function is in the middle. When using the "AllNegative" stopping criterion, the ensemble result is at least better than the worst individual method. If we cannot estimate the situation of a given dataset, the ensemble method with the "AllNegative" stopping criterion can prevent us from getting the worst result.

Data Set	Function	Stop	NMI	Jaccard	Rand Index	F-measure	DNC	ANC
MCS=5 β=5	Product	AllNegative	0.5275	0.0400	0.9384	0.0760	236.9	50.8
	Product	AtLeastOneNegative	**0.5611**	0.0239	0.9389	0.0467	369.6	50.8
	Min	AllNegative	0.5506	0.0341	**0.9391**	0.0656	292.3	50.8
	Min	AtLeastOneNegative	0.5599	0.0248	0.9389	0.0485	352.3	50.8
	Sum	AllNegative	0.2942	**0.1487**	0.9122	**0.2543**	18.5	50.8
	Sum	AtLeastOneNegative	0.5537	0.0228	0.9384	0.0445	400.8	50.8
		Leverage	0.1222	0.0809	0.7749	0.1481	9.4	50.8
		Likelihood Ratio	0.5515	0.0272	0.9388	0.0530	300.5	50.8
MCS=5 β=10	Product	AllNegative	**0.6212**	0.2387	**0.9517**	0.3537	191.6	51.2
	Product	AtLeastOneNegative	0.6016	0.0418	0.9405	0.0802	310.3	51.2
	Min	AllNegative	0.6121	0.1228	0.9453	0.2155	246.6	51.2
	Min	AtLeastOneNegative	0.6006	0.0449	0.9406	0.0858	279	51.2
	Sum	AllNegative	0.5517	**0.4389**	0.9365	**0.6050**	13.1	51.2
	Sum	AtLeastOneNegative	0.5961	0.0432	0.9401	0.0826	356	51.2
		Leverage	0.2741	0.1523	0.7900	0.2605	7.5	51.2
		Likelihood Ratio	0.5992	0.0462	0.9406	0.0883	263.4	51.2
MCS=5 β=20	Product	AllNegative	**0.7931**	0.6977	0.9695	0.8061	79.3	51.8
	Product	AtLeastOneNegative	0.7535	0.4925	0.9705	0.6522	182.9	51.8
	Min	AllNegative	0.7882	**0.7176**	**0.9826**	**0.8276**	107.2	51.8
	Min	AtLeastOneNegative	0.7510	0.4893	0.9702	0.6472	168.9	51.8
	Sum	AllNegative	0.7490	0.4136	0.9226	0.5824	12.5	51.8
	Sum	AtLeastOneNegative	0.7650	0.5529	0.9735	0.7031	164.7	51.8
		Leverage	0.5349	0.2265	0.8215	0.3658	6.8	51.8
		Likelihood Ratio	0.7545	0.5136	0.9714	0.6699	139.5	51.8
MCS=50 β=5	Product	AllNegative	0.4087	0.0394	0.9173	0.0754	213.6	16
	Product	AtLeastOneNegative	**0.4421**	0.0175	0.9173	0.0344	342.6	16
	Min	AllNegative	0.4322	0.0294	**0.9177**	0.0568	275.2	16
	Min	AtLeastOneNegative	0.4410	0.0177	0.9173	0.0348	333.4	16
	Sum	AllNegative	0.2119	**0.1298**	0.8904	**0.2250**	18.9	16
	Sum	AtLeastOneNegative	0.4393	0.0174	0.9170	0.0342	368.5	16
		Leverage	0.1172	0.1016	0.7754	0.1820	9.2	16
		Likelihood Ratio	0.4314	0.0194	0.9172	0.0381	283.7	16
MCS=50 β=10	Product	AllNegative	**0.5257**	0.2557	**0.9398**	0.3836	145.4	16.5
	Product	AtLeastOneNegative	0.5112	0.0581	0.9267	0.1054	298.9	16.5
	Min	AllNegative	0.5178	0.1330	0.9318	0.2276	222.9	16.5
	Min	AtLeastOneNegative	0.5083	0.0439	0.9257	0.0834	304.4	16.5
	Sum	AllNegative	0.5217	**0.3861**	0.9222	**0.5507**	11.9	16.5
	Sum	AtLeastOneNegative	0.5050	0.0446	0.9252	0.0852	307.6	16.5
		Leverage	0.4318	0.2523	0.8302	0.3983	6.2	16.5
		Likelihood Ratio	0.5040	0.0507	0.9259	0.0958	243.3	16.5
MCS=50 β=20	Product	AllNegative	0.8267	0.6414	0.9581	0.7772	12.9	15.8
	Product	AtLeastOneNegative	0.7592	0.6152	0.9684	0.7562	108.1	15.8
	Min	AllNegative	0.8100	0.6846	0.9679	0.8096	32.5	15.8
	Min	AtLeastOneNegative	0.7310	0.5433	0.9620	0.6968	129.4	15.8
	Sum	AllNegative	**0.8713**	0.7191	0.9702	0.8340	11.8	15.8
	Sum	AtLeastOneNegative	0.8409	**0.7550**	**0.9797**	**0.8577**	56.5	15.8
		Leverage	0.7375	0.4098	0.8886	0.5773	6.5	15.8
		Likelihood Ratio	0.7663	0.6430	0.9703	0.7778	67.9	15.8
MCS=100 β=5	Product	AllNegative	0.3064	0.0449	0.8895	0.0843	165.9	10.7
	Product	AtLeastOneNegative	0.3849	0.0138	0.8973	0.0271	346.5	10.7
	Min	AllNegative	0.3729	0.0214	**0.8975**	0.0418	273.3	10.7
	Min	AtLeastOneNegative	0.3803	0.0144	0.8973	0.0285	316.9	10.7
	Sum	AllNegative	0.1788	0.1022	0.8660	0.1846	17.4	10.7
	Sum	AtLeastOneNegative	**0.3871**	0.0144	0.8972	0.0284	364.3	10.7
		Leverage	0.1471	**0.1330**	0.7710	**0.2336**	8.5	10.7
		Likelihood Ratio	0.3727	0.0156	0.8972	0.0306	280.2	10.7
MCS=100 β=10	Product	AllNegative	0.4846	0.2653	**0.9213**	0.4003	109	11.3
	Product	AtLeastOneNegative	0.4528	0.0273	0.9047	0.0531	303.7	11.3
	Min	AllNegative	0.4608	0.1053	0.9110	0.1787	223.8	11.3
	Min	AtLeastOneNegative	0.4507	0.0271	0.9046	0.0526	290.6	11.3
	Sum	AllNegative	0.5075	**0.3948**	0.9131	**0.5630**	10.3	11.3
	Sum	AtLeastOneNegative	0.4610	0.0324	0.9050	0.0628	325.2	11.3
		Leverage	**0.5587**	0.3423	0.8532	0.5038	6.2	11.3
		Likelihood Ratio	0.4458	0.0287	0.9046	0.0558	250.8	11.3
MCS=100 β=20	Product	AllNegative	0.8164	0.7675	0.9745	0.8675	17.8	11.2
	Product	AtLeastOneNegative	0.7408	0.5866	0.9567	0.7255	89.2	11.2
	Min	AllNegative	0.8032	0.7530	0.9729	0.8583	18.8	11.2
	Min	AtLeastOneNegative	0.7419	0.5987	0.9585	0.7294	78	11.2
	Sum	AllNegative	**0.8744**	**0.8254**	**0.9808**	**0.9037**	10.8	11.2
	Sum	AtLeastOneNegative	0.8087	0.6887	0.9676	0.8106	63.5	11.2
		Leverage	0.8318	0.5755	0.9291	0.7300	6.8	11.2
		Likelihood Ratio	0.7614	0.6550	0.9638	0.7851	52.2	11.2

Table 4.4: Ensemble results on simulated datasets

4.7 Conclusion

In this chapter, we connect modularity-based methods with correlation analysis and use different correlation measures to change the objective function of modularity-based methods. An upper bound analysis is conducted to analyze the bias of different objective functions and the bias is validated by our experiments. With respect to the widely accepted measure, Normalized Mutual Information, to compare the partition determined by the algorithm with the ground truth, Likelihood Ratio is better and more robust. We proposed the way to estimate the graph parameters which can help us to select the right measure. In addition, different measures can be used for different purposes. For example, Probability Ratio can be used if we want to fairly partition the students in the class into small groups for class projects, and we might use Leverage to find relatively large groups for marketing campaigns. If we want to find the real community structure but don't know the parameter setting of the real dataset, we can use the ensemble method with the "AllNegative" stopping criterion to prevent us from getting the worst result. In the future, we will investigate more correlation measures and work towards overlapping partitioning which is more realistic.

CHAPTER 5
CONCLUSION

In this dissertation, we did research on correlation analysis for binary data. First, we studied 18 different correlation measures and provide guidelines for users to select good correlation measures according to their situation. Second, we applied our correlation analysis to two important applications: market basket analysis and network community detection. For market basket analysis, we propose an FCI framework to decouple the correlation measure from the need for efficient search of correlated itemset. In addition, we proposed several algorithms to speed up the pair search or repeated search. For network community detection, we connect modularity-based methods with correlation analysis and use different correlation measures to change the objective function of modularity-based methods. In addition to the above two applications, there are also many promising applications related to our correlation research. In the data mining area, itemset mining and association rules is a big research topic. My research on correlation analysis brings new insights to a lot of research questions by replacing the measure Support with other correlation measures. In the past ten years, much research has extended itemset mining and association rules to handle time series data and graph data like frequent subgraph mining [49] and itemset mining on time series data [55]. We can see a big opportunity to improve the research on itemset mining and association rules to handle time series data and graph data from the correlation analysis perspective. In classification, Liu et al. [53] made use of association rules for classification problems. Jalali-Heravi et al. [47] conducted the research on how correlation measures can improve associative classifiers proposed by Liu et al., and there are many unanswered questions on this topic. An-

other related research question is how to make use of correlation measures to select concise and useful association rules [13]. In social network area, the influential nodes search can either be used for word-of-mouth marketing or infectious disease prevention. The traditional methods assume a person has the same impact on all his or her friends. In reality, a person in a network is usually more influenced by his or her close friends, and is able to influence only a few others. This observation requires us to study which correlation measure can more effectively measure the impact to each other, and how we can adjust the impact if we get information other than the topology of the social network. Another important issue is the role of the person across different communities. A person connected to many different communities plays a very important role in passing the message or disease. This requires us to extend the fundamental research to overlapping community detection. In all, our correlation analysis has provided better solutions for many practical problems, and has many research opportunities for other practical problems.

Property 1. *M is equal to a certain constant number C when all the items in the itemset are statistically independent.*

Proof. **Support:** When $tp = ep = 0.1$, the Support is 0.1. When $tp = ep = 0.2$, the Support is 0.2. There is no such constant number C for Support.

Any-confidence, All-confidence, Bond: For a pair $\{A, B\}$ with $tp = 0.1$, $P(A) = 0.2$, and $P(B) = 0.5$, the Any-confidence is 0.5. For a pair $\{A, B\}$ with $tp = 0.1$, $P(A) = 0.4$, and $P(B) = 0.25$, the Any-confidence is 0.4. There is no such constant number C. The same example applies to All-confidence and Bond.

IS: When $tp = ep$, $IS = \sqrt{tp}$ which is not constant. Therefore, there is no such constant number C for IS.

Other General Correlation Measures: When $tp = ep$, the Simplified χ^2-statistic, Probability ratio, Leverage, Likelihood Ratio, BCPNN, SCWCC, Two-way Support, and SCWS are all equal to 0. Therefore, the constant number C for them is 0.

ϕ-coefficient, Added Value: When A and B are independent from each other, ϕ-coefficient and Added Value are equal to 0. Therefore, the constant number C for them is 0.

Relative Risk, Odds Ratio, Conviction: When A and B are independent from each other, Relative Risk, Odds Ratio, and Conviction are equal to 1. Therefore, the constant number C for them is 1.

Property 2. *M monotonically increases with the increase of $P(S)$ when all the $P(I_i)$ remain the same.*

Proof. **Support:** Since the Support is equal to tp, the Support increases with tp when ep is fixed.

Any-confidence: When all the $P(I_i)$ remain the same, $min(P(I_1), P(I_2), ..., P(I_m))$ remain the same. Then Any-confidence increases with the increase of tp according to the

formula.

All-confidence: When all the $P(I_i)$ remain the same, $max(P(I_1), P(I_2), ..., P(I_m))$ remain the same. Then All-confidence increases with the increase of tp according to the formula.

Bond: When all the $P(I_i)$ remain the same, $P(I_1 \cup I_2 \cup ... \cup I_m)$ could increase or decrease. Then Bond might decrease with the increase of tp when all the $P(I_i)$ remain the same.

Simplified χ^2-statistic, SCWCC, and SCWS: When all the $P(I_i)$ remain the same, ep stays the same. When $tp \geq ep$, $(tp - ep)^2$ increases with the increase of tp. The Simplified χ^2-statistic, $n \cdot (tp - ep)^2/ep$, increases with the increase of tp. When $tp < ep$, $(tp - ep)^2$ decreases with the increase of tp. The Simplified χ^2-statistic, $-n \cdot (tp - ep)^2/ep$, increases with the increase of tp. Similar to the Simplified χ^2-statistic, SCWCC and SCWS increase with the increase of tp when ep stays the same.

Probability Ratio, BCPNN, IS, and Two-way Support: When all the $P(I_i)$ remain the same, ep stays the same. Probability Ratio, $ln(tp/ep)$, increases with the increase of tp. Similar to Probability Ratio, BCPNN, IS and Two-way Support increase with the increase of tp when ep stays the same.

Leverage: When all the $P(I_i)$ remain the same, ep stays the same. Leverage, $tp - ep$, increases with the increase of tp.

Likelihood Ratio: When all the $P(I_i)$ remain the same, ep stays the same.

When $tp > ep$,

$$\begin{aligned} LikelihoodRatio(S) &= n \cdot tp \cdot (ln(tp) - ln(ep)) + n \cdot (1 - tp) \cdot (ln(1 - tp) - ln(1 - ep)) \\ &= n \cdot tp \cdot ln(tp) - n \cdot tp \cdot ln(ep) + n \cdot ln(1 - tp) - n \cdot ln(1 - ep) \\ &\quad -n \cdot tp \cdot ln(1 - tp) + n \cdot tp \cdot ln(1 - ep) \\ &= n \cdot tp \cdot ln\frac{tp}{1 - tp} + n \cdot ln(1 - tp) - n \cdot ln(1 - ep) + n \cdot tp \cdot ln\frac{1 - ep}{ep} \end{aligned}$$

If we consider $LikelihoodRatio(S)$ as a function of tp, then

$$\begin{aligned} LikelihoodRatio(S)' &= n \cdot ln\frac{tp}{1-tp} + n \cdot tp \cdot \frac{1-tp}{tp} \cdot \frac{(1-tp)-(-1)\cdot tp}{(1-tp)^2} \\ &\quad +n \cdot \frac{-1}{1-tp} + n \cdot ln\frac{1-ep}{ep} \\ &= n \cdot ln(\frac{tp}{1-tp} \cdot \frac{1-ep}{ep}) \end{aligned}$$

Since $tp > ep$, $tp/ep > 1$ and $(1-ep)/(1-tp) > 1$, then $LikelihoodRatio(S)' > 0$. In other words, Likelihood Ratio increases with the increase of tp when $tp > ep$.

When $tp < ep$,

$$\begin{aligned} LikelihoodRatio(S) &= -n \cdot tp \cdot (ln(tp) - ln(ep)) - n \cdot (1-tp) \cdot (ln(1-tp) - ln(1-ep)) \\ &= -n \cdot tp \cdot ln(tp) + n \cdot tp \cdot ln(ep) - n \cdot ln(1-tp) + n \cdot ln(1-ep) \\ &\quad +n \cdot tp \cdot ln(1-tp) - n \cdot tp \cdot ln(1-ep) \\ &= n \cdot tp \cdot ln\frac{1-tp}{tp} - n \cdot ln(1-tp) + n \cdot ln(1-ep) + n \cdot tp \cdot ln\frac{ep}{1-ep} \end{aligned}$$

If we consider $LikelihoodRatio(S)$ as a function of tp, then

$$\begin{aligned} LikelihoodRatio(S)' &= n \cdot ln\frac{1-tp}{tp} + n \cdot tp \cdot \frac{tp}{1-tp} \cdot \frac{-tp-(1-tp)}{tp^2} \\ &\quad -n \cdot \frac{-1}{1-tp} + n \cdot ln\frac{ep}{1-ep} \\ &= n \cdot ln(\frac{1-tp}{tp} \cdot \frac{ep}{1-ep}) \end{aligned}$$

Since $tp < ep$, $ep/tp > 1$ and $(1-tp)/(1-ep) > 1$, then $LikelihoodRatio(S)' > 0$. In other words, Likelihood Ratio increases with the increase of tp when $tp < ep$. In all, Likelihood Ratio increases with the increase of tp.

ϕ-coefficient: When f_{1+} and f_{+1} remain the same but f_{11} increases, f_{10} and f_{01} decrease because $f_{1+} = f_{11} + f_{10}$ and $f_{+1} = f_{11} + f_{01}$. Since f_{0+} is the same and f_{01} decreases, then f_{00} increases. According to the formula, the ϕ correlation coefficient increases.

Relative Risk: When f_{1+} and f_{+1} remain the same but f_{11} increases, f_{0+} remains the same and f_{01} decreases. According to the formula, Relative Risk increases.

Odds Ratio: When f_{1+} and f_{+1} remain the same but f_{11} increases, f_{10} and f_{01} decrease

because $f_{1+} = f_{11} + f_{10}$ and $f_{+1} = f_{11} + f_{01}$. Since f_{0+} is the same and f_{01} decreases, then f_{00} increases. According to the formula, Odds Ratio increases.

Conviction: When f_{1+} and f_{+1} remain the same but f_{11} increases, f_{10} decreases because $f_{1+} = f_{11} + f_{10}$. According to the formula, Conviction increases.

Added Value: When f_{1+} and f_{+1} remain the same but f_{11} increases, Added Value increases according to the formula.

Property 3. *M monotonically decreases with the increase of any $P(I_i)$ when the remaining $P(I_k)$ and $P(S)$ remain unchanged.*

Proof. **Support:** Since the Support is equal to tp, the Support stays the same with the increase of ep when tp is fixed.

Any-confidence: For any $P(I_j)$ which is not equal to $min(P(I_1), P(I_2), ..., P(I_m))$, Any-confidence stays the same with the increase of $P(I_j)$ when the remaining $P(I_k)$ and tp remain unchanged.

All-confidence: For any $P(I_j)$ which is not equal to $max(P(I_1), P(I_2), ..., P(I_m))$, All-confidence stays the same with the decrease of $P(I_j)$ when the remaining $P(I_k)$ and tp remain unchanged.

Bond: With the increase of a certain $P(I_j)$, $P(I_1 \cup I_2 \cup ... \cup I_m)$ might be the same. Then Bond might stay the same with the decrease of $P(I_j)$ when the remaining $P(I_k)$ and tp remain unchanged.

Simplified χ^2-statistic, SCWCC, and SCWS: With the increase of a certain $P(I_j)$, ep increases. When $tp \geq ep$, $\chi^2(S) = n \cdot (tp - ep)^2/ep$. If we consider the Simplified χ^2-statistic as a function of ep, then $\chi^{2'}(S) = n \cdot (ep^2 - tp^2)/ep^2$. Since $0 \leq ep \leq tp \leq 1$, $ep^2 \leq tp^2$. Therefore, $\chi^{2'}(S) \leq 0$. Similarly, when $tp < ep$, $\chi^2(S) = -n \cdot (tp - ep)^2/ep$ and $\chi^{2'}(S) = -n \cdot (ep^2 - tp^2)/ep^2 < 0$. In all, the Simplified χ^2-statistic decreases with the increase of ep. Similar to the Simplified χ^2-statistic, SCWCC and SCWS decrease with the increase of ep when tp stays the same.

Probability Ratio, BCPNN, IS, and Two-way Support: With the increase of a certain $P(I_j)$, ep increases. When tp is fixed, Probability Ratio decreases with the increase of ep. Similar to Probability Ratio, BCPNN, IS and Two-way Support decrease with the increase of ep when tp stays the same.

Leverage: With the increase of a certain $P(I_j)$, ep increases. When tp is fixed, Leverage decreases with the increase of ep.

Likelihood Ratio: With the increase of a certain $P(I_j)$, ep increases.

When $tp > ep$,

$$\begin{aligned} LikelihoodRatio(S) &= n \cdot tp \cdot (ln(tp) - ln(ep)) + n \cdot (1 - tp) \cdot (ln(1 - tp) - ln(1 - ep)) \\ &= n \cdot tp \cdot ln(tp) - n \cdot tp \cdot ln(ep) + n \cdot ln(1 - tp) - n \cdot ln(1 - ep) \\ &\quad -n \cdot tp \cdot ln(1 - tp) + n \cdot tp \cdot ln(1 - ep) \\ &= n \cdot tp \cdot ln\frac{tp}{1 - tp} + n \cdot ln(1 - tp) - n \cdot ln(1 - ep) + n \cdot tp \cdot ln\frac{1 - ep}{ep}. \end{aligned}$$

If we consider $LikelihoodRatio(S)$ as a function of ep, then

$$\begin{aligned} LikelihoodRatio(S)' &= \frac{n}{1 - ep} - \frac{n \cdot tp}{(1 - ep) \cdot ep} \\ &= \frac{n \cdot (ep - tp)}{(1 - ep) \cdot ep}. \end{aligned}$$

Since $tp > ep$, then $LikelihoodRatio(S)' < 0$. In other words, Likelihood Ratio decreases with the increase of ep when $tp > ep$. Similarly, when $tp < ep$, we can prove Likelihood Ratio decreases with the increase of ep. In all, Likelihood Ratio decreases with the increase of ep.

ϕ-coefficient: We have

$$\phi = \frac{P(AB) - P(A) * P(B)}{\sqrt{P(A) * P(B) * (1 - P(A)) * (1 - P(B))}}.$$

If $P(AB)$ and $P(B)$ remain the same but $P(A)$ increases, we need to prove ϕ de-

creases. If we consider ϕ as a function of $P(A)$, then

$$\begin{aligned}
\phi' &= \frac{1}{\sqrt{P(B)\cdot(1-P(B))}}\cdot\left(\frac{P(AB)-P(A)\cdot P(B)}{\sqrt{P(A)\cdot(1-P(A))}}\right)' \\
&= \frac{1}{\sqrt{P(B)\cdot(1-P(B))}} \\
&\quad \cdot\frac{-P(B)\cdot\sqrt{P(A)\cdot(1-P(A))}-(P(AB)-P(A)\cdot P(B))\cdot\frac{1-2\cdot P(A)}{2\cdot\sqrt{P(A)\cdot(1-P(A))}}}{P(A)\cdot(1-P(A))} \\
&= \frac{P(AB)\cdot P(A)-P(AB)/2-P(A)\cdot P(B)/2}{(\sqrt{P(A)\cdot(1-P(A))})^3} \\
&= \frac{(P(AB)-P(B))\cdot P(A)/2+(P(A)-1)\cdot P(AB)/2}{(\sqrt{P(A)\cdot(1-P(A))})^3}
\end{aligned}$$

Since $P(AB) < P(B)$ and $P(A) < 1$, $\phi' < 0$. Therefore, ϕ decreases as $P(A)$ increases. Similarly, we can prove ϕ decreases when $P(AB)$ and $P(A)$ remain the same but $P(B)$ increases.

Relative Risk: f_{11} remains the same. When f_{1+} increases and f_{+1} stays the same, f_{11}/f_{1+} decreases and f_{01}/f_{0+} increases. In this case, Relative Risk decreases. When f_{1+} stays the same and f_{+1} increases, f_{11}/f_{1+} is the same and f_{01}/f_{0+} increases. In this case, Relative Risk decreases. In all, Relative Risk decreases when *tp* is the same and *ep* increases.

Odds Ratio: When f_{11} remains the same but f_{1+} or f_{+1} increases, then f_{10} or f_{01} increases and f_{00} decreases. According to the formula, Odds Ratio decreases.

Conviction: f_{11} remains the same. When f_{1+} increases and f_{+1} stays the same, f_{10} increases and f_{00} decreases. $Conviction = (f_{11}+f_{10})\cdot(f_{10}+f_{00})/f_{10} = f_{11}+n-f_{+1}+f_{11}\cdot f_{00}/f_{10}$. Conviction decreases. When f_{1+} stays the same and f_{+1} increases, f_{10} is the same and f_{00} decreases. $Conviction = (f_{11}+f_{10})\cdot(f_{10}+f_{00})/f_{10} = f_{10}+f_{11}+f_{00}+f_{11}\cdot f_{00}/f_{10}$. Conviction decreases. In all, Conviction decreases when *tp* is the same and *ep* increases.

Added Value: Since $AddedValue = P(B|A)-P(B) = P(A\cap B)/P(A)-P(B)$, it decreases when $P(A)$ or $P(B)$ increases if $P(A\cap B)$ is fixed.

Property 4. *The upper bound of M gets closer to the constant C when $P(S)$ is close to 0.*

Proof. **Support, Any-confidence, All-confidence, Bond, IS:** There is no such constant number C. Therefore, it is impossible to be closer to C for them.

Remaining General Correlation Measures: Given tp and $m \geq 2$, we have $tp^m \leq ep \leq ((m-1+tp)/m)^m$. For the remaining general correlation measures which satisfy properties 1 and 3, they reach their upper bound when $ep = tp^m < tp$. According to the formula shown in Table 2.9, we can check their upper bounds when $tp \to 0$.

The Pair-only Correlation Measure: For the pair-only correlation measures, $tp^2 \leq ep \leq ((1+tp)/2)^2$ given tp and $m = 2$. They reach their upper bound when $ep = tp^2$. It means $f_{10} = 0$, $f_{01} = 0$, and $f_{00} = n - f_{11}$. According to the formula shown in Table 2.9, we can check their upper bounds when $tp \to 0$.

Property 5. *M gets closer to C (including negative correlation cases) when an independent item is added to S.*

Proof. **Support, Any-confidence, All-confidence, Bond, IS:** Since there is no such constant number C, it is impossible to be closer to C for them.

Simplified χ^2-statistic: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then, when $tp \neq ep$, we need to prove $|\chi^2(S')| \leq |\chi^2(S)|$. Since $|\chi^2(S)| = n \cdot (tp - ep)^2/ep$ and $|\chi^2(S')| = n \cdot (tp \cdot p - ep \cdot p)^2/(ep \cdot p) = p \cdot n \cdot (tp - ep)^2/ep$, $|\chi^2(S')| = p \cdot |\chi^2(S)| \leq |\chi^2(S)|$.

Probability Ratio: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. $ProbabilityRatio(S') = (tp \cdot p)/(ep \cdot p) = tp/ep = ProbabilityRatio(S)$. Therefore, Property 6 is violated.

Leverage: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then when $tp \neq ep$, we need to prove $|Leverage(S')| \leq |Leverage(S)|$. Since $|Leverage(S)| = |tp - ep|$ and $|Leverage(S')| = |tp \cdot p - ep \cdot p|$, $|Leverage(S')| = p * |Leverage(S)| \leq |Leverage(S)|$.

Likelihood Ratio: Let $S' = S \cup I$, where item I is independent from the set S. The actual

probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then when $tp \neq ep$, we need to prove $|LikelihoodRatio(S')| \leq |LikelihoodRatio(S)|$. Since

$$|LikelihoodRatio(S)| = \frac{\binom{n}{k} tp^k \cdot (1-tp)^{(n-k)}}{\binom{n}{k} ep^k \cdot (1-ep)^{(n-k)}}$$

and

$$|LikelihoodRatio(S')| = \frac{\binom{n}{k \cdot p} (tp \cdot p)^{(k \cdot p)} \cdot (1 - tp \cdot p)^{(n-k \cdot p)}}{\binom{n}{k \cdot p} (ep \cdot p)^{(k \cdot p)} \cdot (1 - ep \cdot p)^{(n-k \cdot p)}},$$

we need to prove: when $tp \neq ep$,

$$\frac{\binom{n}{k \cdot p} (tp \cdot p)^{(k \cdot p)} \cdot (1 - tp \cdot p)^{(n-k \cdot p)}}{\binom{n}{k \cdot p} (ep \cdot p)^{(k \cdot p)} \cdot (1 - ep \cdot p)^{(n-k \cdot p)}} < \frac{\binom{n}{k} tp^k \cdot (1-tp)^{(n-k)}}{\binom{n}{k} ep^k \cdot (1-ep)^{(n-k)}}.$$

If we consider the left-hand side as a function of p,

$$f(p) = \frac{\binom{n}{k \cdot p} (tp \cdot p)^{(k \cdot p)} \cdot (1 - tp \cdot p)^{(n-k \cdot p)}}{\binom{n}{k \cdot p} (ep \cdot p)^{(k \cdot p)} \cdot (1 - ep \cdot p)^{(n-k \cdot p)}},$$

then the right-hand side is the value of $f(p)$ when $p = 1$.

Let

$$\begin{aligned} g(p) &= ln(f(p)) \\ &= k \cdot (ln(tp) - ln(ep)) \cdot p + (n - k \cdot p)(ln(1 - tp \cdot p) \\ &\quad -ln(1 - ep \cdot p)). \end{aligned}$$

According to Taylor's theorem, when $0 < tp \cdot p < 1$,

$$ln(1 - tp \cdot p) = -\sum_{i=1}^{\infty} \frac{(tp \cdot p)^i}{i}.$$

Similarly, when $0 < ep \cdot p < 1$,

$$ln(1 - ep \cdot p) = -\sum_{i=1}^{\infty} \frac{(ep \cdot p)^i}{i}.$$

Therefore,

$$\begin{aligned} g(p) &= k \cdot ln\left(\frac{tp}{ep}\right) \cdot p + (n - k \cdot p) \cdot \sum_{i=1}^{\infty} \frac{(ep \cdot p)^i - (tp \cdot p)^i}{i} \\ &= k \cdot ln\left(\frac{tp}{ep}\right) \cdot p + n \cdot \sum_{i=1}^{\infty} \left(\frac{ep^i - tp^i}{i} \cdot p^i\right) - \\ & \quad k \cdot p \cdot \sum_{i=1}^{\infty} \left(\frac{ep^i - tp^i}{i} \cdot p^i\right) \\ &= \left[k \cdot ln\left(\frac{tp}{ep}\right) - n \cdot (tp - ep)\right] \cdot p + \\ & \quad \sum_{i=2}^{\infty} \left[k \cdot \frac{tp^{(i-1)} - ep^{(i-1)}}{i-1} - n \cdot \frac{tp^i - ep^i}{i}\right] \cdot p^i \end{aligned}$$

Since $k = n \cdot tp$,

$$\begin{aligned} g(p) &= k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp}\right] \cdot p + \\ & \quad \sum_{i=2}^{\infty} \left(n \cdot tp \cdot \frac{tp^{(i-1)} - ep^{(i-1)}}{i-1} - n \cdot \frac{tp^i - ep^i}{i}\right) \cdot p^i \\ &= k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp}\right] \cdot p + \\ & \quad \sum_{i=2}^{\infty} \left(\frac{tp^i - tp \cdot ep^{(i-1)}}{i-1} - \frac{tp^i - ep^i}{i}\right) \cdot n \cdot p^i \\ &= k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp}\right] \cdot p + \\ & \quad \sum_{i=2}^{\infty} \frac{tp^i - ep^i - i \cdot (tp - ep) \cdot ep^{(i-1)}}{i \cdot (i-1)} \cdot n \cdot p^i \end{aligned}$$

Let $f_1(i) = tp^i - ep^i - i \cdot (tp - ep) \cdot ep^{(i-1)}$. Then

$$g(p) = k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp}\right] \cdot p + \sum_{i=2}^{\infty} \frac{f_1(i)}{i \cdot (i-1)} \cdot n \cdot p^i.$$

Since $tp^i - ep^i = (tp - ep) \cdot \sum_{j=0}^{i-1} \left(tp^{(i-1-j)} \cdot ep^j\right)$ and $i \cdot ep^{(i-1)} = \sum_{j=0}^{i-1} ep^{(i-1)}$, then

$$\begin{aligned} f_1(i) &= (tp - ep) \cdot \sum_{j=0}^{i-1} \left(tp^{(i-1-j)} \cdot ep^j\right) - (tp - ep) \cdot \sum_{j=0}^{i-1} ep^{(i-1)} \\ &= (tp - ep) \cdot \left[\sum_{j=0}^{i-1} \left(tp^{(i-1-j)} \cdot ep^j\right) - \sum_{j=0}^{i-1} ep^{(i-1)}\right] \\ &= (tp - ep) \cdot \sum_{j=0}^{i-1} \left(tp^{(i-1-j)} \cdot ep^j - ep^{(i-1)}\right) \\ &= (tp - ep) \cdot \sum_{j=0}^{i-1} \left(tp^{(i-1-j)} - ep^{(i-1-j)}\right) \cdot ep^j. \end{aligned}$$

Therefore,

$$\begin{aligned} g(p) &= k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp} \right] \cdot p + \\ &\sum_{i=2}^{\infty} \frac{\sum_{j=0}^{i-1} \left(tp^{(i-1-j)} - ep^{(i-1-j)} \right) \cdot ep^j}{i \cdot (i-1)} \cdot n \cdot (tp - ep) \cdot p^i \end{aligned}$$

Given the coefficient of p in $g(p)$,

$$k \cdot \left[ln\left(\frac{tp}{ep}\right) - 1 + \frac{ep}{tp} \right]$$

Suppose

$$x = \frac{tp}{ep}, f_2(x) = lnx - 1 + \frac{1}{x}.$$

Then

$$f_2'(x) = \frac{1}{x} - \frac{1}{x^2}.$$

When $tp > ep$, $x > 1$ and $f_2'(x) > 0$. Therefore, $f_2(x) > f_2(1) = 0$ when $x > 1$. When $tp < ep$, $0 < x < 1$ and $f_2'(x) < 0$. Therefore, $f_2(x) > f_2(1) = 0$ when $0 < x < 1$. Thus, the coefficient of p is larger than 0 when $tp \neq ep$.

Given the coefficient of p^i in $g(p)$,

$$\frac{\sum_{j=0}^{i-1} \left(tp^{(i-1-j)} - ep^{(i-1-j)} \right) \cdot ep^j}{i \cdot (i-1)} \cdot n \cdot (tp - ep)$$

When $tp > ep$, $tp - ep > 0$ and $tp^{(i-1-j)} - ep^{(i-1-j)} > 0$. Therefore,

$$\frac{\sum_{j=0}^{i-1} \left(tp^{(i-1-j)} - ep^{(i-1-j)} \right) \cdot ep^j}{i \cdot (i-1)} \cdot n \cdot (tp - ep) > 0.$$

When $tp < ep$, $tp - ep < 0$ and $tp^{(i-1-j)} - ep^{(i-1-j)} < 0$. Therefore,

$$\frac{\sum_{j=0}^{i-1} \left(tp^{(i-1-j)} - ep^{(i-1-j)} \right) \cdot ep^j}{i \cdot (i-1)} \cdot n \cdot (tp - ep) > 0.$$

Thus, the coefficient of $p^i (i = 2, 3, ...)$ is larger than 0 when $tp \neq ep$.

Given the function $f_3(x) = \sum_{i=1}^{\infty} \alpha_i \cdot x^i$ $(0 < x < 1)$, if all the coefficients $\alpha_i > 0$, the function $f_3(x)$ monotonically increases with x. Therefore, the functions $g(p)$ and $f(p)$ monotonically increase with p.

Therefore, $f(p) < f(1)$. We get $|LikelihoodRatio(S')| < |LikelihoodRatio(S)|$, when $tp \neq ep$.

BCPNN: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then when $tp \neq ep$, we need to prove $|BCPNN(S')| \leq |BCPNN(S)|$. When $tp > ep$, $(tp-ep) \cdot cc > (tp-ep) \cdot p \cdot cc$. We get $tp \cdot cc + ep \cdot p \cdot cc > ep \cdot cc + tp \cdot p \cdot cc \Rightarrow tp \cdot ep \cdot p + tp \cdot cc + ep \cdot p \cdot cc + cc^2 > tp \cdot ep \cdot p + ep \cdot cc + tp \cdot p \cdot cc + cc^2 \Rightarrow (tp+cc) \cdot (ep \cdot p + cc) > (ep+cc) \cdot (tp \cdot p + cc) \Rightarrow |BCPNN(S)| > |BCPNN(S')|$. Similarly, when $tp < ep$, we also get $|BCPNN(S)| > |BCPNN(S')$.

SCWCC: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then when $tp \neq ep$, we need to prove $|SCWCC(S')| \leq |SCWCC(S)|$. Since $p^2 \cdot ep + cc \cdot p^2 < p \cdot ep + cc$, we have $\frac{p^2}{ep \cdot p + cc} < \frac{1}{ep+cc} \Rightarrow \frac{(tp \cdot p - ep \cdot p)^2}{ep \cdot p + cc} < \frac{(tp-ep)^2}{ep+cc}$. We get $|SCWCC(S)| > |SCWCC(S')|$

Two-way Support: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then when $tp \neq ep$, we need to prove $|TwowaySupport(S')| \leq |TwowaySupport(S)|$. $|TwowaySupport(S')| = tp \cdot p \cdot |ln\frac{tp \cdot p}{ep \cdot p}| = tp \cdot p \cdot |ln\frac{tp}{ep}| < tp \cdot |ln\frac{tp}{ep}| = |TwowaySupport(S)|$.

SCWS: Let $S' = S \cup I$, where item I is independent from the set S. The actual probability of S' is $tp \cdot p$ and the expected probability of S' is $ep \cdot p$. Then, when $tp \neq ep$, we need to prove $|SCWS(S')| \leq |SCWS(S)|$. Since $|SCWS(S)| = n \cdot tp \cdot (tp - ep)^2/ep$ and $|SCWS(S')| = n \cdot tp \cdot p \cdot (tp \cdot p - ep \cdot p)^2/(ep \cdot p) = p^2 \cdot n \cdot tp \cdot (tp-ep)^2/ep$, $|SCWS(S')| = p^2 \cdot |SCWS(S)| \leq |SCWS(S)|$.

Pair-only Correlation Measures: If we add an independent item, the itemset size is no longer 2. These measure cannot measure 3-itemset correlation. Therefore, they don't satisfy this property.

Property 6. *The lower bound of M gets closer to the lowest possible function value when $P(S)$ is closer to 0.*

Proof. **Support:** The lower bound of Support is tp, the lower bound gets closer to the lowest possible function value 0 when tp is close to 0.

Any-confidence: Since $tp \leq P(I_i) \leq 1$, $tp \leq min(P(I_1), P(I_2), ..., P(I_m)) \leq ((m-1+tp)/m)^m$. The lower bound of Any-confidence is $tp/(1-(1-tp)/m)^m$. It gets closer to the lowest possible function value 0 when tp is close to 0.

All-confidence: Since $tp \leq P(I_i) \leq 1$, $max(P(I_1), P(I_2), ..., P(I_m)) \leq 1$. The lower bound of All-confidence is tp. It gets closer to the lowest possible function value 0 when tp is close to 0.

Bond: No matter how close tp is to 0, $P(I_1 \cup I_2 \cup ... \cup I_m)$ always has the chance to be 0. Then the lower bound of Bond is tp. It gets closer to the lowest possible function value 0 when tp is close to 0.

Remaining General Correlation Measures: Given tp and $m \geq 2$, $tp^m \leq ep \leq ((m-1+tp)/m)^m$. For the remaining general type correlation measures which satisfy properties 1 and 3, they reach their lower bounds when $ep = ((m-1+tp)/m)^m > tp$. According to the formula shown in Table 2.9, we can check their lower bounds when $tp \to 0$.

The Pair-only Correlation Measures: For the pair-only correlation measures, $tp^2 \leq ep \leq ((1+tp)/2)^2$ given tp and $m = 2$. They reach their lower bounds when $ep = ((1+tp)/2)^2 > tp$. It means $f_{10} = (1-tp)/2$, $f_{01} = (1-tp)/2$, and $f_{00} = 0$. According to the formula shown in Table 2.9, we can check their lower bounds when $tp \to 0$.

Property 7. *M gets further away from C (including negative correlation cases) with increased sample size when all the $P(I_i)$ and $P(S)$ remain unchanged.*

Proof. Since any correlation measure is a function of the sample size isolated from other parameters, we use the original version of each measure to test against this property. If the original version contains the parameter of the sample size, it satisfies this property. Otherwise, it doesn't.

Additional Property 1. *M remains the same when exchanging the frequency count f_{11} with f_{00} and f_{10} with f_{01}.*

Proof. **Leverage:** For Leverage, $Leverage_{before} = f_{11} - (f_{11} + f_{10})/n \cdot (f_{11} + f_{01})/n$ before the exchanging operation. After the operation, $Leverage_{after} = f_{00} - (f_{00} + f_{01})/n \cdot (f_{00} + f_{10})/n$. Since $n = f_{11} + f_{10} + f_{01} + f_{00}$, $Leverage_{before} = [f_{11} \cdot (f_{11} + f_{10} + f_{01} + f_{00}) - (f_{11} + f_{10}) \cdot (f_{11} + f_{01})]/n = (f_{11} \cdot f_{00} - f_{10} \cdot f_{01})/n$ and $Leverage_{after} = (f_{00} \cdot f_{11} - f_{01} \cdot f_{10})/n$. We get $Leverage_{after} = Leverage_{before}$.

ϕ-coefficient: For ϕ-coefficient, $\phi_{before} = (f_{11} \cdot f_{00} - f_{10} \cdot f_{01})/\sqrt{f_{1+} \cdot f_{0+} \cdot f_{+1} \cdot f_{+0}}$. After the operation, $\phi_{after} = (f_{00} \cdot f_{11} - f_{01} \cdot f_{10})/\sqrt{f_{0+} \cdot f_{1+} \cdot f_{+0} \cdot f_{+1}}$. We get $\phi_{after} = \phi_{before}$.

Odds Ratio: For Odds Ratio, $OddsRatio_{before} = f_{11} \cdot f_{00}/(f_{10} \cdot f_{01})$. After the operation, $OddsRatio_{after} = f_{00} \cdot f_{11}/(f_{01} \cdot f_{10})$. We get $OddsRatio_{after} = OddsRatio_{before}$.

Others: Given the example of $f_{11} = 1$, $f_{10} = 2$, $f_{01} = 3$, and $f_{00} = 4$, this exchanging operation changes the value of Support, Any-confidence, All-confidence, Bond, Simplified χ^2-statistic, Probability Ratio, Likelihood Ratio, BCPNN, SCWCC, IS, Two-way Support, SCWS, Relative Risk, Conviction, and Added Value.

Additional Property 2. *M remains the same by only increasing f_{00}.*

Proof. **Any-confidence, All-confidence, Bond, IS:** $AnyConfidence = f_{11}/min(f_{11} + f_{10}, f_{11} + f_{01})$, $AllConfidence = f_{11}/max(f_{11} + f_{10}, f_{11} + f_{01})$, $Bond = f_{11}/(f_{11} + f_{10} + f_{01})$, and $IS = f_{11}/\sqrt{(f_{11} + f_{10}) \cdot (f_{11} + f_{01})}$. No matter how f_{00} is changed, their value stays the same.

Others: Suppose the original two-by-two table is as follow: $f_{11} = 1$, $f_{10} = 2$, $f_{01} = 3$, and $f_{00} = 4$. Then we increase f_{00} to 94 and keep others the same. This increasing operation changes the value of Support, Simplified χ^2-statistic, Probability Ratio, Leverage, Likelihood Ratio, BCPNN, SCWCC, Two-way Support, SCWS, ϕ-coefficient, Relative Risk, Odds Ratio, Conviction, and Added Value.

Additional Property 3. *M remains the same under the row/column scaling operation from Table T to T', where T is a contingency table with frequency counts [f_{11}; f_{10}; f_{01}; f_{00}], T' is a contingency table with scaled frequency counts [$k_1k_3f_{11}$; $k_2k_3f_{10}$; $k_1k_4f_{01}$; $k_2k_4f_{00}$], and k_1, k_2, k_3, k_4 are positive constants.*

Proof. **Odds Ratio:** Originally, $OddsRatio_{before} = f_{11} \cdot f_{00}/(f_{10} \cdot f_{01})$. After the operation, $OddsRatio_{after} = k_1 \cdot k_3 \cdot f_{11} \cdot k_2 \cdot k_4 \cdot f_{00}/(k_2 \cdot k_3 \cdot f_{10} \cdot k_1 \cdot k_4 f_{01}) = f_{11} \cdot f_{00}/(f_{10} \cdot f_{01})$. We get $OddsRatio_{after} = OddsRatio_{before}$

Others: Suppose the original two-by-two table is as follow: $f_{11} = 1$, $f_{10} = 2$, $f_{01} = 3$, and $f_{00} = 4$. Then we specify $k_1 = 1$, $k_2 = 2$, $k_3 = 3$, and $k_4 = 4$ to do the row/column scaling operation. This operation changes the value of Support, Any-confidence, All-confidence, Bond, Simplified χ^2-statistic, Probability Ratio, Leverage, Likelihood Ratio, BCPNN, SCWCC, IS, Two-way Support, SCWS, ϕ-coefficient, Relative Risk, Conviction, and Added Value.

REFERENCES

[1] Charu C. Aggarwal, Jian Pei, and Bo Zhang. On privacy preservation against adversarial data mining. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 510–516, New York, NY, USA, 2006. ACM.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *SIGMOD '93: Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 207–216, New York, NY, USA, 1993. ACM.

[3] Jac M. Anthonisse. The rush in a directed graph. Technical report, Stichting Mathemastisch Centrum, Amsterdam, The Netherlands, 1971.

[4] A. Arenas, J. Duch, A. Fernandez, and S. Gomez. Size reduction of complex networks preserving modularity. *NEW JOURNAL OF PHYSICS*, 9:176, 2007.

[5] J. P. Bagrow. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(05):P05001+, 2008.

[6] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science (New York, N.Y.)*, 286(5439):509–512, October 1999.

[7] A. Bate, M. Lindquist, I. R. Edwards, S. Olsson, R. Orre, A. Lansner, and R. M. De Freitas. A bayesian neural network method for adverse drug reaction signal generation. *European Journal of Clinical Pharmacology*, 54(4):315–321, 1998.

[8] Roberto J. Bayardo. Efficiently mining long patterns from databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, New York, NY, USA, 1998. ACM.

[9] S. Boettcher and A. G. Percus. Extremal optimization for graph partitioning. *PHYS.REV.E*, 64:026114, 2001.

[10] Bela Bollobas. *Modern Graph Theory*. Springer, 1998.

[11] Christian Borgelt. Efficient implementations of Apriori and Eclat. In *ICDM '03: Proc. 3rd IEEE Int. Conf. on Data Mining, Workshop on Frequent Itemset Mining Implementations*, 2003.

[12] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE Trans. on Knowl. and Data Eng.*, 20:172–188, February 2008.

[13] Yannick Le Bras, Philippe Lenca, and Stéphane Lallich. Optimonotone measures for optimal rule discovery. *Computational Intelligence*, 2012.

[14] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *SIGMOD '97: Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 265–276, New York, NY, USA, 1997. ACM.

[15] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD '97: Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 255–264, New York, NY, USA, 1997. ACM.

[16] Doug Burdick. Mafia: A maximal frequent itemset algorithm for transactional databases. In *ICDE '01: Proceedings of the 17th International Conference on Data Engineering*, page 443, Washington, DC, USA, 2001. IEEE Computer Society.

[17] Jingchun Chen and Bo Yuan. Detecting functional modules in the yeast protein–protein interaction network. *Bioinformatics*, 22:2283–2290, September 2006.

[18] Jiyang Chen, Osmar R. Zaane, and Randy Goebel. Detecting communities in social networks using max-min modularity. In *SDM'09*, pages 978–989, 2009.

[19] Hong Cheng, Xifeng Yan, Jiawei Han, and Chih-Wei Hsu. Discriminative frequent pattern analysis for effective classification. In *ICDE'07*, pages 716–725, 2007.

[20] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111+, December 2004.

[21] Anne Condon and Richard M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Struct. Algorithms*, 18:116–140, March 2001.

[22] Leon Danon, Albert D. Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(9):P09008–09008, September 2005.

[23] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Dev.*, 17:420–425, September 1973.

[24] Yon Dourisboure, Filippo Geraci, and Marco Pellegrini. Extraction and classification of dense implicit communities in the web graph. *ACM Transactions on the Web*, 3:1–36, 2009.

[25] Lian Duan and W. Nick Street. Finding maximal fully-correlated itemsets in large databases. In *ICDM '09: Proc. Int. Conf. on Data Mining*, pages 770–775, Miami, FL, USA, 2009.

[26] Lian Duan and W. Nick Street. Selecting the right correlation measure for binary data. *Under review*, 2012.

[27] Lian Duan and W. Nick Street. Speeding up correlation search for binary data. *Under review*, 2012.

[28] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Physical Review E*, 72(2):027104+, August 2005.

[29] William Dumouchel. Bayesian data mining in large frequency tables, with an application to the fda spontaneous reporting system. *The American Statistician*, 53(3):177–202, 1999.

[30] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[31] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Graph Theory 1736-1936*, 1736.

[32] H. Everett. 'Relative State' formulation of quantum mechanics. *Reviews of Modern Physics*, 29:454–462, 1957.

[33] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

[34] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[35] Linton C. Freeman. *The Development of Social Network Analysis: A Study in the Sociology of Science*. Empirical Press, 2004.

[36] Katsuki Fujisawa, Yukinobu Hamuro, Naoki Katoh, Takeshi Tokuyama, and Katsutoshi Yada. Approximation of optimal two-dimensional association rules for categorical attributes using semidefinite programming. In *DS '99: Proc. 2nd Int. Conf. on Discovery Science*, pages 148–159, London, UK, 1999. Springer-Verlag.

[37] Liqiang Geng and Howard J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys*, 38(3):9, 2006.

[38] Rumi Ghosh and Kristina Lerman. Community detection using a measure of global influence. In *Proceedings of the Second international conference on Advances in social network mining and analysis*, SNAKDD'08, pages 20–35, Berlin, Heidelberg, 2010. Springer-Verlag.

[39] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.

[40] Karam Gouda and Mohammed J. Zaki. Genmax: An efficient algorithm for mining maximal frequent itemsets. *Data Min. Knowl. Discov.*, 11(3):223–242, 2005.

[41] Roger Guimera and Luis A. Nunes Amaral. Functional cartography of complex metabolic networks. *NATURE*, 433:895, 2005.

[42] Roger Guimerà, Marta S. Pardo, and Lu'is A. Nunes Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101+, August 2004.

[43] Tias Guns, Siegfried Nijssen, and Luc De Raedt. Itemset mining: A constraint programming perspective. *Artif. Intell.*, 175:1951–1983, August 2011.

[44] John H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, USA, 1992.

[45] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

[46] Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

[47] Mojdeh Jalali-Heravi and Osmar R. Zaïane. A study on interestingness measures for associative classifiers. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1039–1046, New York, NY, USA, 2010. ACM.

[48] Christopher Jermaine. Finding the most interesting correlations in a database: How hard can it be? *Information Systems*, 30(1):21–46, 2005.

[49] Chuntao Jiang, Frans Coenen, and Michele Zito. A Survey of Frequent Subgraph Mining Algorithms. *Knowledge Engineering Review*, 2012.

[50] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, 78:046110, Oct 2008.

[51] E. A. Leicht and M. E. J. Newman. Community Structure in Directed Networks. *Physical Review Letters*, 100(11):118703+, March 2008.

[52] Philippe Lenca, Patrick Meyer, Benot Vaillant, and Stphane Lallich. On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid. *European Journal of Operational Research*, 184(2):610 – 626, 2008.

[53] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating Classification and Association Rule Mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[54] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, December 2007.

[55] Nizar R. Mabroukeh and C. I. Ezeife. A taxonomy of sequential pattern mining algorithms. *ACM Comput. Surv.*, 43(1):3:1–3:41, December 2010.

[56] David J. C. Mackay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 1st edition, June 2003.

[57] S. Mancoridis, B. S. Mitchell, and C. Rorres. Using automatic clustering to produce high-level system organizations of source code. In *In Proc. 6th Intl. Workshop on Program Comprehension*, pages 45–53, 1998.

[58] Claire P. Massen and Jonathan P. K. Doye. Identifying communities within energy landscapes. 71(046101), 2005.

[59] Frederick Mosteller. Association and estimation in contingency tables. *Journal of the American Statistical Association*, 63(321):1–28, 1968.

[60] M. E. J. Newman. Fast algorithm for detecting community structure in networks. September 2003.

[61] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45:167–256, 2003.

[62] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113+, February 2004.

[63] A. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001.

[64] Pang ning Tan and Vipin Kumar. Interestingness measures for association patterns: A perspective. In *KDD 2000 Workshop on Postprocessing in Machine Learning and Data Mining*, 2000.

[65] G. Niklas Norén, Andrew Bate, Johan Hopstadius, Kristina Star, and I. Ralph Edwards. Temporal pattern discovery for trends and transient effects: its application to patient records. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 963–971, New York, NY, USA, 2008. ACM.

[66] Edward R. Omiecinski. Alternative interest measures for mining associations in databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(1):57–69, 2003.

[67] OMOP. Methods section for the disproportionality paper, September 2010. `http://omop.fnih.org/MethodsLibrary`.

[68] Gregory Piatetsky-Shapiro. *Discovery, Analysis, and Presentation of Strong Rules*. AAAI/MIT Press, 1991.

[69] Clara Pizzuti. Community detection in social networks with genetic algorithms. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, GECCO '08, pages 1137–1138, New York, NY, USA, 2008. ACM.

[70] William M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[71] Matthew J. Rattigan, Marc Maier, and David Jensen. Using structure indices for efficient approximation of network properties. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 357–366, New York, NY, USA, 2006. ACM.

[72] Matthew J. Rattigan, Marc Maier, David Jensen Bin Wu, Xin Pei, JianBin Tan, and Yi Wang. Exploiting network structure for active inference in collective classification. In *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, ICDMW '07, pages 429–434, Washington, DC, USA, 2007. IEEE Computer Society.

[73] Henry T. Reynold. *The Analysis of Cross-Classifications*. Free Press, 1977.

[74] C.J. Van Rijsbergen. Foundation of Evaluation. *Journal of Documentation*, 30:365–373, 1974.

[75] John P Scott. *Social Network Analysis: A Handbook*. Sage Publications Ltd., 2000.

[76] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:888–905, August 2000.

[77] Christopher L. Sistrom and Cynthia W. Garvan. Proportions, odds, and risk. *Radiology*, 230(1):12–19, 2004.

[78] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques, 2000.

[79] Einoshin Suzuki. Pitfalls for categorizations of objective interestingness measures for rule discovery. 127:383–395, 2008.

[80] I. Szczech, S. Greco, and R. Slowinski. New property for rule interestingness measures. In *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pages 103 –108, sept. 2011.

[81] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right objective measure for association analysis. *Information Systems*, 29(4):293–313, 2004.

[82] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison Wesley, 2005.

[83] Nikolaj Tatti. Maximum entropy based significance of itemsets. *Knowl. Inf. Syst.*, 17:57–77, October 2008.

[84] Joshua R. Tyler, Dennis M. Wilkinson, and Bernardo A. Huberman. *Email as spectroscopy: automated discovery of community structure within organizations*, pages 81–96. Kluwer, B.V., Deventer, The Netherlands, The Netherlands, 2003.

[85] Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 1275–1276, New York, NY, USA, 2007. ACM.

[86] D.J. Watts. *Small Worlds : The Dynamics of Networks between Order and Randomness*. Princeton University Press, 2003.

[87] Geoffrey I. Webb. Discovering significant patterns. *Mach. Learn.*, 68:1–33, July 2007.

[88] Robert S. Weiss and Eugene Jacobson. A method for the analysis of the structure of complex organizations. *American Sociological Review*, 20(6):pp. 661–668, 1955.

[89] Hui Xiong, Mark Brodie, and Sheng Ma. TOP-COP: Mining top-k strongly correlated pairs in large databases. In *ICDM '06: Proc. sInt. Conf. on Data Mining*, pages 1162–1166, Washington, DC, USA, 2006.

[90] Hui Xiong, Shashi Shekhar, Pang-Ning Tan, and Vipin Kumar. TAPER: A two-step approach for all-strong-pairs correlation query in large databases. *IEEE Trans. on Knowl. and Data Eng.*, 18(4):493–508, 2006.

[91] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger. Scan: a structural clustering algorithm for networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 824–833, New York, NY, USA, 2007. ACM.

[92] S. Ben Yahia, T. Hamrouni, and E. Mephu Nguifo. Frequent closed itemset based algorithms: a thorough structural and analytical survey. *SIGKDD Explor. Newsl.*, 8:93–104, June 2006.

[93] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.

[94] Jian Zhang and Joan Feigenbaum. Finding highly correlated pairs efficiently with powerful pruning. In *CIKM '06: Proc. ACM CIKM Int. Conf. on Information and Knowledge Management*, pages 152–161, New York, NY, USA, 2006. ACM.

[95] Lei Zhang, Qi-ming Zhang, Yi-guo Wang, and Dong-lin Yu. Selecting an appropriate interestingness measure to evaluate the correlation between syndrome elements and symptoms. In *Proceedings of the 15th international conference on New Frontiers in Applied Data Mining*, PAKDD'11, pages 372–383, Berlin, Heidelberg, 2012. Springer-Verlag.

[96] Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis, 2001.

[97] Ning Zhong, Chunnian Liu, and Setsuo Ohsuga. Dynamically organizing kdd processes. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(3):451–473, 2001.

[98] Ning Zhong, Y. Y. Yao, and Setsuo Ohsuga. Peculiarity oriented multi-database mining. In *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 136–146, London, UK, 1999. Springer-Verlag.