ANOMALY DETECTION AND PREDICTIVE ANALYTICS FOR FINANCIAL RISK MANAGEMENT

by

ZHONGMOU LI

A dissertation submitted to the

Graduate School-Newark

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Management

written under the direction of

Dr. Hui Xiong

and approved by

_____

_____

_____

_____

Newark, New Jersey

January 2016

ABSTRACT OF THE DISSERTATION

Anomaly Detection and Predictive Analytics for Financial Risk Management

By ZHONGMOU LI

Dissertation Director: Dr. Hui Xiong

In the big data era, the digital revolution has driven the entire financial industry to collect, store and analyze massive volumes of data nowadays than it ever has in history. With the overwhelming scale of data, new technologies are needed to derive competitive advantage and unlock the power of the data, including the approaches people use for financial risk management.

In this dissertation, we study how advanced data mining techniques can play essential roles in financial risk management. Specifically, we provide case studies to apply data mining techniques in three application scenarios for financial risk management. The first study exploits a special type of fraudulent trading ring pattern in the financial market, and defines the so-called blackhole and volcano patterns to identify the fraud. A blackhole mining framework consisting of two pruning schemes is developed. The first pruning scheme is to exploit the concept of combination dominance to reduce the exponential growth search space. The second pruning scheme is an approximate approach, which can strike a balance between the efficiency and the completeness of blackhole mining.

The second study exploits the problem of contract risk management. In particular, how IT service providers can leverage the experiences and lessons learnt from historical contracts to prevent similar issues from reoccurring in the future, in order to mitigate

the project risks, ensure smooth delivery and continuous profitability. Along this line, we investigate how to predict potential risks for new contracts based on their similarities with existing ones, and develop a new approach as an extension of the Mahalanobis distance metric learning framework to solve the problem.

The third study examines the application of cluster analysis in bankruptcy pattern learning and financial statement fraud detection. By leveraging the domain knowledge in accounting area, valuable features from financial statement can be extracted. Clustering technique is then applied to identify the clustering effect of bankrupt companies in different business sectors. Finally, the most indicative financial features for the bankrupt companies in the business sector can be uncovered from the hidden data and validated by significant tests.

ACKNOWLEDGEMENTS

This work represents the culmination of several years of work, and could not have been accomplished without the love and support from many people.

First, I would like to express my deep gratitude to my mentor and advisor, Prof. Hui Xiong, for his continuous support, guidance and encouragement, which are necessary to survive and thrive the graduate school and the beyond. I thank him for generously giving me motivation, support, time, assistance, opportunities and friendship; for ceaselessly teaching me how to practice the art and science in data mining; and for how to identify key problems with impact, present and evaluate the ideas. He helped making me a better writer, speaker and scholar.

I also sincerely thank my other committee members: Prof. Mengchu Zhou, Prof. Xiaodong Lin, Prof. Spiros Papadimitriou and Prof. Panagiotis Karras. All of them not only provide constructive suggestions and comments on my work and this thesis, but also offer numerous support and help in my career choice, and I am very grateful for them.

Special thanks are due to Dr. Daxin Jiang at Microsoft Research Asia, Dr. Ramendra Sahoo at KPMG, Prof. Wenjun Zhou at University of Tennessee - Knoxville, Prof. Yong Ge at University of North Carolina - Charlotte, and Dr. Hengshu Zhu at Baidu Research, for helping with my job search and career development. Thanks

are also due to Dr. Yanchi Liu, Prof. Qiu Liu, Dr. Shu Tao, Dr. Hang Li, and Prof. Junjie Wu. It was a great pleasure working with all of them. I also owe a hefty amount of thanks to my colleagues and friends Prof. Keli Xiao, Prof. Chuanren Liu, Dr. Chunyu Luo, Prof. Bo Jin, Bin Liu, Zijun Yao, Yanjie Fu, Meng Qu, Constantine Vitt, Jingyuan Yang, Can Chen, Hao Zhong, Farid Razzak, Qingxin Meng, Junming Liu, Dr. Xue Bai, Dr. Liyang Tang, Dr. Chang Tan, Prof. Xiaolin Li, Tong Xu, Chu Guan, Hongting Niu, Guannan Liu, Xinjiang Lu, Leilei Sun, Huang Xu, for their help, friendship and valuable suggestion.

I would like to acknowledge the Department of Management Science and Information Systems (MSIS) and Center for Information Management, Integration and Connectivity (CIMIC) for supplying me with the best imaginable equipment and facilities that helped me to accomplish much of this work.

Finally, my deepest thanks and love to my wife Yan Wei, who made this work possible by providing love and support. Thank my parents, Wei Li and Huijuan Huo, for instilling persistence, determination, and habit of learning in me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Recent years have witnessed a paradigm shift in the financial industry as awareness of the importance of data is becoming widespread. In the big data era, the digital revolution has driven the entire financial industry to collect, store and analyze massive volumes of data nowadays than it ever has in history. With the overwhelming scale of data, new technologies are needed to derive competitive advantage and unlock the power of the data, including the approaches people use for financial risk management.

Risk management in general is the process of identification, analysis and mitigation of uncertainty for business decision making (Saunders, Cornett, & McGraw, 2006), (Chapman & Ward, 1996), (McNeil, Frey, & Embrechts, 2015). It contains various domains and application scenarios, e.g., credit risk - the risk of default on a debt; market risk - the possibility of losses due to the movements in market prices; frauds deliberately committed by human beings - trading fraud, financial statement fraud, and insurance fraud; and project management risk - the potential failure factors in project lifecycles, spread from the design, development, production, and sustainment of a project. In this dissertation, the main focus is on financial risk management, which means the violation of good behaviors in regards to financial losses.

Data mining is the process of automatically discovering useful information in large data repositories (Tan, Steinbach, & Kumar, 2005). Data mining techniques are de-

ployed to scour large databases in order to find novel and useful patterns that might otherwise remain unknown. They also provide capabilities to predict the outcomes of future observations. Data mining techniques have a wide range of applications, including but not limited to finance, telecommunications, retail, healthcare, science and engineering (Koh, Tan, et al., 2011), (Kovalerchuk & Vityaev, 2000), (Kohavi, Mason, Parekh, & Zheng, 2004), (Han, Kamber, & Pei, 2011), (Sasisekharan, Seshadri, & Weiss, 1996).

In this dissertation, we explore the problem of how advanced data mining techniques can play essential roles in financial risk management in today's data-intensive business world. The focus is to develop effective and efficient data analysis techniques to detect financial anomalies and mitigate potential risks. The key challenge is how to address the unique characteristics of different data repositories and develop suitable techniques to meet the specific needs of a particular business application. Specifically, we aim to provide case studies to apply advanced data mining techniques in the following three applications, trading fraud detection, contract risk management, and financial statement fraud detection.

1.1 Trading Fraud Detection

With the development of information technology, it is easy to collect and store massive amount of data in financial service industry. One example is the High-Frequency Trading (HFT) in the financial market (Chordia, Goyal, Lehmann, & Saar, 2013), (Chlistalla, Speyer, Kaiser, & Mayer, 2011), (Gomber, Arndt, Lutat, & Uhle, 2011). In the recent 15 years, people can use computer algorithms to trade securities on a

rapid basis. Since the transaction time has been reduced to microseconds, there are tens of thousands of trades per day for each account on average. Nowadays, HFT accounts for 90% of trades in the U.S. financial market. The big data accumulated has greatly changed the paradigm of the financial market, thus it also requires the change of approaches people use to detect financial trading frauds.

One example of trading frauds was committed by Jerome Kerviel in 2008 (Clark, October 5, 2010), who was a junior level trader at Bank Societe Generale, with a loss at €4.9 billion. How could a junior level trader cause such a massive loss? Investigation afterwards shows that Kerviel began creating fictitious trades starting late 2006, and he had taken a €50 billion directional position of European stock index futures at the beginning of 2008. When the bank closed out his positions after the market experienced a large drop in equity indices, the losses attributed were estimated at €4.9 billion ($7 billion). So how did a junior level trader managed to bypass the internal control system to perform trading that were far beyond his trading authority limit? It is showed that Kerviel created hundreds of thousands of faked hedge trades, in order to hide his huge directional position. A pattern of his movement was uncovered later, which consist of closing out trades within the three day cycle to avoid trigger notice from the bank's internal control system, and shifting the older positions to newly initiated trades. This real-world example demonstrates trading fraud should be identified at its early stage, otherwise it can result in huge financial losses.

In this dissertation, we study a specific type of trading fraud in the financial market, namely the trading ring pattern. Nowadays, government agencies, such as

U.S. Securities and Exchange Commission (SEC), are facing increasing challenges for trading fraud detection. Due to the tremendous number of trading accounts and volume of transactions in the financial market, it is very hard for the system to correlate trading behaviors across multiple accounts. This weakness opens the door for cross-account collaborative fraud, which is difficult to discover, track and resolve because the activities of the fraudsters usually appear to be normal. For instance, consider a group of traders, who want to manipulate the price of a targeted stock for a specific time period. In general, the manipulation consists of two stages. In the first stage, these traders purchase a large volume of shares from public. During this time period, their net volume of shares increase significantly. After the stock price goes up to a certain degree, these traders start to sell off their shares to the public to earn profit. In the second stage, a similar pattern with the opposite direction can be observed. This kind of illegal trading activities is widely known as the trading ring pattern.

The detection of trading ring pattern is of great value to regulators, since it can help to reduce their workload by preliminarily filtering out a huge number of normal patterns automatically, thus let them be more focused on the suspicious patterns that may commit frauds. In addition, the detected trading ring patterns can provide a view of the interactions among some accounts in the trading network, which can help to detect the collaborative fraud activities for suspicious accounts with trading volumes under the threshold of the system.

1.2 Contract Risk Management

Major IT service providers typically manage a large portfolio of contracts with a variety of customers. Due to the complexity of IT systems and the variety of customer environment, service contracts that need to be fulfilled by the provider can vary significantly. Despite such variances, the experiences of service providers have shown that successful (or unsuccessful) contracts do share common characteristics, and the root causes of many issues experienced in contract execution can be traced back to the risks identifiable prior to contract signing. Therefore, to ensure smooth delivery and continuous profitability, it is critical for service providers to leverage the experiences and lessons learnt from the historical contracts to prevent similar issues from reoccurring in the future.

Along this line, in this dissertation, we study the problem of identifying similar contracts and predicting risks for new contracts to improve service provider's ability of risk management. The typical risk management process used by a service provider is as follows. Once received the request for proposal from a potential customer, service provider will conduct pre-bid consulting to draft a solution proposal. Then, as the solution being proposed to the customer, risk managers of the provider will assess both the technical and business aspects of the new contract, during which risks may be identified and the corresponding risk mitigation steps conducted (through e.g., modification of solution, negotiation on service level agreements and price). This is the phase during which risk managers look for similar risks that the provider has experienced in historical contracts. The risk identification and mitigation steps will

be iterated as the contract negotiation continues, until the final contract is agreed upon and signed. After contract signing, the provider will conduct periodic project management reviews, to identify and address issues in contract delivery. Traditionally, the risk assessment method employed is largely qualitative, making it hard to leverage the lessons learnt from historical contracts, and come up with adequate risk mitigation recommendations. We argue that a more quantitative approach to predict the outcome of a contract, as well as its potential risks, is of great value to the provider.

In literature, while there are existing works related to using predictive models for contract risk management, many of them focus on different aspects in the risk management lifecycle. For example, Mojsilovic et al. (Mojsilović, Ray, Lawrence, & Takriti, 2007) use predictive models to estimate the likelihood of revenue erosion in large outsourcing engagements, while Goo et al. (Goo, Kishore, Nam, Rao, & Song, 2007) study factors that influence the duration of IT outsourcing relationships. In addition, there are related studies on risk management in large and complex projects (Leung, Rao Tummala, & Chuah, 1998), (Deleris, Katircioglu, Kapoor, Lam, & Bagchi, 2007). Different from these studies, the focus in this dissertation is on the unique problem of identifying similarity between historical and new contracts, and using this information to predict risks for the latter.

1.3 Financial Statement Fraud Detection

Financial fraud detection is one of the most important aspects in financial risk management. In recent years, there is an increasing trend in financial losses due to finan-

cial fraud and bankruptcy. However, traditional auditing process and risk analysis could not match the increased demand of automatic and efficient detection of financial statement fraud. Hence, how to develop an efficient and effective financial fraud detection framework draws the best interests among investors, public, researchers, auditors and regulators. In general, the published financial statements are one of the most pervasive and consistently available predictor of a companys future performance, since they provide the basis for understanding and evaluating the financial status of a company. However, fraudulent financial reporting can destroy the true picture of a companys financial situation. For example, by manipulating elements such as liabilities, expenses, or losses in the financial statement, the unhealthy financial status can be covered. Nowadays, financial statement fraud is one of the most notable management frauds in the U.S. due to the large financial losses it yields. Therefore, financial statement fraud detection becomes a crucial and pervasive issue in financial risk management.

In literature, there have been numerous researches conducted in the area of financial statement fraud detection. In accounting area, hundreds of financial indicators which extract significant features from financial statement have been proposed to build the predication model of high fraud-potential and high bankrupt-potential firms (Altman, 1968), (Loebbecke, Eining, & Willingham, 1989). These indicators are proposed based on empirical experience. On the other hand, in data mining area, several groups of researchers have devoted a significant amount of effort in using modern methods to study financial statement fraud detection from different perspectives. Most of these works employed supervised learning techniques to solve

classification tasks (Kirkos, Spathis, & Manolopoulos, 2007), (Apparao et al., 2009), including neural network (Green & Choi, 1997), (Fanning & Cogger, 1998), (Lin, Hwang, & Becker, 2003), (Thiprungsri & Vasarhelyi, 2011), and regression methods (Bell & Carcello, 2000), (Abbott, Parker, & Peters, 2002), (Spathis, 2002). While these studies support the significance of classification techniques to analysis financial statement, there has been little research in literature in the direction of applying unsupervised learning techniques to analyze the financial statement data. Moreover, majority of the existing researches only focus on a particular business sector. For instance, bankruptcy predictions have been conducted in the railroad, banking, brokerage, education, and insurance industries (Zmijewski, 1984).

In this dissertation, we would like to find the answer to an interesting question: can we use financial statement information to identify which business sector has the strongest clustering effect, such that the financial features we extracted from the past bankrupt (financial unhealthy) companies in the business sector has the most significant predictive power in the future? The question is not only useful for auditors, but also important for investors. Investors can prevent potential losses by filtering out firms with high potential bankruptcy risks based on the learnt characteristics. Auditors can use the valuable features as an additional decision aid to monitor the financial situation of the client firm.

1.4 Overview

The rest of the dissertation are organized as follows.

Chapter 2 study two special types of trading ring patterns, namely the blackhole

and volcano patterns. Given a directed graph, a blackhole pattern is a group which contains a set of nodes in a way such that the average in-weight of this group is significantly larger than the average out-weight of the same group. In contrast, a volcano pattern contains a set of nodes where the average out-weight is significantly larger than the average in-weight. The problem of finding volcano patterns is a dual problem of mining blackhole patterns. Therefore, we focus on discovering the blackhole patterns and develop a blackhole mining framework. Specifically, we design two pruning schemes for reducing the computational cost by reducing both the number of candidate patterns and the average computation cost for each candidate pattern. The first pruning scheme is to exploit the concept of combination dominance to reduce the exponential growth search space. Based on this pruning approach, we develop the gBlackhole algorithm. On the other hand, the second pruning scheme follows an approximate strategy. We improve the computational efficiency by first screening out nodes with small diff-weights to reduce the size of the graph, and then mining the top-K blackhole patterns in the subgraph induced by the rest of the nodes. This approach strikes a balance between the efficiency and the completeness of blackhole mining.

Chapter 3 investigate how to predict potential risks for new contracts based on their similarities with existing ones. Existing classification methods kNN can be used to partially solve our problem. However, a critical challenge of applying this method is to define the right metric that can be used to gauge similarity (or distance) between contracts. Along this line, we extend the Mahalanobis distance metric learning framework, and formulate a constrained optimization problem such that the learnt

distance for each pair of similar points is close to their actual distance, while each pair of dissimilar points can be well separated. A key advantage of the proposed method is the ability to train model with not only continuous distance measures between contract pairs, but also the binary side information of dissimilar pairs. Finally, experimental results on real-world service contract data show that our proposed approach greatly outperforms existing benchmarks, and can provide more accurate contract risk assessment.

Chapter 4 examines the application of cluster analysis in bankruptcy pattern learning and financial statement fraud detection. By leveraging the domain knowledge in accounting area, unique features from financial statement can be extracted. Clustering technique is then applied to identify the clustering effect of bankrupt companies in different business sectors. Finally the most indicative financial features for the bankrupt companies in the business sector can be uncovered from the hidden data and validated by significant tests.

Last but not least, Chapter 5 concludes this dissertation and discuss the potential future work.

CHAPTER 2

MINING BLACKHOLE AND VOLCANO PATTERNS FOR FINANCIAL FRAUD DETECTION

Given a directed graph, the problem of blackhole mining is to identify groups of nodes, called blackhole patterns, in a way such that the average in-weight of this group is significantly larger than the average out-weight of the same group. The problem of finding volcano patterns is a dual problem of mining blackhole patterns. Therefore, we focus on discovering the blackhole patterns. Indeed, in this chapter, we develop a blackhole mining framework. Specifically, we first design two pruning schemes for reducing the computational cost by reducing both the number of candidate patterns and the average computation cost for each candidate pattern. The first pruning scheme is to exploit the concept of combination dominance to reduce the exponential growth search space. Based on this pruning approach, we develop the gBlackhole algorithm. Instead, the second pruning scheme is an approximate approach, named approxBlackhole, which can strike a balance between the efficiency and the completeness of blackhole mining. Finally, experimental results on real-world data show that the performance of approxBlackhole can be several orders of magnitude faster than gBlackhole, and both of them have huge computational advantages over the brute-force approach. Also, we show that the blackhole mining algorithm can be used to capture some suspicious financial fraud patterns.

2.1 Introduction

Government agencies, such as U.S. Securities and Exchange Commission (SEC), are facing increasing challenges for financial fraud detection. The sophisticated fraud tactics makes detecting and preventing fraud difficult, especially as the number of trading accounts and the volume of transactions grow dramatically. Indeed, the trading networks are vulnerable to these fast-growing accounts and the volume of transactions. In particular, criminals know fraud detection systems are not good at correlating user behaviors across multiple trading accounts. This weakness opens the door for cross-account collaborative fraud, which is difficult to discover, track and resolve because the activities of the fraudsters usually appear to be normal. For instance, consider a trading network with a large number of nodes and directed edges, a trader or a group of traders can perform trading only within several accounts for the purpose of manipulating the market. This kind of illegal trading activities is widely known as the trading ring pattern.

In this chapter, we study a special type of trading-ring patterns, called blackhole patterns. Given a directed graph, a blackhole pattern is a group which contains a set of nodes in a way such that the average in-weight of this group is significantly larger than the average out-weight of the same group. In contrast, a volcano pattern contains a set of nodes where the average out-weight is significantly larger than the average in-weight. In fact, we originate the blackhole and volcano patterns from real-world trading networks. For example, consider a group of traders who are manipulating the market by performing transactions on a specific stock among themselves for a

specific time period. In the first stage, these traders produce a large volume of transactions on this stock by purchasing shares from public. During this time period, the average net volume of shares of the target stock per their trading account will increase significantly, and a blackhole pattern can be observed. After the stock price goes up to a certain degree, these traders start selling off their shares to the public to earn profit. In this stage, these trading accounts form a volcano pattern which have the average out-weight significantly larger than the average in-weight.

The blackhole and volcano patterns can also be observed in other application scenarios. For example, in *sina weibo*[1] , one of the biggest micro-blog online community in China, a new type of cross-account collaborative fraudulent activity has been observed as follows. To get started, the fraudster needs a very popular account P which may take months to accumulate millions of followers. Then, he creates a number of new accounts and wants to increase the popularity of these accounts in a very short time. Let us take one such account L as an example. First, the fraudster frequently posts lots of interesting tweets under account L on a specific popular subject. Next, the account P retweets (shares) all tweets that L posts to make them visible to P's followers. A part of P's followers will start to follow L. In addition, most of the followers are likely to retweet the tweets L post. Due to the network effect, the number of L's followers will further increase. After a short period of time, L accumulates a considerable number of followers. Then, the process moves to the next stage. Each account L within the group starts to retweet other accounts' tweets in order to maximize the network effect of the whole group in the community. In this way, each L can

[1]http://weibo.com

pile a significant number of followers within a very short period of time. To sum up, the fraudster first uses P to solve the cold-start problem, and then let the accounts form a group to maximize the network effect. From the view of normal accounts, this is a fraudulent activity since it impedes the healthy development of the community. If we consider the community as a directed graph in which all accounts are nodes, and there is a directed edge from node A to node B if account A follows account B. Then, during the period of time, we can observe a blackhole pattern within that group of fraudulent accounts, which can help to prevent such kind of fraudulent activities.

We can show that the problem of detecting blackhole patterns is a dual problem of finding volcano patterns. Therefore, we focus on the problem of identifying blackhole patterns. Along this line, we develop a blackhole mining framework in which there are two pruning schemes to deal with the computational challenges. In the first pruning scheme, we introduce the *additivity* property of diff-weight to reduce the computational cost of computing diff-weight of a set of nodes. Also, we propose the concept of combination dominance to help reduce the exponentially growing search space significantly. Based on these pruning techniques, we develop the gBlackhole algorithm to find the top-K blackhole patterns. In contrast, the second pruning scheme follows an approximate strategy, which is exploited for developing the approxBlackhole algorithm. Indeed, the average access time to retrieve information from a node or an edge increases rapidly with the number of nodes. This becomes the bottleneck of the performances of gBlackhole for large graphs. Therefore, we improve the computational efficiency by first screening out nodes with small diff-weights to reduce the size of the graph, and then mining the top-K blackhole patterns in the subgraph induced by the

rest of the nodes. This approach is correct but not complete, and there is a trade-off between the efficiency and completeness.

Finally, experiments on real-world data sets are provided to evaluate the computational performances of the proposed two algorithms: gBlackhole and approxBlackhole. The results show that the approxBlackhole algorithm can be several orders of magnitude faster than the gBlackhole algorithm, and both of them have a huge computational advantage over the brute-force algorithm in terms of both the number of combinations searched and the average computational cost for each combination. The trade-off effect between efficiency and completeness of the approxBlackhole algorithm is also studied. Moreover, the proposed algorithms have been exploited for identifying some suspicious financial fraud patterns in the simulated E-mini S&P 500 futures contract trading data set from U.S. Commodity Futures Trading Commission. The result shows the effectiveness of the blackhole patterns.

2.2 Preliminaries

In this section, we introduce the basic concepts and notations that will be used in this chapter.

Consider a directed graph $G = (V, E)$ (Diestel, 2006), where V is the set of all nodes and E is the set of all edges. Assume that G has no self-loops. A directed edge e in G is denoted as $e = (x, y)$, where x and y are nodes of G and an arc is directed from x to y. Each edge e has a positive weight, denoted as ω_e, associated with this edge.

Definition 1 (connected/weakly connected) *In an undirected graph G, two nodes*

u and v are called connected if G contains a path from u to v. An undirected graph is called connected if every pair of distinct nodes in the graph is connected. A directed graph is called weakly connected if the undirected graph produced by replacing all of its directed edges with undirected edges is a connected graph.

Definition 2 (in-weight/out-weight) *Given a directed graph $G = (V, E)$, B is a set of nodes and $B \subseteq V$. Let $C = V \setminus B$. The in-weight of B is defined as: $In(B) = \sum_{e=(x,y)\in E} \omega_e$, where $x \in C$ and $y \in B$. And the definition of the out-weight of B is in the opposite direction: $Out(B) = \sum_{e=(x,y)\in E} \omega_e$, where $x \in B$ and $y \in C$.*

In the trading network scenario, if we consider the trading network on a specific stock during a period of time as a directed graph, such that the trading accounts are the nodes, the transactions between accounts are the edges, and the transaction volumes are the weights associated with the edges, then for a certain group of trading accounts (nodes) B, the in-weight of B is the total volume of shares that group of traders have purchased from the public during that period of time. Similarly, the out-weight of B is the total volume of shares that group of traders have sold to the public during that period of time.

Figure 2.1 shows an example of the in-weight and out-weight of a set of nodes. The number associated with each edge is the weight of that edge. In this figure, the in-weight of B is $6 + 5 = 11$, while the out-weight is $3 + 3 + 1 + 2 = 9$.

Now, let us get back to the intuition of blackhole and volcano patterns we have discussed in Section 2.1. In the trading network scenario, we would like to find out the cross-account collaborative fraudulent trading activities. Therefore, there should

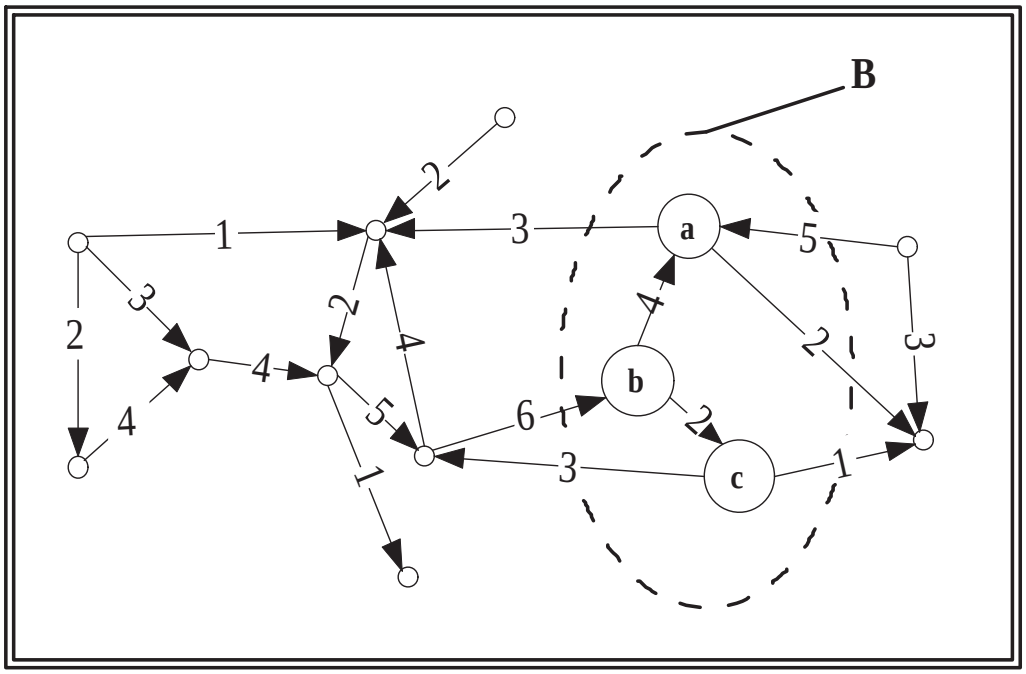


Figure 2.1: Illustration: in-weight and out-weight

be at least two trading accounts and these accounts have some sort of interactions among them. In graph terms, it means this set of nodes has at least 2 nodes and the subgraph induced by these nodes is weakly connected.

Also, in Section 2.1, we have described the two stages that a normal trading-ring pattern usually behaves. In the first stage, the average volume of shares that the group of trading accounts have purchased from the public is significantly larger than the average volume of shares that they have sold, and we consider that group of trading accounts as a blackhole pattern. In graph terms, it means the average of the difference between the in-weight and out-weight of that set of nodes is significant. In contrast, another stage for selling off the stock shares can be viewed in a very similar fashion, and we name the group of trading accounts in this stage as a volcano pattern.

Since we will refer to the difference between the in-weight and out-weight of a set

of nodes a lot in the rest of the chapter, to make our notations easy and precise, we introduce the definition of *diff-weight*.

Definition 3 (diff-weight) *Given a directed graph $G = (V, E)$, B is a set of nodes and $B \subseteq V$. The diff-weight of B is defined as: $Diff(B) = In(B) - Out(B)$.*

Based on the above understandings, we have the definition of blackhole pattern and volcano pattern in a directed graph as follows.

Definition 4 (blackhole pattern) *Given a directed graph $G = (V, E)$, a set of nodes $B \subseteq V$ form a blackhole pattern, if and only if the following two conditions are satisfied: 1) $|B| \geq 2$, and the subgraph $G(B)$ induced by B is weakly connected, and 2) $Diff(B)/|B| > \theta$, where $|B|$ is the cardinality of B, and θ is a pre-defined positive threshold and is typically a very large value.*

Definition 5 (volcano pattern) *Given a directed graph $G = (V, E)$, a set of nodes $Vol \subseteq V$ form a volcano pattern, if and only if the following two conditions are satisfied: 1) $|Vol| \geq 2$, and the subgraph $G(Vol)$ induced by Vol is weakly connected, and 2) $Diff(Vol)/|Vol| < -\theta$, where $|Vol|$ is the cardinality of Vol, and θ is a pre-defined positive threshold and is typically a very large value.*

Please note that, we use the average diff-weight instead of the diff-weight of a set of nodes in defining the blackhole and volcano patterns. The reason is that by only using the diff-weight, we may not end up with finding out the groups of nodes that we really want in some cases. Assume that a set of nodes B is a blackhole pattern we have found by the definition of using diff-weight instead of average diff-weight. By adding

another node v such that $Diff(v) > 0$ and v is connected to B, we can find another blackhole pattern B' with $Diff(B') > Diff(B)$. If we keep repeating this process, we will end up with a B' which is a group of nodes with large diff-weight and large size. However, the average diff-weight of B' may be far less than the average diff-weight of B. In real world, we are more likely interested in B rather than B'. Let us consider the trading network scenario again. B' may represent a group of thousands of trading accounts whose total net volume of shares accumulate rapidly during a certain period, while the average net volume per account may be small. These accounts are more likely to be normal accounts with normal transactions. Meanwhile, B may consist of a couple of trading accounts whose net volume of shares per account increase fast, but the total net volume may not as large as the one of B'. In this situation, B is more likely to perform a trading-ring fraud than B'. In this sense, the average diff-weight of a set of nodes is more important than the diff-weight itself. Thus, we will use the average diff-weight of a set of nodes to define blackhole and volcano patterns in this chapter.

2.3 Problem Formulation

In this section, we formulate the blackhole mining problem. First, we show that the problem of detecting blackhole patterns is a dual problem of finding volcano patterns.

Theorem 1 *The problem of detecting blackhole patterns in a directed graph is a dual problem of detecting volcano patterns in its inverse graph, and vice versa.*

Proof Consider a directed graph $G = (V, E)$. Let $G' = (V, E')$ be the *inverse graph* of G, where all the nodes in G' are the same as in G; while for each edge $e = (x, y) \in E$,

there is an edge $e' = (y, x) \in E'$, and the weight associated with e' is exactly the same as the weight associated with e. Therefore, the in-weight of a set of nodes B in G is the same as the out-weight of B in G', and vice versa. Thus, for each blackhole pattern B in G, we have $(In(B) - Out(B))/|B| > \theta$ in G. Then in G', we have $(Out(B) - In(B))/|B| > \theta$, which is equivalent to $Diff(B)/|B| < -\theta$. Also, in G', we still have $|B| > 2$ and $G(B)$ is weakly connected. Therefore, B forms a volcano pattern in G'. On the other hand, we can also prove that for each volcano pattern Vol in G', it is a blackhole pattern in G. Thus, detecting blackhole patterns in a directed graph is equivalent to detecting volcano patterns in its inverse graph. In a similar fashion, we can also prove that detecting volcano patterns in a directed graph is a dual problem of detecting blackhole patterns in its inverse graph.

According to Theorem 1, we can focus on the problem of detecting blackhole patterns in a directed graph in the rest of the chapter.

An intuitive way to formulate the problem of detecting blackhole patterns in a directed graph $G = (V, E)$ is to find out the set of blackhole patterns, denoted as $Blackhole$, such that for each element $B \in Blackhole$, it is a set of nodes in G, and B satisfies the definition of *blackhole*. On the other hand, there is no other set of nodes $B' \notin Blackhole$ such that B' satisfies the definition of *blackhole*. In sum, we would like to find out the complete and correct set of all blackhole patterns in G.

However, there are certain issues with the above problem formulation. The main issue is how should we decide the value of θ. Since the problem has a combinatorial nature such that we would like to find out the complete and correct set of all blackhole patterns in a directed graph, if we set θ too small, most likely we would end up with

a set with $O(2^n)$ blackhole patterns in it. On the other hand, if we make θ too large, most likely we would get nothing in the end. It is hard and tricky for us to determine the value of θ without trying many times, and obviously it is very inefficient and inconvenient in practice. Actually, we are much more interested in the "outliers" rather than the ordinary ones in the real world. In other words, we are more interested in the blackhole patterns with larger average diff-weights. Therefore, if we just simply find out blackhole patterns with the top-K largest average diff-weights, we will achieve our goal and make things much easier and more efficient. As a result, we have the problem formulation in this chapter as follows.

Top-K Blackhole Pattern Mining. Given a directed graph $G = (V, E)$, the goal is to find out the set of top-K blackhole patterns, denoted as $Blackhole$, such that, 1) $|Blackhole| = K$, 2) for each element $B \in Blackhole$, $B \subseteq V$, $|B| \geq 2$, and the subgraph $G(B)$ induced by B is weakly connected, and 3) for any other set of nodes $B' \notin Blackhole$, if B' satisfies condition 2), then for each element $B \in Blackhole$, we have $Diff(B)/|B| \geq Diff(B')/|B'|$.

2.4 Algorithm Design

In this section, we introduce three algorithms to mine the top-K blackhole patterns in a directed graph. To simplify the discussion, let n denote the number of nodes, m denote the number of edges, and $\alpha = m/n$ be the average in-and-out node degree of the graph.

2.4.1 A Brute-Force Approach

In this subsection, we present a brute-force approach for detecting blackhole patterns in a directed graph. Since the goal is to find out the set of top-K blackhole patterns, the intuition is really simple: we check out all combinations of nodes in V from size 2 to n using the exhaustive search method. For each combination B, if the subgraph $G(B)$ induced by B is weakly connected, we keep a record of B as well as its average diff-weight in a list. Finally, we sort the list and output the blackhole patterns with the top-K largest average diff-weights.

In practice, in order to save space and make the algorithm more efficient, we maintain a priority queue *Blackhole* of size K to record the current top-K blackhole patterns along with their average diff-weights. It is initialized with K empty sets with key value of $-\infty$ and the elements in it are sorted by the descending order of their key values. During the whole procedure, *Blackhole* keeps updated to make sure it has recorded the current top-K blackhole patterns among combinations which have already been checked so far. Finally, when the exhaustive search procedure completes, *Blackhole* will record the set of blackhole patterns with the top-K largest average diff-weights as the result.

Figure 2.2 shows the pseudo code of the brute-force approach to detect the top-K blackhole patterns in a directed graph G, in which $Subset_i = \{B \mid B \subseteq V, and\ |B| = i\}$ is the set of all combinations of nodes in V of size i, and the function *Is_Connected()* checks whether a directed graph is weakly connected or not.

ALGORITHM *Brute-Force* $(G = (V, E),\ K)$

Input:

G: the input directed graph

V: the set of all nodes

E: the set of all edges

Output:

Blackhole: priority queue of current top-K blackholes

1. Initialize *Blackhole*
2. **for** $i \leftarrow 2$ **to** n **do**
3. **for** each combination of nodes $B \in Subset_i$ **do**
4. **if** *Is_Connected(*$G(B)$*)* **and**
5. $Diff(B)/|B| > Blackhole.extract\_min()$ **then**
6. $Blackhole.delete\_min()$
7. $Blackhole.insert(B, Diff(B)/|B|)$
8. **end if**
9. **end for**
10. **end for**
11. **return** *Blackhole*

Figure 2.2: A brute-force approach

The computational complexity of the brute-force approach is $T = O(M \cdot N)$, where $N = 2^n - n - 1$ is the number of combinations of size no less than 2, and M is the average computational cost for each combination B. We use depth-first search (DFS) to check whether B is weakly connected or not, and meanwhile we can compute $Diff(B)$. During the DFS, each node in B is accessed once as well as all its edges. Therefore, the computational cost of each combination B is $M = O((\alpha + 1) \cdot |B|)$. Since the average size of B is $n/2$, then the computational complexity of the brute-force approach is $T = O(M \cdot N) = O((m + n) \cdot 2^n)$.

This approach is obviously computationally prohibitive due to its combinatorial nature. As the number of nodes n increases, the number of combinations increases exponentially. This makes the brute-force approach unrealistic to use in practice when n is large. Since $T = O(M \cdot N)$, the computational complexity can be reduced by decreasing either the number of combinations or the average computational cost for each combination. Along this line, we introduce two pruning schemes in the following subsections to help reduce the computational cost.

2.4.2 The Additivity Property of Diff-weight

In brute-force approach, we first compute $Diff(B)$ by calculating $In(B)$ and $Out(B)$ separately, and then do the subtraction. As mentioned in last subsection, the computational cost of calculating $Diff(B)$ is $O((\alpha + 1) \cdot |B|) = O(m + n)$, which is very time consuming. Consider that we will need to calculate $Diff(B)$ very frequently in the blackhole mining process, we need a more efficient way for that.

The main challenge is that the set of edges between B and C are changing dy-

namically for different set of nodes. Is it possible that we can pre-compute a value for each node in V separately, and for each set of nodes B, we can compute $Diff(B)$ by adding up the values of each node in B, regardless the set of edges between B and C? The following theorem solves this problem. We name this theorem as *additivity property of diff-weight*.

Theorem 2 *(Additivity Property of Diff-weight) The diff-weight of a set of nodes B equals to the summation of the diff-weight of each node in B, i.e.* $Diff(B) = \sum_{v \in B} Diff(v)$.

Proof

$Diff(B) = In(B) - Out(B)$

$= \sum_{x \in C, y \in B} \omega_e - \sum_{x \in B, y \in C} \omega_e$

$= (\sum_{x \in C, y \in B} \omega_e + \sum_{x \in B, y \in B} \omega_e) - (\sum_{x \in B, y \in C} \omega_e + \sum_{x \in B, y \in B} \omega_e)$

$= \sum_{x \in C \bigcup B, y \in B} \omega_e - \sum_{x \in B, y \in C \bigcup B} \omega_e$

$= \sum_{x \in V, y \in B} \omega_e - \sum_{x \in B, y \in V} \omega_e$

$= \sum_{v \in B} In(v) - \sum_{v \in B} Out(v)$

$= \sum_{v \in B} (In(v) - Out(v))$

$= \sum_{v \in B} Diff(v)$, where $e = (x, y) \in E$

Let us illustrate this property using Figure 2.1 as an example. In this figure, we have $Diff(a) = (4 + 5) - (3 + 2) = 4$, $Diff(b) = 6 - (4 + 2) = 0$, and $Diff(c) = 2 - (3 + 1) = -2$. Thus, we have $\sum_{v \in B} Diff(v) = 4 + 0 + (-2) = 2$, which equals

to $Diff(B) = (6 + 5) - (3 + 3 + 2 + 1) = 2$. This simple example illustrates the *additivity property of diff-weight*.

Based on this property, we can compute $Diff(B)$ in a different way as the following. First, we compute the diff-weight of each node in V, and record the result in a list L. This is a one-time effort and the computational cost is $O(|V| + |E|) = O(m + n)$. Then, for each combination B, $Diff(B)$ can be calculated by adding up the diff-weight of each node in B, which can be retrieved from the list L. By using this method, the computational cost can be reduced to $O(|B|)$. As a result, for the **if** statement in the brute-force approach, we check the condition $Diff(B) > Blackhole.min()$ first, only if it is true, we then check the condition *Is_Connected(*$G(B)$*)*, since the computational cost of the former one is smaller.

2.4.3 The Measure of Enumeration

In this subsection, we introduce a measure to fulfill the procedure of enumerating all combinations of nodes in V from size 2 to n. Basically, it consists of two steps, (1) sort the nodes in the graph by a certain criteria, and (2) for each $i = 2, 3, \ldots, n$, find a particular order to list all combinations of nodes with size i.

For step (1), recall that in last subsection, we have calculated the diff-weight of each node, and know that the value of $Diff(B)$ equals to the summation of the diff-weight of each node in B. Since our goal is to find out blackhole patterns with the top-K largest average diff-weight, it implies that nodes with larger diff-weight have higher probability to appear in the top-K blackhole patterns. Therefore, it is very natural to sort the nodes in V by their diff-weights in a decreasing order, and

start the enumeration procedure from combinations consisting of nodes with larger diff-weights. We denote the nodes after sorting as $\{v_1, v_2, \ldots, v_n\}$, where v_1 has the largest diff-weight while v_n has the smallest.

For step (2), we use the lexicographic order (a.k.a. dictionary order) to list all combinations of nodes of size i (Knuth, 2011), not only because it has been widely used, more importantly, enumerating combinations in this order is consistent with our intuition of starting from nodes with larger diff-weights. Next, we briefly describe the basic idea of the lexicographic order.

Given a general alphabet set $S = \{s_1, s_2, \ldots, s_n\}$, the elements in this set are ordinal, which means each one of them is comparable with other elements by a pre-defined order. We denote this order as the lexicographic order of this general alphabet set S. Without the loss of generality, we let $s_1 \prec s_2 \prec \ldots \prec s_n$, where the notion "$\prec$" here denotes lexicographically less than. In our case, we consider V as the alphabet set, and define the lexicographic order in V as $v_1 \prec v_2 \prec \ldots \prec v_n$, where $Diff(v_1) \geq Diff(v_2) \geq \ldots \geq Diff(v_n)$.

Since the order of elements in a combination does not matter, without the loss of generality, we assume that $\beta_1 \prec \beta_2 \prec \ldots \prec \beta_i$ for the combination $\beta = (\beta_1, \beta_2, \ldots, \beta_i)$ of size i. In the rest of the chapter, we assume that the elements in a combination are lexicographically ordered if not mentioned otherwise. Next, we give the definition of the lexicographic order of two combinations.

Definition 6 (lexicographic order for combinations) *Let* $\beta = (\beta_1, \beta_2, \ldots, \beta_i)$ *and* $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_i)$ *be two combinations of the same size* i *over* S. *We say that*

β is lexicographically less than γ, denoted as $\beta \prec \gamma$, if and only if for $j = 1, 2, \ldots, i$, there exist some j, such that $\beta_j \neq \gamma_j$, and for the smallest such j, $\beta_j \prec \gamma_j$.

The lexicographic order generates combinations in a particular order such that for any two combinations β and γ with the same size, if $\beta \prec \gamma$, β is generated earlier than γ. For example, the lexicographic order of the combinations of size 3 over alphabet set $\{1, 2, 3, 4, 5\}$ is $(1, 2, 3)$, $(1, 2, 4)$, $(1, 2, 5)$, $(1, 3, 4)$, $(1, 3, 5)$, $(1, 4, 5)$, $(2, 3, 4)$, $(2, 3, 5)$, $(2, 4, 5)$, $(3, 4, 5)$, if we define the lexicographic order for this example as $1 \prec 2 \prec 3 \prec 4 \prec 5$. By using the lexicographic order, we can therefore enumerate all combinations of nodes of size i for each $i \geq 2$.

In practice, we employ the recursion of depth i to implement the enumeration process. We use a pointer *pt* to indicate which node at which level is currently being visited in the recursion. Level 1 is the first level of the recursion while level i is the deepest. *pt* initially points to node v_1 at level 1. When *pt* is pointing to node v_j ($k \leq j \leq n - i + k$) at level k ($k = 1, 2, \ldots, i - 1$), we let v_j be the kth element of combination B and move *pt* to node v_{j+1} at level $k + 1$. When *pt* jumps out of level $k + 1$, we erase v_j from the kth element of B first, and if $j < n - i + k$, we move *pt* to node v_{j+1} at level k and repeat the same process as for node v_j at level k, otherwise, we let *pt* jump out of level k and return to level $k - 1$. If *pt* is pointing to node v_j at level i, the process is similar to the one for previous levels, except we check out whether the current combination B is a top-K blackhole pattern instead of diving into the deeper level, since we have already reached the last level of the recursion. When *pt* jumps out of level 1 and returns to level 0, the procedure completes and all

combinations of size i have been enumerated in the lexicographic order. More details are available in the next subsection.

2.4.4 A Pruning Scheme

In this subsection, we propose a pruning scheme to control the exponential growth of the number of combinations, which can be helpful to reduce the computational cost.

First, we introduce the concept of combination dominance.

Definition 7 (combination dominance) *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a general alphabet set. $\beta = (\beta_1, \beta_2, \ldots, \beta_i)$ and $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_i)$ are two combinations of the same size i over S. We say that β dominates γ, denoted as $\beta \vdash \gamma$, if and only if $\beta_j \preceq \gamma_j$ for any $j = 1, 2, \ldots, i$, and for at least one j, $\beta_j \prec \gamma_j$.*

Again, we take the alphabet set $\{1, 2, 3, 4, 5\}$ as an example. Among all combinations of size 3 over this set, $(1, 2, 4) \vdash (1, 2, 5)$, $(1, 3, 4) \vdash (2, 3, 5)$, $(1, 2, 3)$ dominates any other combinations, $(3, 4, 5)$ is dominated by any other combinations, $(1, 4, 5)$ and $(2, 3, 4)$ cannot dominate each other.

Next, we introduce several lemmas to help prune the exponentially growing search space.

Lemma 1 *If we consider V as the alphabet set, and define the lexicographic order in V as $v_1 \prec v_2 \prec \ldots \prec v_n$, then for any two combinations of nodes with the same size i over V, denoted as B and B', if $B \vdash B'$, then we have $Diff(B)/|B| > Diff(B')/|B'|$.*

Proof Recall that in Section 2.4.3, we have sorted the nodes in V by their diff-weights in a decreasing order, thus, 1) for any two nodes v and v' in V, if $v \prec v'$, then we have

$Diff(v) > Diff(v')$. In addition, since $B \vdash B'$, if we denote $B = (v_{B_1}, v_{B_2}, \ldots, v_{B_i})$ and $B' = (v_{B'_1}, v_{B'_2}, \ldots, v_{B'_i})$, according to Definition 7, we have 2) $v_{B_j} \preceq v_{B'_j}$ for any $j = 1, 2, \ldots, i$, and for at least one j, $v_{B_j} \prec v_{B'_j}$. Therefore, based on 1) and 2), we can get $Diff(v_{B_j}) \geq Diff(v_{B'_j})$ for any $j = 1, 2, \ldots, i$, and for at least one j, $Diff(v_{B_j}) > Diff(v_{B'_j})$. According to the *additivity property of diff-weight* and since $|B| = |B'|$, we have $Diff(B)/|B| > Diff(B')/|B'|$.

Lemma 2 *Let β and γ be two combinations of size i over the same alphabet set S. If $\beta \vdash \gamma$, then $\beta \prec \gamma$.*

Proof According to Definition 7, if $\beta \vdash \gamma$, then $\beta_j \preceq \gamma_j$ for any $j = 1, 2, \ldots, i$, and for at least one j, $\beta_j \prec \gamma_j$. Therefore, there exists at least one j such that $\beta_j \neq \gamma_j$, and for the smallest j among them, $\beta_j \prec \gamma_j$. According to Definition 6, we have $\beta \prec \gamma$.

Lemma 1 and Lemma 2 can be very useful to prune the exponentially growing search space. In the brute-force approach, if the average diff-weight of a combination B is no greater than the current minimum key in the priority queue *Blackhole*, we simply skip B and move forward to the next combination. However, according to Lemma 1, for any combination B' which is dominated by B, we have $Diff(B')/|B'| < Diff(B)/|B| \leq Blachole.extract\_min()$. Thus, there is no chance for such B' to appear in *Blackhole*. In addition, according to Lemma 2, $B' \succ B$, which indicates that B' will be checked later than B. Based on the above analysis, we can have the following theorem as the guideline of the pruning scheme.

Theorem 3 *(Pruning Strategy) For a combination B of size i, if we skip B since the average diff-weight of B is no greater than the current minimum key in the priority*

queue Blackhole, then for any combination B' which is dominated by B, it can be simply pruned without checking.

Although we would like to prune all combinations that are dominated by B, we also need to consider the cost of finding them out. If we have to check whether each upcoming combination is dominated by B, then this pruning strategy will have no computational savings. We need to find a simple and systematic way that can help to prune combinations which are dominated by B as many as possible with little cost, even though a few of them are missed and have to be checked out later. Along this line, we have the following pruning rule.

Lemma 3 *(Pruning Rule) For a size-i combination $B = (v_{B_1}, v_{B_2}, \ldots, v_{B_i})$, if we skip B for the reason that $Diff(B)/|B| \leq Blachole.extract\_min()$, then we can let pointer pt jump out of level i immediately. If there exists a k $(1 \leq k \leq i-1)$ such that $B_{k+1} - B_k = B_{k+2} - B_{k+1} = \cdots = B_i - B_{i-1} = 1$, then pt can directly return to level $k-1$, otherwise, pt returns to level $i-1$.*

Proof If there exists a k $(1 \leq k \leq i-1)$ such that $B_{k+1} - B_k = B_{k+2} - B_{k+1} = \cdots = B_i - B_{i-1} = 1$, consider combination $B' = (v_{B'_1}, v_{B'_2}, \ldots, v_{B'_i})$, where $B_j < B'_j \leq n-i+j$ $(j = k, k+1, \ldots, i)$ and $B_j = B'_j$ $(j = 1, 2, \ldots, k-1)$ if $k > 1$. It is clear that $B \vdash B'$. Since $Diff(B)/|B| \leq Blachole.extract\_min()$, according to the pruning strategy, we can prune all such B's without checking. Note that these B's are combinations that directly follow B in the lexicographic order. In the recursion, we can do this pruning by simply let *pt* jump out of level i, and return directly to level $k-1$. When there is no such k, the proof is very similar and we skip it here.

ALGORITHM *Find-Combinations* $(L, n, i, j, k, B, Blackhole)$

Input:

L: list of diff-weights of each node; n: number of nodes

i: size of combination; j: starting node index

k: current level number; B: current combination of nodes

Blackhole: priority queue of current top-K blackholes

```
1.  for p ← j to n − i + k do
2.    if k < i then
3.      B[k] = (p, L[p])
4.      Find-Combinations(L, n, i, p + 1, k + 1, B, Blackhole)
5.      if B[k + 1].index − B[k].index == 1 then
6.        break
7.    else
8.      B[k] = (p, L[p])
9.      if Diff(B)/|B| > Blackhole.extract_min() then
10.       if Is_Connected(G(B)) then
11.         Blackhole.delete_min()
12.         Blackhole.insert(B, Diff(B)/|B|)
13.     else
14.       break
```

Figure 2.3: The find-combinations algorithm

Then, we can incorporate this pruning rule into the enumeration process described in Section 2.4.3 to help reduce the search space. We name this pruning scheme as the *Find-Combinations* algorithm. Basically, this pruning scheme follows a general sort-and-prune strategy, which is widely used for the algorithm design (Cormen, Leiserson, Rivest, & Stein, 2009), such as the development of association rule mining algorithms (Tan et al., 2005). The pseudo code is given in Figure 2.3. B is an array of length i which records the current combination of nodes, and is initialized empty. Each element in it is a structure which consists of the index of a node in L after sorting ($B[k].index$), and the diff-weight of the node ($B[k].weight$). All array indices in the pseudo code start from 1.

Figure 2.4 gives an example to illustrate how our pruning scheme works. In the example, we would like to detect the top-2 blackhole patterns in a directed graph. Figure 2.4(a) shows the input graph and the corresponding list of diff-weights of each node after sorting. The procedure of checking out all combinations of size 3 by using our pruning scheme is given in the following subfigures. The priority queue *Blackhole* in each subfigure records the current top-2 blackhole patterns as well as their diff-weights. Since the procedure is very simple and the figures themselves are quite self-explained, we simply skip the explanation and leave it to the readers. Note that finally we only need to check 5 combinations out of all $C_6^3 = 20$ combinations of size 3 by using this pruning scheme.

Finally, we would like to discuss the main reasons why we develop this pruning scheme to solve the top-K blackhole mining problem. First of all, there is no pre-determined "global" order for the average diff-weights among all combina-

tions of nodes over V. For example, let $V = \{v_1, v_2, v_3, v_4, v_5\}$ where $Diff(v_1) \geq Diff(v_2) \geq \ldots \geq Diff(v_5)$. Then for combinations $B = (v_1, v_3, v_5)$ and $B' = (v_1, v_5)$, if $Diff(v_3) > 0$, we have $Diff(B)/|B| > Diff(B')/|B'|$, otherwise, $Diff(B)/|B| < Diff(B')/|B'|$. Therefore, for different input graphs, the order of $Diff(B)/|B|$ and $Diff(B')/|B'|$ cannot be determined without checking them out. Even for combinations of the same size, there is still no such "global" order. For example, let $B'' = (v_2, v_3, v_4)$, for case $Diff(v_1, \ldots, v_5) = (10, 6, -4, -5, -7)$, we have $Diff(B)/|B| > Diff(B'')/|B''|$. However, if $Diff(v_1, \ldots, v_5) = (10, 6, 4, -7, -13)$, we have the opposite order. Thus, for different input graphs, the order of $Diff(B)/|B|$ and $Diff(B'')/|B''|$ still cannot be determined until checking them out. Therefore, in order to keep the process simple, organized and systematic, we use the measure introduced in Section 2.4.3 to enumerate all combinations of nodes in V from size 2 to n and check out their eligibility for the top-K blackhole patterns in lexicographic order, and meanwhile, we introduce the concept of combination dominance as a "local" order to help develop the pruning scheme to reduce the search space.

2.4.5 The gBlackhole Approach

While the search space has been reduced by using the pruning scheme introduced in last subsection, it is still possible to apply other pruning strategies to the mining process. Note that for a node $v \in V$, v can only form blackhole patterns with nodes within the same weakly connected component (WCC) in G, since the subgraph induced should be weakly connected. Therefore, if a directed graph has several WCCs, a divide-and-conquer strategy can be exploited by first identifying these WCCs, then

applying the pruning scheme on each of them, and finally merging all the results. This pruning strategy can divide a large exponentially growing search space into several smaller ones, and thus can further reduce the computational cost.

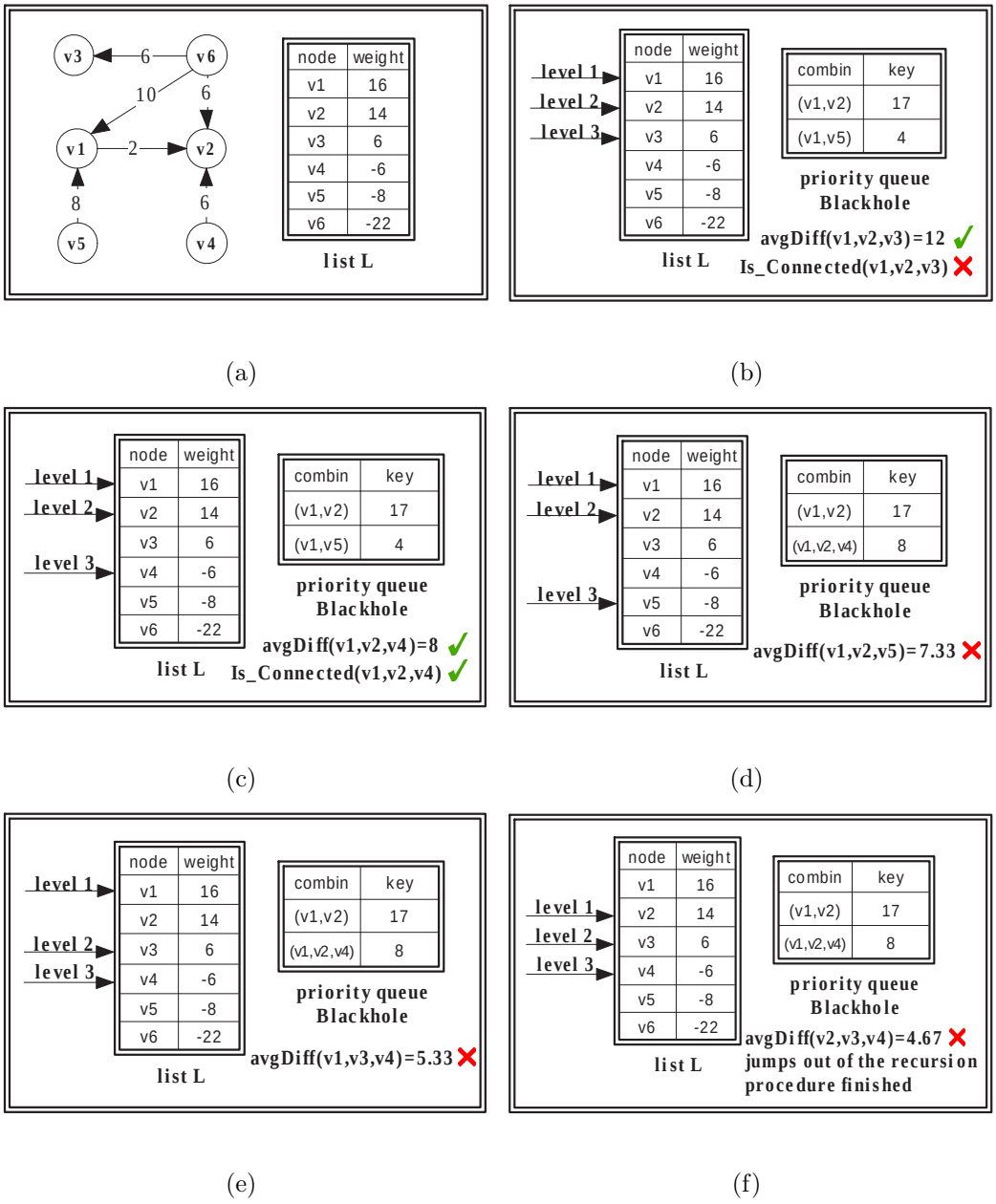


Figure 2.4: An example to illustrate the pruning scheme

Along this line, we combine the pruning scheme based on combination dominance

with this divide-and-conquer strategy and develop an algorithm, named *gBlackhole*, to detect the top-K blackhole patterns in a directed graph. Figure 2.5 shows the pseudo code of this approach. In this approach, we start with the WCC which contains the node with the largest diff-weight, and find out the top-K blackhole patterns in it. Next, we check out all the combinations of nodes in the second largest WCC by comparing them with the current top-K blackhole patterns. The above process repeats until we have found all WCCs.

The theoretical computational complexity of the gBlackhole algorithm is hard to analyze. As discussed in Section 2.4.1, the computational complexity can be expressed as $T = O(M \cdot N)$, where N is the number of combinations searched, and M is the average computational cost for each combination. As shown in the experiment section, the gBlackhole algorithm has a huge computational advantage over the brute-force approach, since the pruning effect in the gBlackhole algorithm is significant in terms of the number of pruned combinations.

2.4.6 The approxBlackhole Approach

The gBlackhole approach is shown to be an efficient approach that can reduce the computational complexity dramatically. However, as the number of nodes and edges of the graph increases, the average access time to retrieve information from a node or an edge increases rapidly, especially when the graph is too large to fit into the memory. Thus, the dramatic increase of M, which is the average computational cost for each combination, becomes the bottleneck of the performance of the gBlackhole algorithm for large graphs.

ALGORITHM *gBlackhole* $(G = (V, E),\ K)$

Input:

G: the input directed graph

V: the set of all nodes

E: the set of all edges

Output:

Blackhole: priority queue of current top-K blackholes

1. Initialize *Blackhole*
2. **for** each node $v \in V$ **do**
3. $Diff(v) \leftarrow In(v) - Out(v)$
4. **end for**
5. Sort all nodes descendingly by their diff-weights in L
6. **for** each weakly connected component WCC in G **do**
7. **for** $i \leftarrow 2$ **to** n_{WCC} **do**
8. Initialize B
9. *Find-Combinations*$(L, n, i, 1, 1, B, Blackhole)$
10. **end for**
11. **end for**
12. **return** *Blackhole*

Figure 2.5: The gBlackhole approach

The intuition of reducing the average access time to a large graph is to reduce the size of that graph. Since we are only interested in blackhole patterns with the top-K largest average diff-weights, it is not necessary for us to search through the entire graph. Most of the top-K blackhole patterns only consist of nodes with very large diff-weights. Therefore, the chance for nodes with small diff-weights to appear in the top-K blackhole patterns is very tiny so that we can filter them out safely in an early stage. In this way, we can reduce the size of the graph and significantly improve the computational efficiency.

We develop a new efficient algorithm according to the analysis above. We only reserve nodes with top p% largest diff-weights, and filter out the rest of them, where p is a user specified parameter. Then, we discover the top-K blackhole patterns in the subgraph induced by these nodes. Please note that, the diff-weights of the nodes in the subgraph are the same as their corresponding diff-weights in the original graph. We use the graph structure of the subgraph and node weights from the original graph. Clearly, this approach is correct but not complete. There is a trade-off between the efficiency and completeness. We name this approximation approach as the *approxBlackhole* algorithm.

Experimental results in the next section will show that the approxBlackhole algorithm can further reduce the computational complexity over the gBlackhole algorithm, in terms of both the number of combinations searched and the average computational cost for each combination. The trade-off effect between efficiency and completeness will also be discussed.

2.5 Experimental Results

In this section, we provide an empirical study to evaluate the performances of brute-force, gBlackhole and approxBlackhole algorithms. Here, we set $K = 100$ in all the experiments.

2.5.1 The Experimental Setup

Experimental Data. The experiments were conducted on four real-world data sets: `Wiki`, `Citation`, `Slashdot`, and `Trading`. Table 2.1 shows some basic characteristics of these data sets.

Table 2.1: Data characteristics

Data set	# nodes	# egdes	avgDegree(α)	# WCC
`Wiki`	7,115	103,689	14.57	24
`Citation`	34,546	421,578	12.20	61
`Slashdot`	77,360	905,468	11.70	1
`Trading`	100	264,600	2646	1

`Wiki` **Data Set.** There are 7,115 nodes and 103,689 edges in the `Wiki` data set (Leskovec, Huttenlocher, & Kleinberg, 2010a) (Leskovec, Huttenlocher, & Kleinberg, 2010b). The network contains all administrator elections and vote history data from the inception of Wikipedia till January 2008. Nodes in the network represent Wikipedia users and a directed edge from node i to node j represents that user i voted on user j. Since there are no weights associated with the edges in the original data set, we assign the weight of each edge to be 1.

`Citation` **Data Set.** There are 34,546 nodes and 421,578 edges in the `Citation` data set (Leskovec, Kleinberg, & Faloutsos, 2005) (Gehrke, Ginsparg, & Kleinberg, 2003). Arxiv HEP-PH (high energy physics phenomenology) citation graph is from the e-print arXiv and covers all the citations within a dataset of 34,546 papers with 421,578 edges. If a paper i cites paper j, the graph contains a directed edge from i to j. If a paper cites, or is cited by, a paper outside the dataset, the graph does not contain any information about this. The `Citation` data set covers papers in the period from January 1993 to April 2003. For this data set, we also assign the weight of each edge to be 1.

`Slashdot` **Data Set.** There are 77,360 nodes and 905,468 edges in the `Slashdot` data set (Leskovec, Lang, Dasgupta, & Mahoney, 2008). Slashdot is a technology-related news website known for its specific user community. The website features user-submitted and editor-evaluated current primarily technology oriented news. In 2002 Slashdot introduced the Slashdot Zoo feature which allows users to tag each other as friends or foes. The network contains friend/foe links between the users of Slashdot. The network was obtained in November 2008. Also, we assign the weight of each edge to be 1.

`Trading` **Data Set.** This data set was generated by U.S. Commodity Futures Trading Commission (CFTC), which simulates the transaction-level data for regular transactions in the E-mini S&P 500 futures contract market (Adamic, Brunetti, Harris, & Kirilenko, 2010). The E-mini S&P 500 futures contract is a highly liquid, fully electronic, cash-settled contract traded on the CME GLOBEX trading platform. It is designed to track the price movements of the S&P 500 Index - the most widely

followed benchmark of stock market performances.

This data set is simulated based on the data characteristics of the real-world transaction data. There are 5,163,274 transactions among 100 simulated trading accounts for 20 consecutive trading days. Each transaction has the following data fields: date and time, unique transaction ID, executing trading account, opposite trading account, buy or sell flag (for the executing trading account), price, and quantity. We construct a trading network based on transactions in one particular trading day, which has 264,600 transactions. Each trading account represents a node in the network, and a directed edge from node i to node j represents that there is a transaction between account i and j, such that i is in a short position and j is in a long position. The weight associated with each edge is the trading volume of that transaction.

Experimental Platform. All the experiments were performed on a Dell Optiplex 960 Desktop with Intel Core 2 Quad Processor Q9550 (2.83 GHz) and 4 GB of memory running the Windows XP Professional Service Pack 3 OS.

Experimental Tool. We used *LEDA (Library of Efficient Data types and Algorithms)* (Mehlhorn & Naher, 1999), which is one of the best resources available to support combinatorial computing, as our experimental tool. *LEDA* offers a complete collection of well-implemented C++ data structures and types, especially for graphs. It supports all the basic graph operations in an efficient way. The data structure that is used to represent a directed graph in *LEDA* is the adjacency list.

2.5.2 The Pruning Effect on the Number of Combinations

In this subsection, we provide a comparison of the number of combinations, which have been searched, using different algorithms on different data sets. The results are shown in Table 2.2. *approx-10%* denotes the approxBlackhole algorithm with parameter $p = 10\%$.

Table 2.2: The number of combinations searched

Algorithm	`Wiki`	`Citation`	`Slashdot`
brute-force	6.74E+2141	2.41E+10399	4.79E+23287
gBlackhole	25,579	128,513	155,251
approx-20%	11,228	79,431	25,372
approx-10%	9,231	68,147	18,192
approx-5%	8,112	62,791	12,747
approx-2%	7,356	57,600	8,134

In Table 2.2, we can see that the pruning effect of both gBlackhole and approx-Blackhole algorithms are very significant. The search space can be reduced thousands of orders of magnitude by applying the gBlackhole algorithm to all three data sets. In addition, by using the approxBlackhole algorithm, the search space can be further pruned out by $50\% - 95\%$. The experimental results validate our theoretical analysis in previous sections. We will give more detailed comparisons between the gBlackhole and approxBlackhole algorithms in the following subsections.

2.5.3 The Pruning Effect on the Average Combination Size

In this subsection, we provide a comparison of the average combination size using different algorithms on different data sets. As shown in Figure 2.6, the average combination size in brute-force algorithm is the largest, while the average size in approxBlackhole algorithm is the smallest, and the average size decreases as p gets smaller. This result is consistent with the intuition, since the number of nodes in the induced subgraph is getting smaller as p decreases. The results show that the effect of reducing the average combination size is significant.

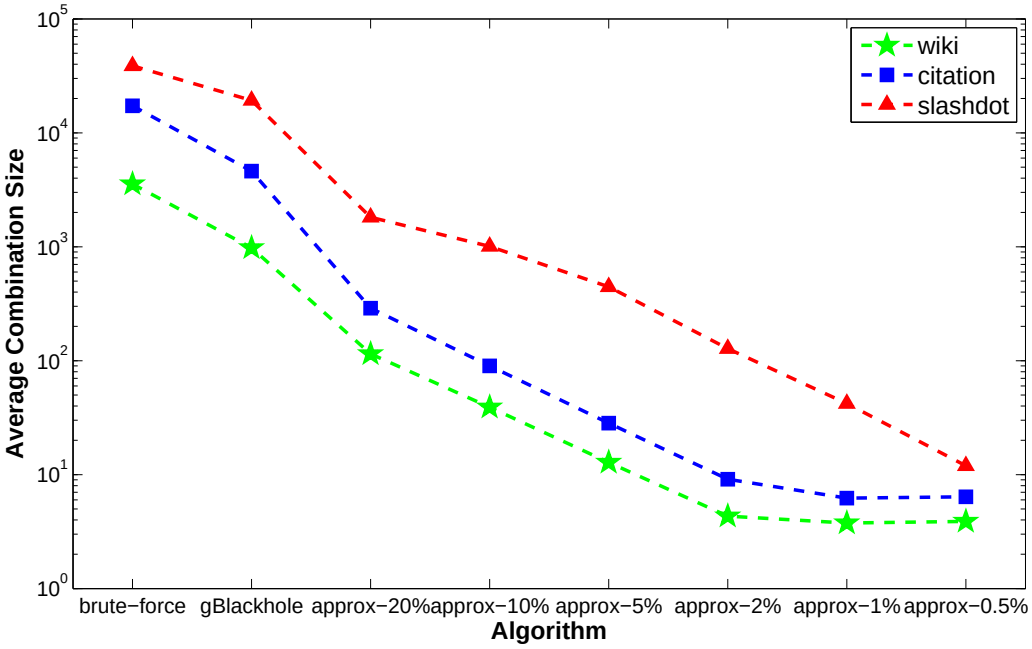


Figure 2.6: The pruning effect of the average combination size

2.5.4 The Performances of the gBlackhole Algorithm

In this subsection, we provide experiments to study how the running time of the gBlackhole algorithm changes with different input graph sizes. Here, we perform

sampling on a large graph to derive a series of graphs with different sizes. In this way, this group of graphs will have the same graph properties, which makes the comparison less biased.

Regarding the sampling method, we refer to Leskovec and Faloutsos's work in 2006 (Leskovec & Faloutsos, 2006). We chose the *Random Walk* (RW) sampling method, since it performs the best among all the sampling methods to meet the scale-down criteria, which measure how good the sampled graph inherits the graph properties from the original graph. Next, we sampled a series of graphs from the `Slashdot` data set with different number (percentage) of nodes, and applied the gBlackhole algorithm on each of them. We repeated the procedure for 10 times, and average the corresponding results as the performance of the gBlackhole algorithm on input graphs with different sizes.

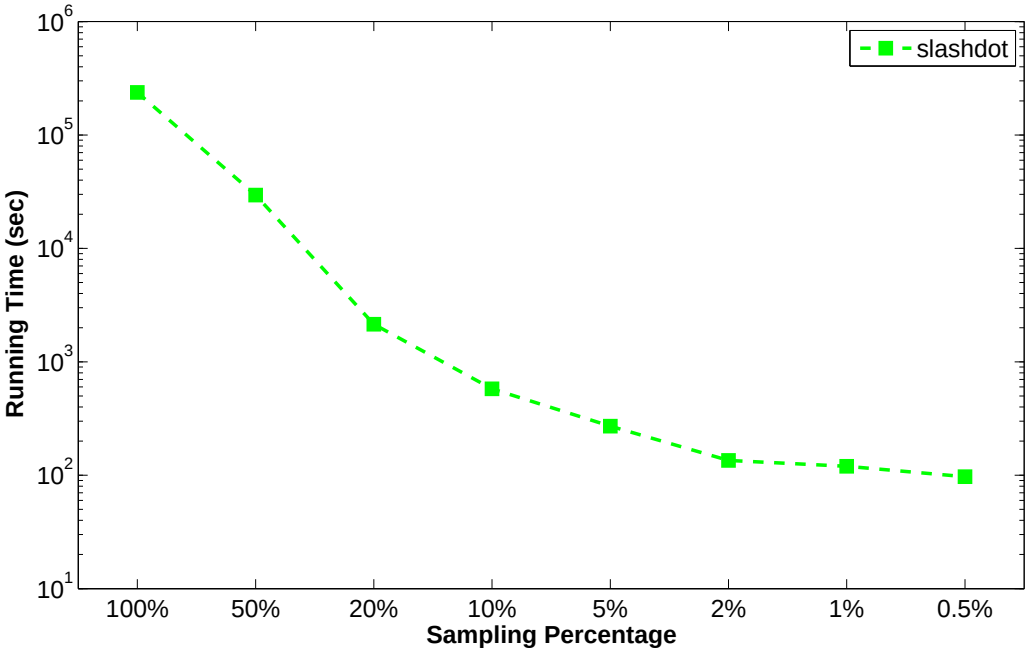


Figure 2.7: The running time of gBlackhole on data sets with different sizes

The experimental results are shown in Figure 2.7. In general, the running time of the gBlackhole algorithm decreases in an approximately exponential way as the size of the graph shrinks. This result proves our theoretical analysis in Section 2.4, because the size of the input graph decreases, both the number of combinations (N) and the average combination size (M) decrease. Especially, the value of N follows an approximately exponential decrease, which results in the exponential decrease of the running time of the gBlackhole algorithm.

2.5.5 gBlackhole vs. approxBlackhole

In this subsection, we compare the performances of gBlackhole and approxBlackhole algorithms on different data sets.

As shown in Figure 2.8, the running time of the approxBlackhole algorithm is much smaller than the gBlackhole algorithm over all three data sets, which can save at least 90% of the running time, regardless what the value of p is. The results are consistent with the theoretical analysis in previous sections.

Meanwhile, as shown in Table 2.3, the completeness rates of detecting the top-K blackhole patterns in different data sets by using the approxBlackhole algorithm are excellent. The completeness rate is calculate as C/K, where C is the number of common blackhole patterns detected by both gBlackhole and approxBlackhole algorithms. For `Wiki` and `Slashdot` data sets, the completeness rates are all 100% for different p. For the `Citation` data set, the results are also very good. Even for the case of $p = 0.5\%$, although the completeness rate is only 69%, the approxBlackhole algorithm finds out all the top 40 blackhole patterns, which are much more important

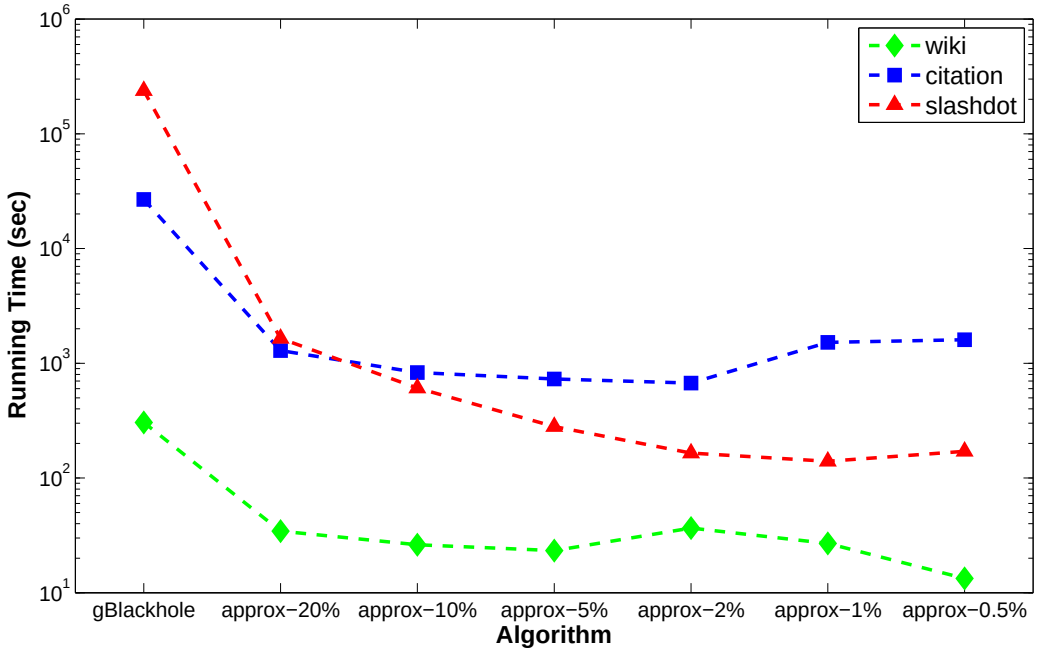


Figure 2.8: A comparison of gBlackhole and approxBlackhole algorithms

than the missing 31 blackhole patterns.

The performance of the approxBlackhole algorithm depends on the properties of the input graph. Consider Figure 2.8 again, we can observe that the running time of the approxBlackhole algorithm on the `Citation` data set increases significantly when p becomes very small. This is very different from what we have observed in the other two data sets. Also, the running time of the approxBlackhole algorithm on `Citation` is much longer than on `Slashdot` when p is very small, although the size of `Slashdot` is much larger. Note that `Citation` is a citation network, while `Slashdot` is a social network, they have different graph properties. Figure 2.9 shows how the average node degree of the two networks evolves as p decreases, which we think may be one of the reasons to explain the observations above. In the figure, comparing with `Slashdot`, we can see that the average node degree of `Citation` decreases continuously and

Table 2.3: The completeness rate of the approxBlackhole algorithm

Algorithm	`Wiki`	`Citation`	`Slashdot`
approx-20%	100%	100%	100%
approx-10%	100%	100%	100%
approx-5%	100%	100%	100%
approx-2%	100%	100%	100%
approx-1%	100%	81%	100%
approx-0.5%	100%	69%	100%

becomes very small (less than 4) as p decreases, which makes the subgraph induced from `Citation` less and less connected comparing to the original graph. Though the number of nodes decreases as p decreases, there are considerable combinations, which used to be weakly connected and could be inserted into the *Blackhole* priority queue after they had been checked out, are no longer weakly connected in the subgraph any more when p gets small. Therefore, these combinations are not qualified to become a blackhole pattern and have to be skipped after checking. The number of combinations that have to be checked will increase, which makes the running time becomes longer when p is very small. It may also be the reason why the completeness rate decreases when p gets small.

2.5.6 Blackhole Patterns in Trading Network

In this subsection, we show the results of applying the blackhole mining framework on the `Trading` data set. The running time of the gBlackhole algorithm on this data

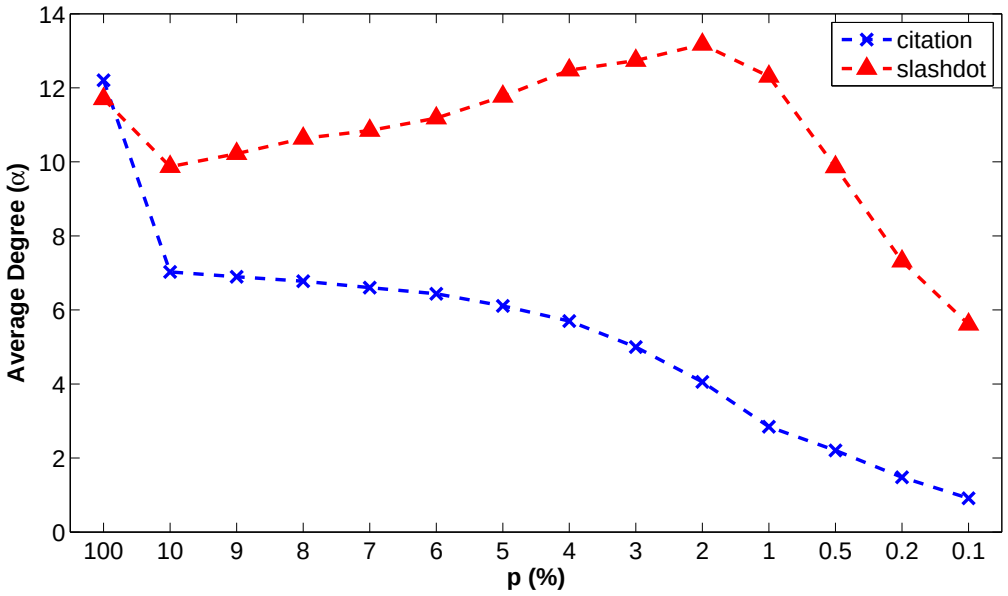


Figure 2.9: The average node degree (α) on different data sets with different p

set is about 1s.

Here, we have detected a group of blackhole patterns in this `Trading` data set. Due to the page limitation, we only show one suspicious blackhole pattern (in our opinion). In Figure 2.10, we can see that this is a blackhole pattern consists of four trading accounts, which forms a very typical trading-ring pattern. During this trading day, account A first places a big amount of sell orders in the market. A small part of them are matched with some buy orders in the market; however, the volume of sell orders highly overwhelms the volume of buy orders during that period, which makes the sell orders still remain a lot. This large volume of sell orders placed in the market makes other people feel that there is an anticipation in the market that the S&P 500 index will fall down in the near future. Therefore, these public accounts start to short the S&P 500 index at a lower price than A. Meanwhile, A's collaborators B, C, and

D, place buy orders to match with the sell orders from the public accounts, which get them a large amount of long positions at a low price. A revokes its remaining sell orders in the market, and starts to long the S&P 500 index to offset its early short position, and finally gets a long position. By doing this way, these accounts can accumulate a large amount of long positions at a low price. This is just a simple description about their collaborative strategy. In practice, it would be much more complicated. This detected blackhole pattern can be seen as an alarm for CFTC to oversee the accounts' owners future movements in both the futures contract market and the underlying stock market.

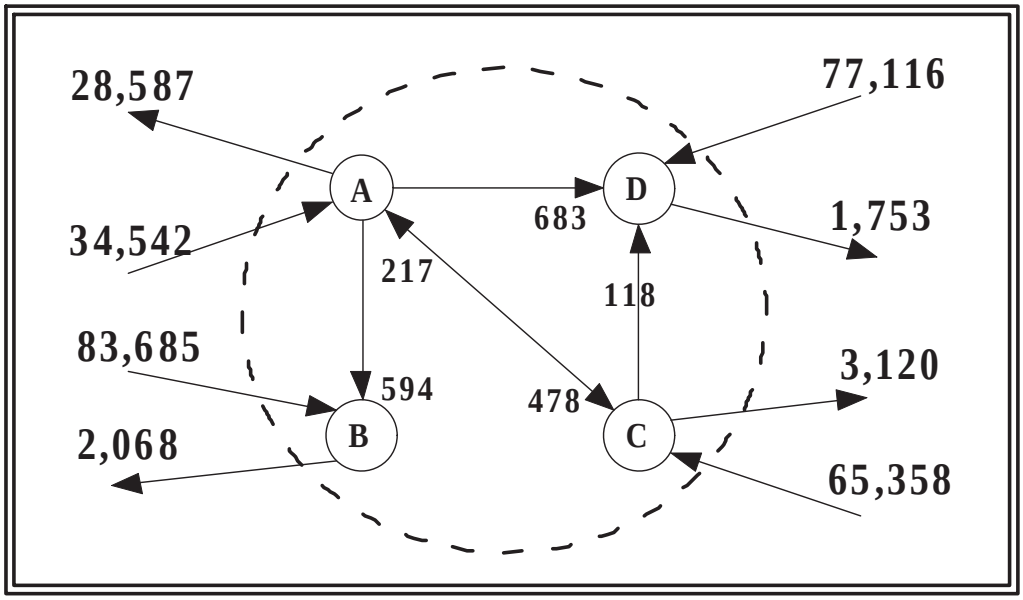


Figure 2.10: A suspicious blackhole pattern in the `Trading` data set

Clearly, there can be other explanations regarding this blackhole pattern, and our concerns can be a false alarm. However, the purpose of this blackhole mining framework is to reduce the workload of people by preliminarily filtering out a huge number of normal patterns automatically, thus people can be more focused on the

suspicious patterns that may commit frauds. Also, blackhole patterns can provide us a view of the interactions among some nodes in the network, which can help to detect the collaborative fraud activities. For example, the trading system will trigger an alarm when the trading volume on a specific account exceeds a certain threshold. However, it cannot identify the interactions among these accounts. Our blackhole patterns will not only be able to identify the interactions, but may also able to find out accounts which is under the threshold that collaborate with the suspicious accounts. However, the use of blackhole patterns is still preliminary and more comprehensive studies are expected in the future.

2.6 Related Work

This chapter is an extension of our preliminary work (Li, Xiong, Liu, & Zhou, 2010). In (Li et al., 2010), we proposed a simplified version of blackhole patterns, where the blackhole patterns have the assumption that there is no traffic flow out of the blackhole. Also, there is no weight associated with the edge. In the preliminary work, we provided a heuristic approach to solve the problem. The preliminary solution was based on a set of pattern-size-independent pruning rules and a divide-and-conquer strategy, named as the iBlackhole-DC algorithm, which was designed by exploiting the pruning rules drived from the simplified definition of blackhole patterns.

In general, other related works can be grouped into three categories. The first category includes the work on frequent subgraph mining, which studies how to efficiently find frequent subgraphs in the graph data. For instance, Jiang et al. (Jiang, Xiong, Wang, & Tan, 2009) proposed a measure for mining globally distributed frequent

subgraphs in a single labeled graph. Meanwhile, there are many works in mining frequent subgraphs in multiple labeled graphs, such as (Yan & Han, 2002), (Cook & Holder, 1994), (Huan, Wang, & Prins, 2003), (J. Wang, Hsu, Lee, & Sheng, 2006), (C. Wang, Wang, Pei, Zhu, & Shi, 2004), and (Kuramochi & Karypis, 2005). The problems of detecting blackhole and volcano patterns are different from the above works, since their definitions are different. Also, blackhole and volcano patterns may not be frequent subgraphs and vice versa.

The second category includes the works for detecting community structures in large networks. Communities in a network are groups of nodes in which connections are dense, but between which connections are sparse (Newman, 2004). There are a lot of works on how to detect communities in a network. For instance, Newman and Girvan proposed a betweenness-based method in (Newman & Girvan, 2004) and (Girvan & Newman, 2002), Hopcroft (Hopcroft, Khan, Kulis, & Selman, 2003) proposed a stable method, and Ghosh (Ghosh & Lerman, 2008) proposed a global influence based method to detect community structures. In addition, there are some other methods from a probabilistic view (Pathak, DeLong, Banerjee, & Erickson, 2008) (Zhou, Manavoglu, Li, Giles, & Zha, 2006) (Steyvers, Smyth, Rosen-Zvi, & Griffiths, 2004), rather than based on the links in the network. All the methods of detecting community structures are based on certain definitions and criteria. However, the definitions of blackhole and volcano patterns are different from the above definitions of communities. While the blackhole and volcano patterns are more focusing on small groups which satisfy some specific requirements, the community structures in a network are more vague and flexible. In terms of the scope in the network, blackhole and volcano

patterns are more localized, while community structures are more globalized.

Finally, there is a third category of research works related to this study, which is called anomaly or outlier detection (Hawkins, 1980) (Johnson & Wichern, 1998) (Barnett & Lewis, 1994). Anomaly or outlier detection aims to find out objects different from most other objects in the data. Many efforts have been made to deal with this problem and lots of approaches derived from statistics (Barnett & Lewis, 1994), machine learning (Breunig, Kriegel, Ng, & Sander, 2000) (Papadimitriou, Kitagawa, Gibbons, & Faloutsos, 2003), and data mining (Chaudhary, Szalay, & Moore, 2002) (Lazarevic & Kumar, 2005) have been developed. Most of these methods focus on traditional data types, such as vectors. Moreover, there are emerging studies on detecting outliers in graphs. Noble and Cook (Noble & Cook, 2003) studied anomalous subgraphs in general graphs using Minimum Description Length (MDL). Autopart (Chakrabarti, 2004) applied MDL to find out anomalous edges. Sun et al. (Sun, Qu, Chakrabarti, & Faloutsos, 2005) detected anomalous nodes in bipartite graphs using random walk. OutRank also employed random walk and connectivity of nodes to detect outliers (Moonesinghe & Tan, 2008). OddBall (Akoglu, McGlohon, & Faloutsos, 2010) focused on detecting anomalous nodes in weighted graphs. For all the approaches mentioned above, they mainly tackle either nodes or edges in undirected graphs. In contrast, our work focuses on detecting a well-defined anomalous structures in a directed graph.

2.7 Summary

In this chapter, we formulated the problem of mining blackhole and volcano patterns in a directed graph. These two patterns could be observed in many application scenarios, such as the trading-ring for stock market manipulation. Indeed, it was essentially a combinatorial problem for mining blackhole or volcano patterns. To reduce the complexity of the problem, we first showed that the problem of finding blackhole patterns was a dual problem of finding volcano patterns. Thus, we could be only focused on mining blackhole patterns. To that end, we proposed two pruning approaches to reduce the computational cost by decreasing both the number of combinations and the average computational cost for each combination. In the first pruning approach, we introduced the concept of combination dominance to help develop a pruning technique to reduce the exponentially growing search space. Based on this pruning technique, we developed the gBlackhole algorithm for finding top-K blackhole patterns. The second pruning approach, named the approxBlackhole algorithm, was an approximate algorithm to further decrease the computational complexity over the gBlackhole algorithm. This approxBlackhole algorithm first filtered out nodes with small diff-weights to reduce the size of the graph, and then found the top-K blackhole patterns in the subgraph induced by the rest of the nodes. There was a trade-off between the efficiency and completeness of the approxBlackhole algorithm.

Finally, experimental results on real-world data sets showed that the approxBlackhole algorithm could be several orders of magnitude faster than the gBlackhole algorithm, and both of them had a huge computational advantage over the brute-force

approach. Finally, we showed the effectiveness of this blackhole mining framework by identifying some suspicious financial fraud patterns in the simulated E-mini S&P 500 futures contract trading data from U.S. Commodity Futures Trading Commission.

CHAPTER 3

ASSESSING SIMILARITY BETWEEN SERVICE CONTRACTS FOR RISK PREDICTION

Major IT service providers typically manage a large portfolio of contracts with a variety of customers. To ensure smooth delivery and continuous profitability, it is critical for the service providers to leverage the experiences and lessons learnt from the historical contracts and prevent similar issues from reoccurring in the future. In this context, we investigate how to predict potential risks for new contracts based on their similarities with existing ones. A critical challenge along this line is to effectively measure the similarity between the contracts. To this end, extending from the Mahalanobis distance metric learning framework, we develop a new approach to gauge contract similarity using expert assessment data collected prior to contract signing (so called "contract fingerprints"). A key advantage of the proposed method is the ability to train model with not only continuous distance measures between contract pairs, but also the binary side information of dissimilar pairs. Finally, experimental results on real-world service contract data show that our proposed approach greatly outperforms existing benchmarks, and can provide more accurate contract risk assessment.

3.1 Introduction

The advance of IT services enables enterprises to offload the management of IT systems and processes to specialized service providers, so that they can focus on their core business. Due to the complexity of IT systems and the variety of customer environment, service contracts that need to be fulfilled by the provider can vary significantly. Despite such variances, the experiences of service providers have shown that successful (or unsuccessful) contracts do share common characteristics, and the root causes of many issues experienced in contract execution can be traced back to the risks identifiable prior to contract signing. To ensure smooth delivery and continuous profitability, it is critical for service providers to leverage the experiences and lessons learnt from the historical contracts to prevent similar issues from reoccurring in the future.

To this end, we study the problem of identifying similar contracts and predicting risks for new contracts to improve service provider's ability of risk management. Figure 3.1 shows a typical risk management process used by a service provider. Once received the request for proposal from a potential customer, service provider will conduct pre-bid consulting to draft a solution proposal. Then, as the solution being proposed to the customer, risk managers of the provider will assess both the technical and business aspects of the new contract, during which risks may be identified and the corresponding risk mitigation steps conducted (through e.g., modification of solution, negotiation on service level agreements and price). This is the phase during which risk managers look for the déjà vu moments that can help identify similar risks

that the provider has experienced in historical contracts. The risk identification and mitigation steps will be iterated as the contract negotiation continues, until the final contract is agreed upon and signed. After contract signing, the provider will conduct periodic project management reviews, to identify and address issues in contract delivery. Traditionally, the risk assessment method employed is largely qualitative, making it hard to leverage the lessons learnt from historical contracts, and come up with adequate risk mitigation recommendations. We argue that a more quantitative approach to predict the outcome of a contract, as well as its potential risks, is of great value to the provider.

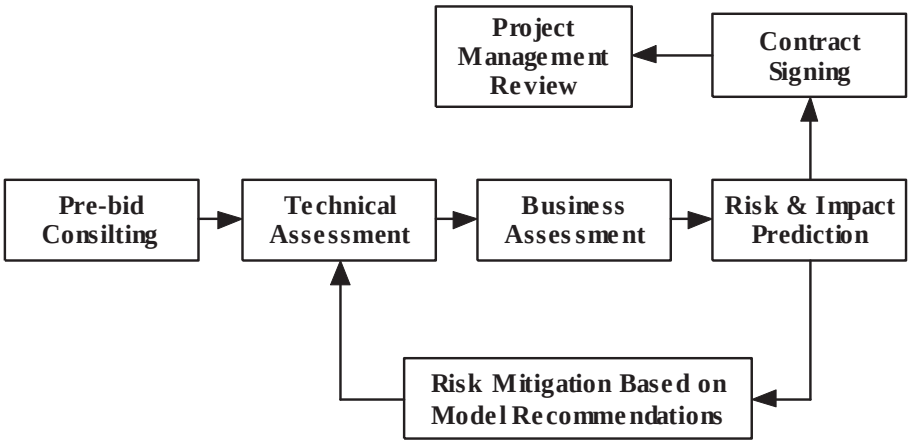


Figure 3.1: A typical contract risk management process

The input data that can be used for prediction are expert assessments on the contract prior to contract signing. In our case, these are structured questionnaire data which cover various aspects of technical and business details of the contracts. Answers to all m questions are scalar. Thus, each contract can be represented by a vector of m attributes. The actual similarity (or distance) between contracts can be evaluated in two ways. For contracts that were in trouble (i.e., under-performing

against the original financial target), root causes were typically conducted as part of the project management review. We measure the similarity between a pair of such contracts by the extent to which their identified root causes overlap. Between contracts that are healthy, we do not measure their similarities. Between a healthy contract and a troubled contract, we deem them dissimilar. In this context, our goal is to leverage the above training data to predict for a new contract, given only its pre-contract assessment data, the potential outcome (i.e., healthy vs. troubled), and the likely root causes (hence the corresponding risks) it is exposed to if troubled.

Existing classification methods, such as k-nearest neighbor (kNN), can be used to partially solve our problem. However, the common challenge of applying such methods is to define the right metric that can be used to gauge similarity (or distance) between contracts. Our problem shares some common characteristics with some of the known distance metric learning problems: given a set of points and their pairwise similarity or dissimilarity constraints as the input, it aims to learn a Mahalanobis distance metric. However, existing methods only deal with binary distance measures (i.e., pairs of points that are either similar or dissimilar), while we need to deal with the unique training data that include both continuous measurements associated with each pair of similar contracts (e.g., both financially in trouble), and data regarding dissimilar pairs (e.g., a healthy contract vs. a troubled one). In this chapter, we formulate a constrained optimization problem, with the objective of learning a Mahalanobis distance metric, such that the learnt distance for each pair of similar points is close to their actual distance, while each pair of dissimilar points can be well separated. In addition, the distance metric needs to be positive semi-definite in order to

ensure it satisfies non-negativity and triangle inequality.

To solve the problem, we first show that this optimization problem is convex, which ensures any local minimum solution is the global minimum. Then, we introduce the algorithm of Gradient Descent alternating Iterative Projection (GDIP) to solve the problem in an indirect but efficient way, since the widely used Newton's method becomes computationally prohibitive due to the quadratically increased parameters of the learnt metric. The proposed algorithm consists of two steps. In the first step, we use the gradient descent method to optimize the objective function. In the second step, we use iterative projections to satisfy the constraints. Finally, with extensive experiments on real-world contract data, we show that the prediction performance of using distance metric identified by our method outperforms the performances of using existing ones by a significant margin.

3.2 Background

In this section, we introduce the data set that is used for contract risk prediction in this chapter and the measurement that is employed to define similarity between contracts.

3.2.1 Contract Fingerprint and Performance Data

We collected the expert assessment data from the risk management process as depicted in Figure 3.1. During the contract engagement process, three separate expert assessments are conducted, $\mathcal{T}$, $\mathcal{P}$, and $\mathcal{C}$, with $\mathcal{T}$ focusing on evaluating the technical risks involved in the solution design, $\mathcal{P}$ and $\mathcal{C}$ focusing on evaluating the business risks related to business proposals and contracts. All three assessments are conducted in

the form of questionnaire: $\mathcal{T}$ has 226 questions, $\mathcal{P}$ has 82 questions, and $\mathcal{C}$ has 26 questions. Together, these 334 questions compose the features, or *finger prints*, of a service contract. The values of all these features are integers between 0 and 10 after normalization, and each contract can be represented by a vector of 334 attributes.

All historical contracts are labeled into two classes, *healthy* contracts that have met their financial targets and *troubled* contracts that have missed their financial targets. For each troubled contract, the project management team conducts root cause analysis after the financial outcome is reported. And as a result, a troubled contract will be assigned with several root causes selected from a predefined set. These root causes are developed by experts through years of experience to cover most of the common problems encountered during contract execution. If a contract is healthy, then no root cause analysis is conducted. Table 3.1 gives an illustrative example of the data set we just described. The data set can be viewed as a matrix. Each row represents a feature in contract fingerprint data and each column represents a service contract. The features $\mathcal{T}.i$, $\mathcal{P}.i$, and $\mathcal{C}.i$ correspond to the questions from assessments $\mathcal{T}$, $\mathcal{P}$, and $\mathcal{C}$, respectively. Note that if a feature has the value of "N/A", it means the answer to the corresponding question is missing or not available. The label of a contract, 1 or -1, indicates whether a contract is troubled or healthy. The value in each root cause row ($RC.i$) indicates whether the root cause is applicable (i.e., with value "1") to a troubled contract or not (i.e., with value "0").

Table 3.1: Data set sample for illustration purpose

ID	1	2	3	4	5
$\mathcal{T}.1$	2	3	2	1	3
$\mathcal{T}.2$	1	*N/A*	*N/A*	*N/A*	*N/A*
$\mathcal{P}.1$	*N/A*	*N/A*	2	*N/A*	2
$\mathcal{P}.2$	2	3	4	*N/A*	1
$\mathcal{C}.1$	1	6	1	*N/A*	4
$\mathcal{C}.2$	3	*N/A*	7	*N/A*	*N/A*
Label	1	1	1	1	-1
$RC.1$	0	1	0	0	*N/A*
$RC.2$	1	0	0	1	*N/A*

3.2.2 Contract Distance Measurement

We denote H as the set of healthy contracts, and T as the set of troubled contracts. Based on the label information, we obtain the sets of pairwise equivalence constraints (S) and inequivalence constraints (D). We also define the following distance measures for each pair of contracts in S.

Specifically, for each pair of service contracts $\mathbf{x}_i$ and $\mathbf{x}_j$:

(1) If $\mathbf{x}_i \in T$ and $\mathbf{x}_j \in T$, assign $(\mathbf{x}_i, \mathbf{x}_j)$ into S. In addition, apply the Jaccard dissimilarity index (Tan et al., 2005), which is a widely used metric measuring distance between sets, to define the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ as shown in Eq. (3.1). Here, $RC(\mathbf{x}_i)$ and $RC(\mathbf{x}_j)$ are the sets of root causes of contract $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively. In other words, the more root causes two contracts share in common, the closer is the

distance between them.

$$d_{ij} = d_J(\mathbf{x}_i, \mathbf{x}_j) = 1 - \frac{|RC(\mathbf{x}_i) \cap RC(\mathbf{x}_j)|}{|RC(\mathbf{x}_i) \cup RC(\mathbf{x}_j)|} \tag{3.1}$$

(2) If $\mathbf{x}_i \in H$ and $\mathbf{x}_j \in T$, or vice versa, assign $(\mathbf{x}_i, \mathbf{x}_j)$ into D. We label any troubled contract and any healthy contract as dissimilar in order to separate the two classes from each other.

(3) If $\mathbf{x}_i \in H$ and $\mathbf{x}_j \in H$, no constraints are applied, as there is no root cause or other side information available to determine the distance between them.

3.3 Contract Risk Prediction

Given the data set described in last section, we would like to predict (1) whether a new contract is more likely to be a healthy or troubled one; and if a troubled one, (2) what are the most likely root causes for the trouble. We adopt the well-known k-nearest neighbor (kNN) approach for both predictions. Namely, for a new contract $\mathbf{x}$ and a given distance metric, we identify its k-nearest neighbors, $k\text{NN}(\mathbf{x})$. Then we predict the label of $\mathbf{x}$ by the "voting" of its kNNs as shown in Eq. (3.2):

$$l(\mathbf{x}) = sgn(\sum_{i=1}^{k} w_i \cdot l(k\text{NN}(\mathbf{x})_i)) \tag{3.2}$$

where $l(\mathbf{x})$ is the label of $\mathbf{x}$ (i.e., 1 for troubled contract and -1 for healthy contract), sgn is the sign function, $k\text{NN}(\mathbf{x})_i$ is $\mathbf{x}$'s ith nearest neighbor, and $w_i = 1/log_2(1+i)$ is the voting weight of the ith nearest neighbor. Note that $k\text{NN}(\mathbf{x})$ are sorted in the ascending order of their distance to $\mathbf{x}$, i.e., $k\text{NN}(\mathbf{x})_1$ is $\mathbf{x}$'s most nearest neighbor, while $k\text{NN}(\mathbf{x})_k$ has the furthest distance to $\mathbf{x}$ among its kNNs.

If a contract $\mathbf{x}$ is predicted as troubled, we further predict its potential root causes. Let R be the set of all predefined root causes. For each root cause $rc \in R$, we define the vote of rc as follows:

$$v(rc) = \sum_{i=1}^{k} w_i \cdot \mathbf{1}\{rc \in RC(k\text{NN}(\mathbf{x})_i)\} \tag{3.3}$$

where $RC(\mathbf{x})$ is the set of root causes of $\mathbf{x}$, $k\text{NN}(\mathbf{x})_i$ is $\mathbf{x}$'s ith nearest neighbor, $w_i = 1/log_2(1+i)$ is the voting weight of the ith nearest neighbor, and $\mathbf{1}\{\cdot\}$ is the indicator function such that $\mathbf{1}\{\text{True}\} = 1$ and $\mathbf{1}\{\text{False}\} = 0$. We then select the root causes with top-t highest votes as the predicted root causes of $\mathbf{x}$.

3.4 Problem Formulation

Given the prediction approach described in last section, the key problem is to define an appropriate distance metric to measure the similarity between contracts. Suppose there are a set of contracts $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where each $\mathbf{x}_i \in \mathbb{R}^m$ is a vector of m numerical features. For training data, we are given the sets of pairwise equivalence and inequivalence constraints: $S = \{(\mathbf{x}_i, \mathbf{x}_j) \,|\, \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are similar}\}$ and $D = \{(\mathbf{x}_i, \mathbf{x}_j) \,|\, \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are dissimilar}\}$. We are also given a continuous distance measurement $d_{ij} \in (0, 1)$ between each pair of points $(\mathbf{x}_i, \mathbf{x}_j) \in S$. The distance metric we seek to define is basically a Mahalanobis distance: $d_A(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||_A = ((\mathbf{x}_i - \mathbf{x}_j)^T A(\mathbf{x}_i - \mathbf{x}_j))^{1/2} = \langle A, (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \rangle_F^{1/2}$, where A is a $m \times m$ real symmetric matrix, and $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product of two matrices with the same size, such that $\langle A, B \rangle_F = \sum_i \sum_j A_{ij} B_{ij}$. A specifies a family of Mahalanobis distance metrics over $\mathbb{R}^m$, and learning such a distance metric is equivalent to first introducing a linear transformation that rescales each point $\mathbf{x}_i$ to $A^{1/2}\mathbf{x}_i$ and then

applying the standard Euclidean metric to the rescaled data. Specifically, setting $A = I$ gives the Euclidean distance. Note that our problem setting is different from the traditional distance metric learning (Xing, Ng, Jordan, & Russell, 2002), which only considers the binary input (i.e., similar or dissimilar pairs).

Our objective is to find out a full ranked Mahalanobis distance metric A, such that the learnt distance matches with the given distance as much as possible, while satisfying the additional constraints of similar and dissimilar pairs. Furthermore, in order to ensure A is a distance metric which satisfies non-negativity and triangle inequality, we require A to be positive semi-definite, i.e., $A \succeq 0$.

We formulate our distance metric learning problem as the following constrained optimization problem:

$$\min_{A} \quad SSE(A) = \frac{1}{2} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in S} (||\mathbf{x}_i - \mathbf{x}_j||_A^2 - d_{ij}^2)^2 \tag{3.4}$$

$$s.t. \qquad ||\mathbf{x}_i - \mathbf{x}_j||_A^2 \geq 1, \forall (\mathbf{x}_i, \mathbf{x}_j) \in D \tag{3.5}$$

$$A \succeq 0 \tag{3.6}$$

We choose the objective function that minimizes the sum of squared errors between the learnt Mahalanobis distance and the actual distance for each pair of contracts in S. Note that we use the term $(||\mathbf{x}_i - \mathbf{x}_j||_A^2 - d_{ij}^2)^2$ rather than its widely used form (Xing et al., 2002) of $(||\mathbf{x}_i - \mathbf{x}_j||_A - d_{ij})^2$ in Eq. (3.4) in order to simplify the form of its first-order derivative, which will be further discussed in the next section. However, this change still keeps the same monotonicity as its original term. In order to separate pairs of dissimilar contracts, we simply let the distance for each pair of contracts in

D be greater than 1, which is the largest possible distance between a pair of similar contracts.

3.5 Algorithm Design

The optimization problem formulated in last section has an objective function that is linear in parameter A. In addition, constraints (3.5) and (3.6) can be easily verified as convex. Thus, the problem is a convex optimization problem, which implies that any local minimum solution is the global minimum. Therefore, we can derive an efficient local-minima-free algorithm to solve the problem.

Let $\mathbf{z}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. Then $SSE(A) = \frac{1}{2}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S}(\mathbf{z}_{ij}^T A\mathbf{z}_{ij} - d_{ij}^2)^2$. In order to find out the local minimum, we set SSE's first order derivative to be $\mathbf{0}$:

$$\begin{aligned}\frac{d}{dA}SSE &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S}(\mathbf{z}_{ij}^T A\mathbf{z}_{ij} - d_{ij}^2)\cdot\frac{d(\mathbf{z}_{ij}^T A\mathbf{z}_{ij} - d_{ij}^2)}{dA}\\ &= \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S}(\mathbf{z}_{ij}^T A\mathbf{z}_{ij} - d_{ij}^2)\mathbf{z}_{ij}\mathbf{z}_{ij}^T = \mathbf{0}\end{aligned}$$

The generally used Newton's method can become computationally prohibited when the size of A is large, due to its quadratically increased parameters. It requires $O(n^6)$ time to invert the Hessian over n^2 parameters (Xing et al., 2002). In addition, constraints (3.5) and (3.6) make it even harder to enforce. To handle the computational challenge, we introduce Gradient Descent alternating Iterative Projection (GDIP), an indirect but efficient approach to solve this optimization problem. Figure 3.2 shows the pseudo code of the algorithm. Here, $||\cdot||_F$ is the Frobenius norm of a matrix, i.e., $||A||_F = \langle A, A\rangle_F^{1/2}$.

The algorithm consists of two iterative steps. In the first step, we use the gradient

1. **Iterate**
2. $A := A - \gamma \nabla SSE(A)$
3. **Iterate**
4. $A := argmin_{A'}\{||A' - A||_F : A' \succeq 0\}$
5. **foreach** $(\mathbf{x}_i, \mathbf{x}_j) \in D$
6. $A := argmin_{A'}\{||A' - A||_F : ||\mathbf{x}_i - \mathbf{x}_j||^2_{A'} \geq 1\}$
7. **end for**
8. **until** A converges
9. **until** convergence

Figure 3.2: Gradient descent alternating iterative projection algorithm

descent method to optimize the objective function SSE (line 2 in Figure 3.2). In the second step, we apply the method of iterative projections to ensure constraints (3.5) and (3.6) hold (line 3-8 in Figure 3.2). Both steps can be done inexpensively as we discuss next.

3.5.1 Gradient Descent

The gradient descent method is a first-order optimization algorithm. Starting from a random point, it takes steps proportional to the negative of the gradient of the function (i.e., first order derivative) at the current point, and iteratively approaches a local

minimum (Boyd & Vandenberghe, 2004). Assume that a multivariable function $F(\mathbf{x})$ is defined and is differentiable in a neighborhood of a point $\mathbf{a}$, $F(\mathbf{x})$ decreases fastest if one goes from $\mathbf{a}$ in the direction of the negative gradient of F at $\mathbf{a}$, i.e., $-\nabla F(\mathbf{a})$. In other words, let $\mathbf{b} = \mathbf{a} - \gamma \nabla F(\mathbf{a})$, where γ is the step size and usually a small enough number, then $F(\mathbf{a}) \geq F(\mathbf{b})$. Starting from a random guess $\mathbf{x}_0$ for a local minimum of F, one can calculate the sequence $\{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots\}$ where $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \nabla F(\mathbf{x}_t), t \geq 0$. Obviously, $F(\mathbf{x}_0) \geq F(\mathbf{x}_1) \geq F(\mathbf{x}_2) \geq \cdots$. With certain assumptions on the function F (e.g., F is a convex function) and particular choices of γ (e.g., chosen via a line search method that satisfies the Wolfe conditions), it is guaranteed that the sequence will converge to a local minimum (Boyd & Vandenberghe, 2004). In practice, the sequence is considered as converged (i.e., the stopping criterion) when $||\nabla F(\mathbf{x})|| \leq \eta$, where η is a small and positive number.

In our case, we set $A_0 = I$, and iteratively let $A_{t+1} = A_t - \gamma_t \nabla SSE(A_t)$. Note that γ can be changed at every iteration t, which will be discussed in the next subsection. When $||\nabla SSE(A_t)|| \leq 1$, we consider A_t as converged and the iteration procedure stops. Since the local minimum is also the global minimum, the gradient descent method can lead to the minimum value of the objective function.

3.5.2 Backtracking Line Search

The gradient descent method helps to determine a descent direction $-\nabla F(\mathbf{x})$, along which the objective function $F(\mathbf{x})$ will be reduced at point $\mathbf{x}$. We also need to determine how far $\mathbf{x}$ should move along the descent direction (i.e., γ), which can be solved using the widely used line search method.

In our approach, we adopt a simple yet effective method called backtracking line search (Boyd & Vandenberghe, 2004). This method requires defining parameters $0 < \alpha < 1$ and $0 < \beta < 1$. It starts with unit step size and then reduces it by factor β until the stopping condition $F(\mathbf{x} + \gamma\Delta\mathbf{x}) \leq F(\mathbf{x}) + \alpha\gamma\nabla F(\mathbf{x})^T\Delta\mathbf{x}$ (i.e., the Armijo rule) is met. Figure 3.3 shows the customized backtracking line search method to determine the step size γ_t for the tth iteration of the gradient descent.

1. $\gamma_t := 1$
2. **while** $SSE(A_t - \gamma_t \nabla SSE(A_t)) >$
   $SSE(A_t) - \alpha\gamma_t \nabla SSE(A_t)^T \nabla SSE(A_t)$
3. $\gamma_t := \beta\gamma_t$
4. **end while**

Figure 3.3: Backtracking line search algorithm

3.5.3 Iterative Projection

The first step of the tth iteration (i.e., gradient descent) gives us the current optimal matrix A_t. For the second step, we identify the best possible substituting matrix A'_t of A_t, such that it is the closest to A_t in the Frobenius norm, among all the matrices that satisfy constraints (3.5) and (3.6). This can be achieved by repeatedly projecting A_t onto the spaces of $C_1 = \{A'_t : A'_t \succeq 0\}$ and $C_2 = \{A'_t : ||\mathbf{x}_i - \mathbf{x}_j||^2_{A'_t} \geq 1, \forall(\mathbf{x}_i, \mathbf{x}_j) \in D\}$

until A_t converges.

The first projection step onto C_1 can be done by eigen-decomposition (line 4 in Figure 3.2). We first factorize A_t into its canonical form: $A_t = Q\Lambda Q^T$, where Q is an orthogonal matrix (the columns of which are eigenvectors of A_t), and Λ is a real and diagonal matrix (having the corresponding eigenvalues of A_t on the diagonal). We replace Λ's negative diagonal values with 0 and denote it as $\tilde{\Lambda}$, i.e., $\tilde{\Lambda} = max(\Lambda, \mathbf{0})$, and let $A'_t = Q\tilde{\Lambda}Q^T$. In this way, we get a matrix A'_t which is positive semi-definite and has the minimum Frobenius distance to A_t. By replacing A_t with A'_t (i.e., $A_t := A'_t$), we have the projection of A_t onto C_1.

Constraint (3.5) consists of a sequence of single linear constraints; therefore, projecting A_t onto C_2 can be done by a series of single projections, where each single projection is associated with one linear constraint (line 5-7 in Figure 3.2). For each $(\mathbf{x}_i, \mathbf{x}_j) \in D$, we project A_t onto the space $\{A'_t : ||\mathbf{x}_i - \mathbf{x}_j||^2_{A'_t} \geq 1\}$ (line 6 in Figure 3.2). This projection can be utilized by formulating into another optimization problem as shown in Eq. (3.7), which finds out a substitution matrix A'_t that is the closest to A_t in Frobenius form and satisfies the constraint of $||\mathbf{x}_i - \mathbf{x}_j||^2_{A'_t} \geq 1$. In Eq. (3.7), we denote the column vectors after the vectorization of matrix A'_t, A_t and $\mathbf{z}_{ij}\mathbf{z}_{ij}^T$ ($\mathbf{z}_{ij} = \mathbf{x}_i - \mathbf{x}_j$) as $\mathbf{a}'_t$, $\mathbf{a}_t$ and $\mathbf{b}$, respectively. Note that $||\mathbf{x}_i - \mathbf{x}_j||^2_{A'_t} = \langle A'_t, \mathbf{z}_{ij}\mathbf{z}_{ij}^T \rangle_F = \mathbf{a}'^T_t \mathbf{b}$, and the column vector after the vectorization of matrix $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ is $(1\ 4\ 2\ 5\ 3\ 6)^T$.

$$\begin{aligned} \min_{\mathbf{a}'_t} \quad & err(\mathbf{a}'_t) = \frac{1}{2}||\mathbf{a}'_t - \mathbf{a}_t||^2 \\ s.t. \quad & \mathbf{a}'^T_t \mathbf{b} \geq 1 \end{aligned} \tag{3.7}$$

Let $\lambda \geq 0$ be the Lagrange multiplier associated with the constraint, we have the Lagrangian of the objective function as follows:

$$L(\mathbf{a}'_t, \lambda) = \frac{1}{2}||\mathbf{a}'_t - \mathbf{a}_t||^2 - \lambda(\mathbf{a}_t'^T\mathbf{b} - 1) \tag{3.8}$$

Apply the Karush-Kuhn-Tucker (KKT) conditions (Kuhn & Tucker, 1951)(Karush, 1939) to Eq. (3.8), we have the following equations:

$$\frac{\partial L}{\partial \mathbf{a}'_t} = \mathbf{a}'_t - \mathbf{a}_t - \lambda\mathbf{b} = \mathbf{0} \tag{3.9}$$

$$\mathbf{a}_t'^T\mathbf{b} - 1 \geq 0 \tag{3.10}$$

$$\lambda \geq 0 \tag{3.11}$$

$$\lambda(\mathbf{a}_t'^T\mathbf{b} - 1) = 0 \tag{3.12}$$

We start solving from Eq. (3.12). If $\lambda = 0$, from Eq. (3.9) we have $\mathbf{a}'_t = \mathbf{a}_t$. In this case, constraint (3.10) becomes $\mathbf{a}_t^T\mathbf{b} \geq 1$.

If $\lambda \neq 0$, then $\mathbf{a}_t'^T\mathbf{b} - 1 = 0$. Multiply Eq. (3.9) by $\mathbf{b}^T$ from the left-hand side, we will get

$$\mathbf{b}^T\mathbf{a}'_t - \mathbf{b}^T\mathbf{a}_t - \lambda\mathbf{b}^T\mathbf{b} = \mathbf{0} \tag{3.13}$$

Since $\mathbf{a}_t'^T\mathbf{b}$ is a real number, we have $\mathbf{b}^T\mathbf{a}'_t = \mathbf{a}_t'^T\mathbf{b} = 1$. Substituting the corresponding part in Eq. (3.13), we get

$$\lambda = \frac{1 - \mathbf{b}^T\mathbf{a}_t}{\mathbf{b}^T\mathbf{b}} = \frac{-\mathbf{a}_t^T\mathbf{b} + 1}{||\mathbf{b}||^2} \tag{3.14}$$

Combining Eq. (3.9) and (3.14), the close form of $\mathbf{a}'_t$ can be written as

$$\mathbf{a}'_t = \mathbf{a}_t + \lambda\mathbf{b} = \mathbf{a}_t + \frac{-\mathbf{a}_t^T\mathbf{b} + 1}{||\mathbf{b}||^2} \cdot \mathbf{b} \tag{3.15}$$

In this case, constraint (3.11) becomes $\mathbf{a}_t^T\mathbf{b} \leq 1$.

Finally, by combining the above two cases and converting the column vector $\mathbf{a}'_t$ back to matrix A'_t, we can have the expression of A'_t when projecting A_t onto the space $\{A'_t : ||\mathbf{x}_i - \mathbf{x}_j||^2_{A'_t} \geq 1, \textit{where } (\mathbf{x}_i, \mathbf{x}_j) \in D\}$ as shown in Eq. (3.16). The projection of A_t onto C_2 can be achieved by performing a series of single projections and letting A'_t be the new value of A_t at the end of each projection.

$$A_t := A'_t = \begin{cases} A_t, \text{ if } \langle A_t, \mathbf{z}_{ij}\mathbf{z}_{ij}^T\rangle_F \geq 1 \\ A_t + \dfrac{1 - \langle A_t, \mathbf{z}_{ij}\mathbf{z}_{ij}^T\rangle_F}{||\mathbf{z}_{ij}\mathbf{z}_{ij}^T||_F^2} \cdot \mathbf{z}_{ij}\mathbf{z}_{ij}^T, \text{ otherwise} \end{cases} \tag{3.16}$$

Combining the two projection methods discussed above, our method ensures the convergence of A within constraints (3.5) and (3.6).

3.6 Experimental Results

In this section, we provide the experimental results on real-world service contract data obtained from a major IT service provider. We evaluate the overall performance of our proposed method. In particular, we compare our distance metric learning method with the existing methods.

3.6.1 Data Cleaning and Preprocessing

The experiments are conducted on a data set of 470 services contracts. As introduced in Section 3.2, each contract has 334 features, the *healthy/troubled* label, and the identified root causes, if it is a troubled one. However, this data set contains a large portion of missing feature values - 278 out of 334 features have missing values great than 50%, and 321 out of 470 contracts have missing feature values greater than 50%.

We eliminate the data objects or features with missing values more than 50% from the raw data, since measuring distance between contracts with that amount of missing data could lead to inaccurate and misleading results. Finally, the trimmed data set has 149 contracts, among which 43 are troubled contracts and 106 are healthy ones, with each contract having 56 features. There is still a small portion of missing feature values (about 2%) in the data set after cleaning. We estimate the value of missing entries as the mode of the corresponding features.

3.6.2 Baseline Algorithm for Comparison

We use Xing's method (Xing et al., 2002) as the baseline algorithm, since it not only is the most representative approach in distance metric learning, but also has a problem setting similar to ours. Xing formulated the problem of learning a fully ranked Mahalanobis distance metric as a constrained convex optimization problem, based on the pairwise equivalence constraints and inequivalence constraints. The specification of the problem formulation is shown in Eq. (3.17), where $\mathbf{x}_i$ and $\mathbf{x}_j$ are m-dimensional points, S is the set of pairwise equivalence constraints, D is the set of pairwise inequivalence constraints, and A is the Mahalanobis distance metric.

$$\begin{aligned} \min_{A} \quad & SSE(A) = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} ||\mathbf{x}_i - \mathbf{x}_j||_A^2 \\ s.t. \quad & \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in D} ||\mathbf{x}_i - \mathbf{x}_j||_A \geq 1 \\ & A \succeq 0 \end{aligned} \tag{3.17}$$

We obtain the sets of pairwise equivalence or inequivalence constraints used in the baseline method from the side information of the data. Specifically, for each pair of

service contracts $\mathbf{x}_i$ and $\mathbf{x}_j$:

(1) If $\mathbf{x}_i \in T$ and $\mathbf{x}_j \in T$, compute their Jaccard distance d_{ij} from Eq. (3.1). If $d_{ij} \leq \theta$, assign $(\mathbf{x}_i, \mathbf{x}_j)$ into S, otherwise, assign $(\mathbf{x}_i, \mathbf{x}_j)$ into D. $\theta \in [0, 1]$ is a user specified threshold parameter, and when the pairwise Jaccard distance of two points is below the threshold, we consider them as a similar pair, otherwise, a dissimilar pair.

(2) If $\mathbf{x}_i \in H$ and $\mathbf{x}_j \in T$, or vice versa, assign $(\mathbf{x}_i, \mathbf{x}_j)$ into D. In other words, troubled contract and healthy contract must be separated away from each other.

(3) If $\mathbf{x}_i \in H$ and $\mathbf{x}_j \in H$, simply ignore this case.

3.6.3 Performance of Contract Classification

We first evaluate the performance of classifying a new contract as healthy or troubled, when different distance metrics are used. We use leave-one-out cross-validation (Tan et al., 2005), which involves using only one data point from the data set as the validation (testing) data, and the remaining data points as the training data. This procedure is repeated n times such that each data point in the data set is used exactly once as the validation data, where n is the size of the data set.

In each iteration of the cross-validation, one data point $\mathbf{x}$ is chosen as the validation data. The other $n-1$ data points are used as the training data to learn a full ranked Mahalanobis distance metric A via our GDIP algorithm or the baseline method. We can then find out $\mathbf{x}$'s k-nearest neighbors (kNN) under distance metric A, and predict the label of $\mathbf{x}$ by the voting of its kNN as shown in Eq. (3.2).

Combining the predicted labels of all n data points with their actual labels, we can

Table 3.2: Confusion matrix of the best classification result

		Predicted Class	
		+	−
Actual Class	+	43 (TP)	0 (FN)
	−	22 (FP)	84 (TN)

get the confusion matrix that summarizes the number of instances predicted correctly or incorrectly by a classification model (e.g., Table 3.2). For binary classification, the rare class is often denoted as the positive class, while the majority class is denoted as the negative class, based on which we can have the definition of *False/True Positives* (FP/TP) and *False/True Negatives* (FN/TN). In our case, the positive class is the set of troubled contracts while the negative class is the set of healthy contracts, since the former is much more important than the latter. Then we employ three widely used measures (Tan et al., 2005) - *precision* (p), *recall* (r), and *F-measure* (F) - that value the success of detecting the positive class more significantly than detecting the negative class, to analyze the prediction result. The definitions of these measures are as follows:

$$p = \frac{TP}{TP + FP},\ r = \frac{TP}{TP + FN},\ F = \frac{2rp}{r + p}$$

For the baseline Xing's method, we apply different values of threshold parameter (θ) and get the corresponding confusion matrices and F-measures. As an additional baseline, we also calculate the F-measure of simply using the Euclidean distance metric. By varying the value of k, we compare the results of all these baseline methods with our method in Figure 3.4. Note that in the figure, *Xing-0.8* represents Xing's

method with $\theta = 0.8$.

As shown in Figure 3.4, classification result based on distance metric learnt from our GDIP algorithm generally outperforms the results of the other methods, and our method has the better performance when $k \leq 5$, and the best performance at $k = 3$. The corresponding confusion matrix of $k = 3$ is shown in Table 3.2. The recall rate is 100%, which none of the other methods can achieve. This means that using our method, all troubled contracts can be predicted correctly, and there is no troubled contract misclassified as healthy. In the meantime, precision rate is 66%, which means one third of the contracts that the classifier predicts as troubled are actually healthy. Overall, this is the result we desire in practice, since the potential loss of misclassifying a troubled contract is much bigger than misclassifying a healthy contract.

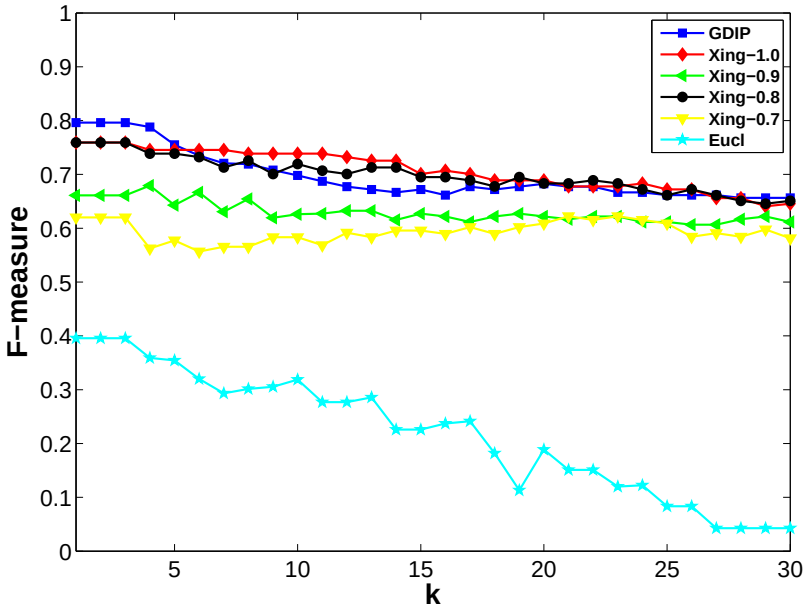


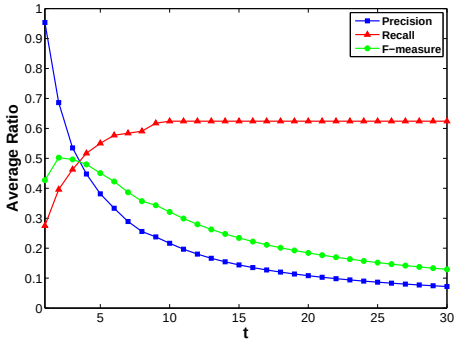
Figure 3.4: Classification performances using different distance metrics

3.6.4 Performance of Root Cause Prediction

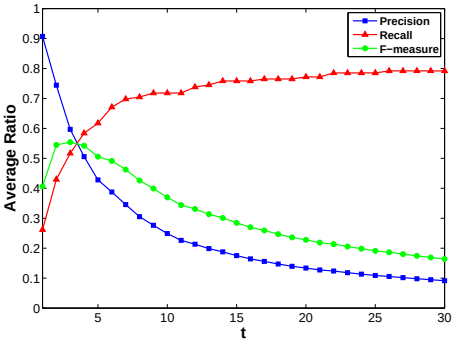
Next, we evaluate the performance of predicting the root causes of a troubled contract, when different distance metrics are used. We again use the leave-one-out cross-validation method in this evaluation. In each iteration, one troubled contract $\mathbf{x}$ is chosen for validation, and the rest $n-1$ contracts are used as the training data to learn a distance metric A. Under metric A, we can obtain $\mathbf{x}$'s k-nearest neighbors, and calculate the vote of each root cause as shown in Eq. (3.3). The predicted root causes of $\mathbf{x}$ are the ones with top-t highest votes. We denote $\mathbf{x}$'s predicted root causes as $RC_p(\mathbf{x})$. By comparing with $\mathbf{x}$'s actual root causes $RC(\mathbf{x})$, we can calculate the *precision* (p), *recall* (r), and *F-measure* (F) when predicting $\mathbf{x}$'s root causes as follows:

$$p = \frac{|RC(\mathbf{x}) \cap RC_p(\mathbf{x})|}{|RC_p(\mathbf{x})|},\ r = \frac{|RC(\mathbf{x}) \cap RC_p(\mathbf{x})|}{|RC(\mathbf{x})|},\ F = \frac{2rp}{r+p}$$

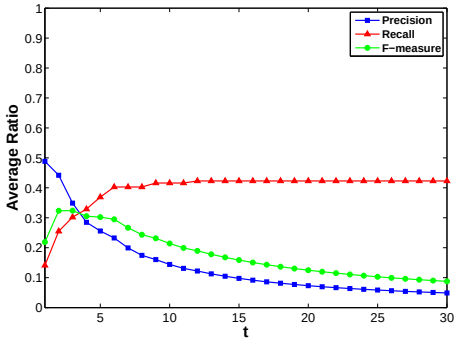
At the end of the cross-validation process, we calculate the averaged precision, recall, and F-measure of predicting root causes of all troubled contracts. We then compare the performance of our method with the baseline algorithms. For Xing's method, we apply different values of threshold parameter (θ) and get the corresponding measures. We also use the Euclidean distance metric as an additional baseline method. By varying the values of k and t, we compare the results of all these baseline methods with our method in Figure 3.5. The figure shows a set of representative results with $k = 3$ and $k = 10$, and Xing's method with $\theta = 0.8$. The observations with other parameter settings are similar.



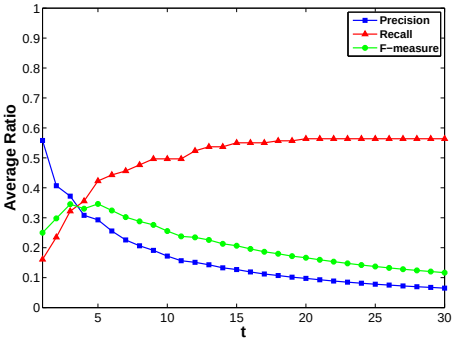
(a) GDIP, $k = 3$



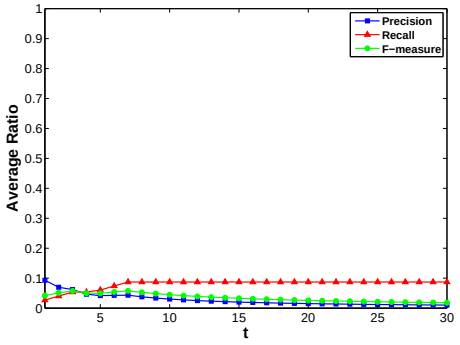
(b) GDIP, $k = 10$



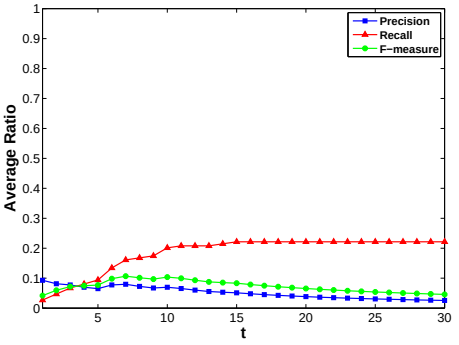
(c) Xing-0.8, $k = 3$



(d) Xing-0.8, $k = 10$



(e) Eucl, $k = 3$



(f) Eucl, $k = 10$

Figure 3.5: Root cause prediction performances of troubled contracts using different distance metrics

Figure 3.5 clearly shows that using distance metric learnt from our GDIP algorithm results in better prediction on root causes. The best performance in terms of

the F-measure are generally achieved at $k = 3$, which is consistent with the contract classification results. In addition, we can also observe the trade-off between precision and recall in the figure. As t increases, precision gradually decreases whereas recall asymptotically approaches its maximum possible value. The maximum value of the average recall is less than 1, since there exist several root causes in the data set that were only observed in few contracts. Thus, the maximum recall value of such contracts cannot be equal to 1, resulting in the maximum value of the average recall less than 1. In practice, the practitioner may prefer the model with higher recall to the one with the best F-measure, since the potential loss of missing a true root cause is much bigger than the cost of examining several falsely predicted root causes.

3.7 Related Work

While there are existing works related to using predictive models for contract risk management, many of them focus on different aspects in the risk management lifecycle. For example, Mojsilovic et al. (Mojsilović et al., 2007) use predictive models to estimate the likelihood of revenue erosion in large outsourcing engagements, while Goo et al. (Goo et al., 2007) study factors that influence the duration of IT outsourcing relationships. In addition, there are related studies on risk management in large and complex projects (Leung et al., 1998), (Deleris et al., 2007). Different from these studies, our work focuses on the unique problem of identifying similarity between historical and new contracts, and using this information to predict risks for the latter.

The problem studied in this chapter is also related to distance metric learning,

which aim to learn a distance metric for the input data space from a given collection of similar and dissimilar pairs of points (Yang & Jin, 2006), using either supervised or unsupervised learning methods. Supervised distance metric learning typically deals with data with equivalence constraints, i.e., pairs of data points in the same class, or inequivalence constraints, i.e., pairs of data points that belong to different classes. For instance, Xing et al. (Xing et al., 2002) formulated the problem as a constrained convex programming problem, and learnt a global distance metric that minimized the distance between the pairs of data points in the equivalence constraints, while separated all pairs of data points in the inequivalence constraints far apart at the same time. Later, many additional methods or extensions were proposed with varying optimization frameworks or for different target use cases (Bar-Hillel, Hertz, Shental, & Weinshall, 2003), (Kwok & Tsang, 2003), (Yang, Jin, & Sukthankar, 2007). In addition, there are some other researches focusing on local adaptive distance metric learning, whose goal is to find feature weights that are adapted to individual test examples (Domeniconi, Gunopulos, et al., 2002), (Peng, Heisterkamp, & Dai, 2002), (Zhang, Tang, & Kwok, 2005). Meanwhile, unsupervised distance metric learning does not deal with input data annotated with labels. Its goal is to learn an underlying low-dimensional manifold where geometric relationships (e.g., distance) between most of the observed data (with no label information) can be preserved. Thus, unsupervised distance metric learning is also known as the dimension reduction. Existing approaches include the linear dimension reduction algorithms, such as Principle Component Analysis (PCA) (Jolliffe, 2005) and Multidimensional Scaling (MDS) (Cox & Cox, 2000), and the nonlinear approaches, e.g., ISOMAP (Tenenbaum, De Silva, &

Langford, 2000) and Local Linear Embedding (LLE) (Saul & Roweis, 2003). Our work leverages the label information annotated with the data, thus falls into the category of supervised distance metric learning. It deals with the unique data set and application as an extension to the existing methods.

3.8 Summary

In this chapter, we investigated the problem of evaluating potential risks for new service contracts by measuring the similarity of contracts. It was motivated by the understanding that experiences and lessons learnt from historical contracts were of great value to IT service providers for risk management. A key challenge was how to define the right metric to measure contract similarity when there were continuous distance measurements between contract pairs. We formulated the problem as an extension of the existing Mahalanobis distance metric learning problem, which could only deal with binary distance measurements. We proposed the Gradient Descent alternating Iterative Projection method (GDIP) to solve the problem efficiently. Finally, experiments on real-world contract fingerprint data showed that distance metric identified through the GDIP approach could lead to better performances than existing ones in terms of both contract classification and root cause prediction.

CHAPTER 4

IDENTIFY INDICATIVE FINANCIAL MEASURES USING FINANCIAL STATEMENT OF BANKRUPT COMPANIES

Traditional auditing process and risk analysis could not match the increased demand of automatic and efficient detection of financial statement fraud. Data mining techniques are used to help address this challenge. In this chapter, clustering technique is applied to identify the clustering effect of bankrupt companies in different business sectors by using the financial statement information. The financial measures that have the most significant indicative power for future bankruptcy have also been uncovered.

4.1 Introduction

Financial fraud detection is one of the most important aspects in financial risk management. In recent years, there is an increasing trend in financial losses due to financial fraud and bankruptcy. A number of high-profile companies, such as Enron, World Corm, Xerox and Lucent, were charged with fraud by the US Security and Exchange Commission (SEC) in the past decade. Hence, how to develop an efficient and effective financial fraud detection framework draws the best interests among investors, public, researchers, auditors and regulators.

In general, the published financial statements are one of the most pervasive and

consistently available predictor of a companys future performance. The financial statements consist of balance sheet, income statement, and cash flow statement. Each statement provides the basis for understanding and evaluating the financial status of a company. However, fraudulent financial reporting can destroy the true picture of a companys financial situation. For example, by manipulating elements such as liabilities, expenses, or losses in the financial statement, the unhealthy financial status can be covered. Nowadays, financial statement fraud is one of the most notable management frauds in the U.S. due to the large financial losses it yields. Therefore, financial statement fraud detection becomes a crucial and pervasive issue in financial risk management.

In literature, there have been numerous researches conducted in the area of financial statement fraud detection. In accounting area, hundreds of financial indicators which extract significant features from financial statement have been proposed to build the predication model of high fraud-potential and high bankrupt-potential firms (Altman, 1968), (Loebbecke et al., 1989). These indicators are proposed based on empirical experience. On the other hand, in knowledge discovery area, several researchers have attempted to adapt data mining techniques to refine financial features from large financial statement datasets. Most of these works employed supervised learning techniques to solve classification tasks (Kirkos et al., 2007), (Apparao et al., 2009). While these studies support the significance of classification techniques to analysis financial statement, there has been little research in literature in the direction of applying unsupervised learning techniques to analyze the financial statement data. Moreover, majority of the existing researches only focus on a particular busi-

ness sector. For instance, bankruptcy predictions have been conducted in the railroad, banking, brokerage, education, and insurance industries (Zmijewski, 1984).

In this chapter, we would like to find the answer to an interesting question: can we use financial statement information to identify which business sector has the strongest clustering effect, such that the financial features we extracted from the past bankrupt (financial unhealthy) companies in the business sector has the most significant predictive power in the future? The question is not only useful for auditors, but also important for investors. Investors can prevent potential losses by filtering out firms with high potential bankruptcy risks based on the learnt characteristics. Auditors can use the valuable features as an additional decision aid to monitor the financial situation of the client firm.

We utilize the advantages in both the accounting and data mining areas to solve the problem. First of all, financial statement data has its unique data characteristics. We leverage the domain knowledge in accounting area to help us select valuable financial features. Then we employ clustering techniques, an unsupervised learning data mining technique of grouping similar data objects while distinguishing dissimilar ones (Tan et al., 2005), to identify business sector in which bankrupt companies has strong clustering effects. Finally the most indicative financial features for the bankrupt companies in the business sector can be uncovered from the hidden data.

Experimental results show that among 6 business sectors (Mining, Manufacturing, Transportation and Utilities, Wholesale, Retail Trade, and Services), retail trade industry has the strongest clustering effect. This result is validated by significant tests. 10 most indicative financial features have also been extracted for the bankrupt

companies in retail business.

4.2 Related Work

Related work can be grouped into two categories. In accounting literature, financial indicators which extract significant features from financial statement are a traditional way to build the predication model of high fraud-potential and high bankrupt-potential firms. In 1968, Altman published one of the most prominent early models of bankruptcy prediction - Altman Z-score Financial Analysis Tool, which is still widely applied today (Altman, 1968). Altman Z-score method predicts the potential bankruptcy of a company by the combination of five financial statement ratios. Other well-known financial indicators, such as LOGDEBT, DEBTEQ, SALGRTH, RECA, RECSAL, INVTA, and GPTA (Fanning & Cogger, 1998), (Feroz, Kwon, Pastena, & Park, 2000), (Loebbecke et al., 1989), (Persons, 2011), (Stice, Albrecht, & Brown, 1991), (Stice, 1991), have also been proposed. These indicators are usually based on empirical experience.

In data mining literature, several groups of researchers have devoted a significant amount of effort in using modern methods to study financial statement fraud detection from different perspectives. In general, previous researches focused on using classification algorithm, such as neural network, regression and decision tree. Green and Choi (Green & Choi, 1997) presented a neural network fraud classification model by using five ratios and three accounts as input. The results showed that neural network had significant capabilities when used as a fraud detection tool. Fanning and Cogger (Fanning & Cogger, 1998) applied neural network to predict management

fraud. Using public available predictors from fraudulent financial statements, they developed a model of eight variables with a high probability of detection. There are also some other works along this line using neural network for financial statement fraud detection (Lin et al., 2003), (Thiprungsri & Vasarhelyi, 2011). Another group of researches applied regression methods to study financial statement fraud detection. For example, Bell and Carcello (Bell & Carcello, 2000) developed a logistic regression model that estimated the likelihood of fraudulent financial reporting for an audit client, conditioned on the presence or absence of several fraud-risk factors. Abbott et al. (Abbott et al., 2002) employed statistical regression analysis to examine if the existence of an independent audit committee mitigated the likelihood of fraud. Spathis (Spathis, 2002) also constructed a model to detect falsified financial statements using logistic regression. The result suggested that it was promising to detect falsified financial statement through the analysis of published financial statements.

Based on earlier works, some comprehensive studies have been proposed in this area. For example, Kirkos et al. (Kirkos et al., 2007) conducted a comprehensive study for fraudulent financial statement (FFS) detection. The dataset included a sample of 38 FFS and 38 non-FFS in Greece, and ten financial variables as potential predictors of FFS. Three data mining models - neural network, decision tree and Bayesian belief network - had been employed. The study investigated the usefulness of these models in the identification of FFS. In addition, Yue et al. (Yue, Wu, Wang, Li, & Chu, 2007) summarized the previous literatures about financial statement fraud detection. The authors reviewed the literature of data mining based financial fraud detection researches and generalized the previous works based on learning tasks and

learning algorithms. While the above studies support the significance of supervised learning techniques to analyze financial statement fraud, there has been little research in literature in the direction of applying unsupervised learning techniques, such as clustering analysis, to financial statement fraud analysis.

4.3 Data Collection and Preprocessing

We collect a list of 801 companies as the target firms of this study in the time period of 1981-2011 that were either charged bankruptcy or suspended from stock market according to the UCLA bankruptcy research database[1] . The 801 companies are grouped into 8 business sectors as follows: A) Agricultural Production Crops, B) Mining, C) Construction, D) Manufacturing, E) Transportation, Communication, Electric, and Gas, F) Wholesale, G) Retail Trade, and I) Services. We exclude the companies in the financial industry from the list since the structure of their financial statements differs from companies in other industries.

Then, we retrieve the raw financial statement data of the listed companies from COMPUSTAT database of Wharton Research Data Service (WRDS)[2] , which is the data source for a large portion of bankruptcy studies in literature. For each company, we extract its financial statements of three consecutive years backwards since the year they were charged for bankruptcy or the year they were announced as trading suspension. As a result, we collect a raw dataset which consists of 2403 instances (the financial statement for one company of one year is treated as an instance) and 125 features for each instance. Among the 125 features, 14 fields are identifiers and

[1] http://lopucki.law.ucla.edu/index.htm

[2] https://wrds-web.wharton.upenn.edu/wrds

company information, while the rest 111 fields are account information extracted from the basic formal financial statements.

4.4 Data Cleaning and Transformation

Most of the features extracted from the basic formal financial statements can not be directly used for research purposes, since they provide very little information. In accounting literature, a large number of financial ratios and indicators which are derived from the raw data have been proposed to gain a deeper insight of a companys financial healthy status. For example, Altmans Z-score, an indicator which is extensively used in the areas of bankruptcy prediction and financial distress analysis, is calculated based on 8 account variables (Altman, 1968). We go through a literature survey in the areas of fraudulent financial statement and financial risk analysis (Altman, 1968), (Fanning & Cogger, 1998), (Feroz et al., 2000), (Loebbecke et al., 1989), (Persons, 2011), (Stice et al., 1991), (Stice, 1991), and select 30 financial indicators that are proved to be significantly indicative as the features of our dataset. Table 4.1 lists the name of the selected 30 features. The detailed calculation formulas and descriptions of these indicators are listed in Appendix A.

On the other hand, we remove the instances for companies with inconsistent account information in different years to preserve data consistency. In addition, we also remove instances from business sectors with insufficient number of companies in order to make sure the remaining sectors are statically comparable to each other. These sectors are the agricultural sector and the construction sector, since they only have 3 and 16 companies respectively.

Table 4.1: Financial indicators derived from raw data

1. Z-SCORE	2. LOGDEBT	3. DEBTEQ	4. TDTA	5. SALGRTH
6. RECSAL	7. RECA	8. INVSAL	9. INVTA	10. COSAL
11. GPTA	12. NPTA	13. RETA	14. ROS	15. ROE
16. ROA	17. NPSAL	18. LTA	19. WCAP	20. NFATA
21. SALTA	22. CACL	23. NIFA	24. CASHTA	25. QACL
26. EBIT	27. LTDTA	28. ACID-TEST	29. SOLVENCY	30. MKVALT

Table 4.2 lists a general summary of the dataset after data cleaning and transformation. Our dataset includes a total of 1611 instances from 537 companies spread in 6 business sectors. The numbers of instances of each business sector are also included.

Table 4.2: Data summary

Business Sector	**# Company**	**# Instance**
B: Mining	25	75
D: Manufacturing	218	654
E: Transportation, Communications, Electric, Gas	102	306
F: Wholesale	26	78
G: Retail Trade	84	252
I: Services	82	246
Grand Total	537	1611

4.5 Data Mining Approach – K-means Clustering

We employ k-means clustering technique to analyze the clustering effect in bankrupt companies of different business sectors. K-means clustering algorithm is a simple, efficient, and well known data mining method for cluster analysis which can be used for a wide variety of data types. It aims to partition n objects into k clusters (usually user specified) in which each object belongs to the cluster with the nearest mean (Tan et al., 2005). First of all, each object is represented by a vector of numerical features in an m-dimensional space. The algorithm randomly chooses k objects out of the n objects as the starting initial centroids of the k clusters. Each object is then assigned to the cluster whose centroid it is closest to based on a predefined distance measure, which varies from the characteristics of the data and the objective of the learning task. After all objects are re-assigned, the centroid of each cluster is re-computed. The process of assigning objects and re-computing centroids is repeated until no object changes its cluster label. The algorithm is proved to converge after a finite number of iterations. The pseudo code of the k-means algorithm is described in Figure 4.1.

4.6 Experimental Evaluation

In this section, we conduct experiments to identify the business sector with the strongest clustering effect and the financial indicators with the most significant indicative power to best describe the financial characteristics of the bankrupt companies in the sector.

1. Select k objects as the initial centroids
2. **repeat**
3. Form k clusters by assigning each object to its closest centroid
4. Re-compute the centroid of each cluster
5. **until** centroids do not change

Figure 4.1: K-means clustering algorithm

4.6.1 Experiment Setup

We employ the k-means algorithm implemented in CLUTO (Karypis, 2007), a software package for clustering low and high dimensional datasets and analyzing the characteristics of various clusters, to analyze our dataset. CLUTO is well-suited for clustering datasets arising in many diverse application areas including information retrieval, customer purchasing transactions, web, GIS, science, and biology.

Regarding the user specified parameters in k-means algorithm, we set the number of clusters k to be 6, since there are 6 natural clusters in the dataset which come from 6 business sectors (mining, manufacturing, transportation, wholesale, retail, and services). Consider the dimensionality of the data (30 dimensions), we choose cosine distance rather than the traditional Euclidean distance as the similarity measure, since it is difficult for the Euclidean distance measure to separate objects in a high

dimensional space where data becomes increasingly sparse.

4.6.2 Experimental Results

K-means algorithm has the uniform effect, which tends to produce clusters with relatively uniform sizes (Xiong, Wu, & Chen, 2009). Since the size of the 6 business sectors varies in a large range, we randomly sample instances from business sectors with equal size to make them comparable with each other. Mining industry is the smallest business sector with 25 companies among all sectors, therefore, we set the sample size for each industry to be 25 (i.e. 75 instances). The derived sample data contains 151 companies and 453 instances (26 companies for wholesale industry). K-means algorithm is then applied on the sample data to get the clustering result.

Table 4.3 shows the clustering result on a sample data. We can observe the distribution of instances from different business sectors in each cluster. For example, cluster 0 has 178 instances in total and 50 (28% in percentage) of them are from sector B. And we assign each cluster a label as the name of the business sector from which objects have the largest proportion. In other words, each instance in that cluster is considered as from the industry with majority objects. For example, sector E (transportation) has the highest percentage in cluster 0 (29%), therefore, cluster 0 is labeled as the transportation industry and the instances from other business sectors in this cluster are all considered as originally sampled from sector E. Clearly, the quality of cluster 0 is not good. In this sample data, cluster 2 has the greatest clustering quality, since 75% instances in the cluster are from the retail trade industry.

The quality of the clustering result is usually measured by intra-cluster compact-

Table 4.3: Clustering result of a sample data

Business Sector	**Clu.0**	**Clu.1**	**Clu.2**	**Clu.3**	**Clu.4**	**Clu.5**	**Total**
B: Mining	50(28%)	5(6%)	—	3(23%)	2(2%)	15(43%)	75
D: Manufacturing	14(8%)	22(25%)	2(3%)	7(54%)	28(34%)	2(6%)	75
E: Transportation	51(29%)	4(5%)	—	—	8(10%)	12(34%)	75
F: Wholesale	12(7%)	37(43%)	13(22%)	1(8%)	15(18%)	—	78
G: Retail Trade	10(6%)	8(9%)	**43(75%)**	—	13(16%)	1(3%)	75
I: Services	41(23%)	11(13%)	—	2(15%)	16(20%)	5(14%)	75
Grand Total	178	87	58	13	82	35	453

ness (i.e. cohesion) and inter-cluster separation (Liu et al., 2013). Cohesion measures how closely related the objects in a cluster are, while separation measures how distinct or well-separated a cluster is from other clusters. In general, cohesion plays a more important role than separation of clustering validation measures. There have been numerous cohesion measures proposed in literature (Liu et al., 2013), and purity and entropy are two of the most widely used cohesion measures among them.

Purity is a simple indicator which measures the purity of a cluster as the ratio of the number of objects from the majority class to the total number of objects in the entire cluster. For example, the purity of cluster 2 is 0.75. A higher value of purity indicates a better clustering result.

Entropy is another cohesion measure that indicates the purity of a clustering result. The entropy of a cluster is calculated as

$$E = -\sum_{i} p_i \cdot log(p_i)$$

where p_i is ratio of number of instances from the ith sector to the total number of instances in the cluster. For example, the entropy of cluster 2 is $-(0.03 * log(0.03) + 0.22 * log(0.22) + 0.75 * log(0.75)) = 0.28$. A lower value of entropy indicates a better clustering result.

The slight difference between our cohesion measures and the ones in literature is that we calculate the purity and entropy of each cluster rather than the overall purity and entropy of all 6 clusters, since the intention of this study focuses on comparing the clustering effect of different clusters.

In order to make the experiment results statically significant, we derive 30 samples from the original dataset, and then calculate the sample mean and standard deviation of purity and entropy for each business sector. The results are listed in Table 4.4. The sample standard deviations are listed in parentheses.

Table 4.4: Sample mean and standard deviation of purity and entropy for each business sector

	Sector B	Sector D	Sector E	Sector F	Sector G	Sector I
Purity	0.48(0.11)	0.38(0.05)	0.32(0.03)	0.51(0.12)	**0.60(0.10)**	0.41(0.11)
Entropy	0.55(0.11)	0.70(0.02)	0.65(0.09)	0.60(0.09)	**0.46(0.11)**	0.53(0.10)

4.6.3 Result Analysis

Based on the results in Table 4.4, we can observe that sector G (retail trade) has the best clustering effects among all 6 business sectors, since it has the highest purity and lowest entropy. We conduct a two-sample z-test to examine whether the sample

means of the purity and entropy of sector G are statistically significant better than those of other business sectors. Therefore, we have the following hypotheses on purity:

Null Hypothesis (H0): $\mu(Purity(G)) = \mu(Purity(*))$

Alternative Hypothesis (H1): $\mu(Purity(G)) > \mu(Purity(*))$

where * stands for sector B, D, E, F and I. The hypotheses on entropy are similar as the ones on purity.

The p-values of the hypothesis tests are shown in Table 4.5. All the p-values are less than 0.01, and most of them are below 0.001. Therefore, it is safe to reject the null hypothesis. In other words, sector G has the best clustering effects among all 6 business sectors, which indicates that bankrupt companies in retail trade industry are similar to each other and share common financial characteristics.

We may explain the strong clustering effect in retail industry from two aspects. First of all, companies in retail industry are not as various as companies in other business sectors, thus there are common characteristics identifiable through a review of their financial statements, cash flow statements, and ratios (McGurr & DeVaney, 1998). Secondly, the retail industry is greatly affected by economic recessions and growth cycles. The reasons for bankruptcy can be very similar for companies in retail business, therefore yields a strong clustering effect.

4.6.4 The Most Significant Financial Indicators

From the above analysis, we conclude that the retail trade industry has the strongest clustering effect. The next thing we would like to study is the financial indicators that have the most significant indicative power to best describe the financial characteristics

Table 4.5: p-values of hypothesis tests on sample means for purity and entropy

	Sector B	Sector D	Sector E	Sector F	Sector I
Hypothesis Test on Purity	***	***	***	***	***
Hypothesis Test on Entropy	***	***	***	***	**
**: p-value < 0.01; ***: p-value < 0.001					

of the companies in retail business sector. We can achieve the goal by leveraging the figure of contributed features produced by CLUTO.

We employ k-means algorithm to group all 537 companies (1611 instances) into 6 clusters, and observe that most of retail trade companies are clustered into cluster 2. The corresponding figure of contributed features is shown as Figure 4.2. In this figure, contributions for each feature to each cluster are displayed by different level of shades (no shade means no contribution). Feature with a deeper level of shade indicates a larger contribution when measuring the intra-cluster similarity, while a lighter level of shade indicates a larger contribution when distinguishing one cluster from others; in other words, features with the deepest and the lightest levels of shade have the highest indicative power.

According to Figure 4.2 and the associated statistics, the five most descriptive features for cluster 2 are NPSAL, LOGDEBT, RETA, DEBTEQ, and Z-score. They explain the similarities among the bankrupt companies in the retail industry. On the other hand, the five most discriminating features for cluster 2 are INVTA, NFATA, SALTA, LTDTA, and INVSAL. They explain the dissimilarities of the retail cluster to other clusters. These 10 features have the most significant indicative power to

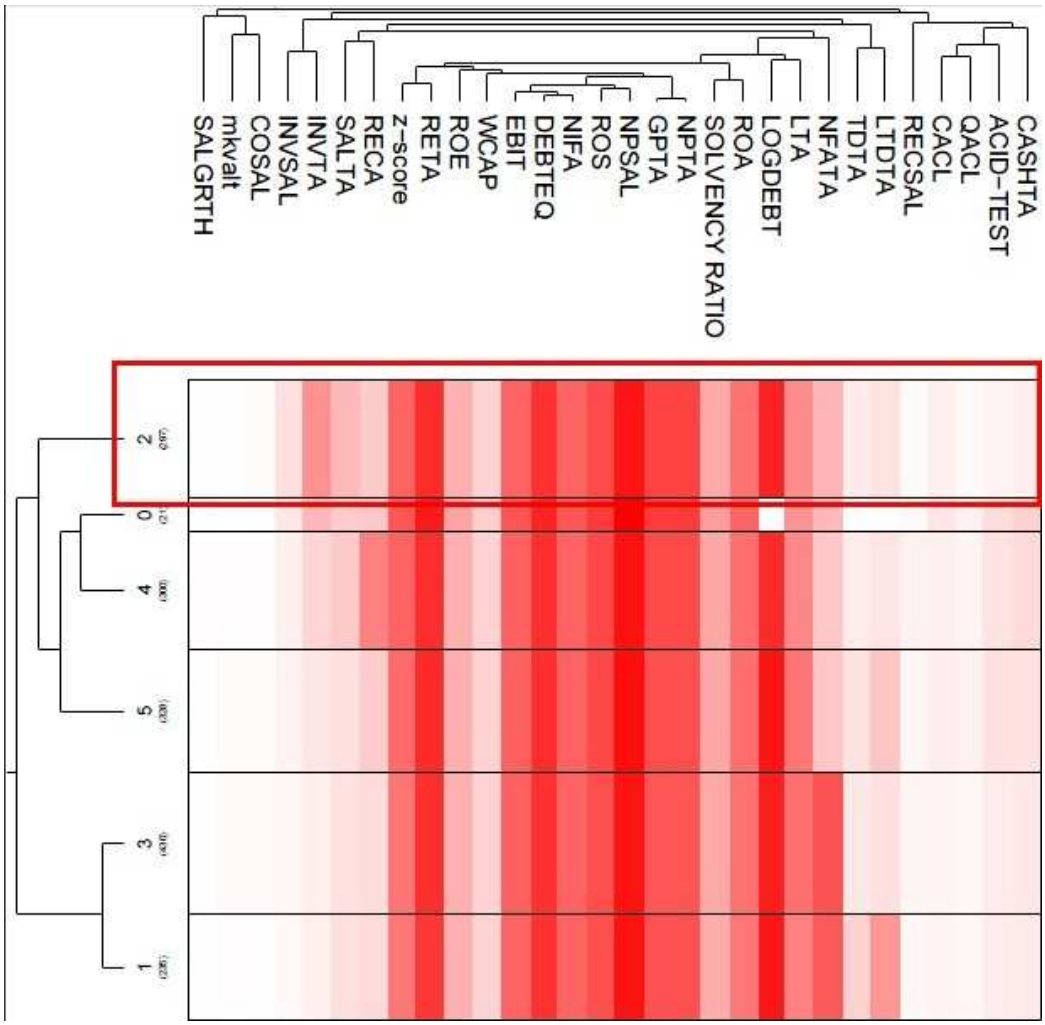


Figure 4.2: Contributed features of each cluster

best describe the financial characteristics of the bankrupt companies in retail business sector. Generally, they represent 4 aspects of the financial activities of a firm (Persons, 2011).

(1) Profitability (RETA, NPSAL, SALTA): Lower profit may cause the firm in bad financial situation. So the values of these features are expected to be negative. Bankrupt companies are usually less profitable.

(2) Financial Leverage (DEBTEQ, LOGDEBT, LTDTA): Higher leverage is typi-

cally associated with higher potential for violations of loan agreements and less ability to obtain additional capital through borrowing. Bankrupt companies usually have high financial leverages.

(3) Asset Liquidity (NFATA, INVTA, INVSAL): Lower liquidity may provide an incentive for the bankruptcy when the firm is in unhealthy financial situation. Bankrupt companies usually have low liquidity.

(4) Overall Financial Position (Z-score): Z-score measures the bankruptcy probability of a firm. A smaller Z-score is more likely to engage in bankrupt crisis.

4.7 Summary

Todays auditing and risk management are becoming increasing important for companys development and governments regulation since it copes with growing number of fraud and bankruptcy cases. Data mining techniques, with advanced clustering and predictive capabilities, could facilitate auditors and managers in accomplishing the task of management fraud detection and early-warning decision making. The aim of this study is to utilize clustering techniques to extract useful knowledge from bankrupt companies using published financial statement data.

Although our study is still in the early stage, we have extracted some valuable information for future researches. From the experimental results, we have learned retail trade industry has the strongest clustering effect among all 6 business sectors. We have also identified 10 of the most indicative financial features for the bankrupt companies in retail business.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Summary

In this dissertation, we explored the problem of how advanced data mining techniques can play essential roles in financial risk management in today's data-intensive business world. The focus was to develop effective and efficient data analysis techniques to detect financial anomalies and mitigate potential risks, by addressing the unique characteristics of particular business applications. Specifically, three case studies were provided to apply advanced data mining techniques in the following applications.

First of all, we explored the problem of detecting a specific type of trading fraud in the financial market, namely the trading ring pattern. We formulated the problem of mining blackhole and volcano patterns in a directed graph. To reduce the complexity of the problem, we first showed that the problem of finding blackhole patterns was a dual problem of finding volcano patterns. Thus, we could be only focused on mining blackhole patterns. To that end, we proposed two pruning approaches to reduce the computational cost by decreasing both the number of combinations and the average computational cost for each combination. In the first pruning approach, we introduced the concept of combination dominance to help develop a pruning technique to reduce the exponentially growing search space. Based on this pruning technique, we

developed the gBlackhole algorithm for finding top-K blackhole patterns. The second pruning approach, named the approxBlackhole algorithm, was an approximate algorithm to further decrease the computational complexity over the gBlackhole algorithm. This approxBlackhole algorithm first filtered out nodes with small diff-weights to reduce the size of the graph, and then found the top-K blackhole patterns in the subgraph induced by the rest of the nodes. There was a trade-off between the efficiency and completeness of the approxBlackhole algorithm.

Secondly, we investigated the problem of evaluating potential risks for new service contracts by measuring the similarity of contracts. It was motivated by the understanding that experiences and lessons learnt from historical contracts were of great value to IT service providers for risk management. A key challenge was how to define the right metric to measure contract similarity when there were continuous distance measurements between contract pairs. We formulated the problem as an extension of the existing Mahalanobis distance metric learning problem, which could only deal with binary distance measurements. We proposed the Gradient Descent alternating Iterative Projection method (GDIP) to solve the problem efficiently.

Last but not least, we examined the application of applying clustering techniques to extract useful knowledge from bankrupt companies using published financial statement data. We utilized the advantages in both the accounting and data mining areas to solve the problem. First of all, we leveraged the domain knowledge in accounting area to select valuable features from financial statement. Then we employed clustering techniques to identify business sector in which bankrupt companies has strong clustering effects. We learnt retail trade industry had the strongest clustering effect

among all 6 business sectors, and also identified 10 of the most indicative financial features for the bankrupt companies in retail business.

5.2 Future Research Direction

In the future, I would like to explore along the following research directions.

Detection of Evolving Blackhole and Volcano Patterns. By incorporating the ideas and techniques from the areas of frequent subgraph mining and community evolution detection, it would be possible to extend the blackhole and volcano patterns to a dynamically evolving fashion. We can study the lifetime cycle of these patterns and utilize to discover new underlying knowledge. For the trading ring pattern example, if some of the evolving blackhole and volcano patterns could be identified at their early stages, mitigation steps could be conducted accordingly. It would also be helpful to detect new collaborative fraudsters when a new blackhole pattern is emerging.

Bankruptcy Prediction using Financial Statement. For the next stage, we plan to focus on the retail trade industry and have a deeper insight into it. We plan to build more concrete bankruptcy prediction model to study the bankrupt companies in the retail trade industry based on the selected 10 features. We can also uncover some hidden some patterns from the bankrupt companies. Another direction of future our research is to improve the performance of the current framework by introducing additional methods. For example, we could take the rare class problem into consideration during the model building procedure. We can also explore some other external information from stock data and companys governance and board

information, and build network connections among companies. Network analysis methods can then be applied in this setting.

Identification of Mini Flash Crash in High-Frequency Trading World. The U.S. financial markets experienced a flash crash on the day of May 6, 2010 - a sudden price drop of more than 5 percent within 3 minutes and recovered and regained most of the losses shortly after. The brief period of extreme intra-day volatility raises a number of questions about the structure and stability of today's financial markets, as well as the increasing role played by high-frequency trading (HFT). Along this line, we would like to investigate the inherent properties of the mini flash crash - a short-term crash with similar patterns to the Flash Crash, and its relationship with HFT. We plan to reveal the characteristics of the mini flash crash using technical analysis, which seeks to identify price patterns and market trends in financial markets by looking at the historical trading data. We need to address the following three challenges. 1) develop quantitative measures to define a mini flash crash; 2) generate technical indicators reflecting the properties of mini flash crash and combine different indictors; 3) handle the computational challenge of the overwhelming real-time data volume.

BIBLIOGRAPHY

Abbott, L. J., Parker, S., & Peters, G. F. (2002). Audit committee characteristics and financial misstatement: A study of the efficacy of certain blue ribbon committee recommendations. *document de travail*.

Adamic, L., Brunetti, C., Harris, J., & Kirilenko, A. (2010). Trading networks. *SSRN eLibrary, http://ssrn.com/paper=1361184* .

Akoglu, L., McGlohon, M., & Faloutsos, C. (2010). Oddball: Spotting anomalies in weighted graphs. In *Proceedings of the 14th pacific-asia conference on knowledge discovery and data mining (pakdd'10)* (p. 410-421).

Altman, E. I. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The journal of finance*, *23*(4), 589–609.

Apparao, G., Singh, A., Rao, G., Bhavani, B. L., Eswar, K., & Rajani, D. (2009). Financial statement fraud detection by data mining. *Inernational Journal of Advanced Networking and Applications*, *1*(3), 159–163.

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2003). Learning distance functions using equivalence relations. In *Proceedings of the 20th international conference on machine learning* (p. 11-18).

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data*. John Wiley and Sons.

Bell, T. B., & Carcello, J. V. (2000). A decision aid for assessing the likelihood of fraudulent financial reporting. *Auditing: A Journal of Practice & Theory*, *19*(1), 169–184.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). Lof: Identifying density-based local outliers. In *Proceedings of the 2000 acm sigmod international conference on management of data (acm sigmod'00)* (p. 93-104).

Chakrabarti, D. (2004). Autopart: Parameter-free graph partitioning and outlier detection. In *Proceedings of the 8th european conference on principles and practice of knowledge discovery in databases (pkdd'04)* (p. 112-124).

Chapman, C., & Ward, S. (1996). *Project risk management: processes, techniques and insights*. John Wiley.

Chaudhary, A., Szalay, A. S., & Moore, A. W. (2002). Very fast outlier detection in large multidimensional data sets. *Data Mining and Knowledge Discovery*.

Chlistalla, M., Speyer, B., Kaiser, S., & Mayer, T. (2011). High-frequency trading. *Deutsche Bank Research*, *7*.

Chordia, T., Goyal, A., Lehmann, B. N., & Saar, G. (2013). High-frequency trading. *Journal of Financial Markets*, *16*(4), 637–645.

Clark, N. (October 5, 2010). Rogue trader at societe generale gets jail term. *New York Times*.

Cook, D. J., & Holder, L. B. (1994). Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research (JAIR)*, *1*, 231-255.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms*. The MIT Press.

Cox, T., & Cox, M. (2000). *Multidimensional scaling*. Chapman & Hall/CRC.

Deleris, L. A., Katircioglu, K., Kapoor, S., Lam, R., & Bagchi, S. (2007). Adaptive project risk management. In *Service operations and logistics, and informatics, 2007. soli 2007. ieee international conference on* (pp. 1–6).

Diestel, R. (2006). *Graph theory (graduate texts in mathematics)*. Springer.

Domeniconi, C., Gunopulos, D., et al. (2002). Adaptive nearest neighbor classification using support vector machines. In *Advances in neural information processing systems (nips)* (pp. 665–672).

Fanning, K. M., & Cogger, K. O. (1998). Neural network detection of management fraud using published financial data. *International Journal of Intelligent Systems in Accounting, Finance & Management*, *7*(1), 21–41.

Feroz, E. H., Kwon, T. M., Pastena, V. S., & Park, K. (2000). The efficacy of red flags in predicting the sec's targets: An artificial neural networks approach. *International Journal of Intelligent Systems in Accounting, Finance & Management*, *9*(3), 145–157.

Gehrke, J., Ginsparg, P., & Kleinberg, J. M. (2003). Overview of the 2003 kdd cup. In *Acm sigkdd explorations 5(2)* (p. 149-151).

Ghosh, R., & Lerman, K. (2008). Community detection using a measure of global influence. In *The 2nd sna-kdd workshop on social network mining and analysis (sna-kdd'08).*

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. In *Proceedings of the national academy of sciences.*

Gomber, P., Arndt, B., Lutat, M., & Uhle, T. (2011). High-frequency trading. *Available at SSRN 1858626*.

Goo, J., Kishore, R., Nam, K., Rao, H. R., & Song, Y. (2007). An investigation of factors that influence the duration of it outsourcing relationships. *Decision Support Systems*, *42*(4), 2107–2125.

Green, B. P., & Choi, J. H. (1997). Assessing the risk of management fraud through neural network technology. *Auditing: A Journal of Practice & Theory*, *16*(1), 14–28.

Han, J., Kamber, M., & Pei, J. (2011). *Data mining: concepts and techniques: concepts and techniques*. Elsevier.

Hawkins, D. (1980). *Identification of outliers*. Chapman and Hall.

Hopcroft, J., Khan, O., Kulis, B., & Selman, B. (2003). Natural communities in large linked networks. In *Proceedings of the 9th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'03).*

Huan, J., Wang, W., & Prins, J. (2003). Efficient mining of frequent subgraphs in the presence of isomorphism. In *Proceedings of the 3rd ieee international conference on data mining (ieee icdm'03).*

Jiang, X., Xiong, H., Wang, C., & Tan, A. H. (2009). Mining globally distributed frequent subgraphs in a single labeled graph. *Data and Knowledge Engineering*, *68*, 1034-1058.

Johnson, R. A., & Wichern, D. W. (1998). *Applied multivariate statistical analysis*. Prentice Hall.

Jolliffe, I. (2005). *Principal component analysis*. Wiley Online Library.

Karush, W. (1939). *Minima of functions of several variables with inequalities as side constraints*. Masters thesis, Dept. of Mathematics, Univ. of Chicago.

Karypis, G. (2007). Cluto - software for clustering high-dimensional datasets. *Internet Website, http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview*.

Kirkos, E., Spathis, C., & Manolopoulos, Y. (2007). Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, *32*(4), 995–1003.

Knuth, D. (2011). *The art of computer programming, volume 4a: Combinatorial algorithms*. Addison-Wesley.

Koh, H. C., Tan, G., et al. (2011). Data mining applications in healthcare. *Journal of healthcare information management*, *19*(2), 65.

Kohavi, R., Mason, L., Parekh, R., & Zheng, Z. (2004). Lessons and challenges from mining retail e-commerce data. *Machine Learning*, *57*(1-2), 83–113.

Kovalerchuk, B., & Vityaev, E. (2000). *Data mining in finance: advances in relational and hybrid methods*. Springer Science & Business Media.

Kuhn, H., & Tucker, A. (1951). Nonlinear programming. In *Second berkeley symposium on mathematical statistics and probability* (pp. 481–492).

Kuramochi, M., & Karypis, G. (2005). Finding frequent patterns in a large sparse graph. *Data Mining and Knowledge Discovery*, *11*(3), 243-271.

Kwok, J., & Tsang, I. (2003). Learning with idealized kernels. In *Proceedings of the 20th international conference on machine learning* (p. 400-407).

Lazarevic, A., & Kumar, V. (2005). Feature bagging for outlier detection. In *Proceedings of the 11th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'05)* (p. 157-166).

Leskovec, J., & Faloutsos, C. (2006). Sampling from large graphs. In *Proceedings of the 12th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'06)* (p. 631-636).

Leskovec, J., Huttenlocher, D., & Kleinberg, J. (2010a). Predicting positive and negative links in online social networks. In *Proceedings of the 19th international world wide web conference (www'10).*

Leskovec, J., Huttenlocher, D., & Kleinberg, J. (2010b). Signed networks in social media. In *Proceedings of the 28th acm conference on human factors in computing systems (chi'10).*

Leskovec, J., Kleinberg, J., & Faloutsos, C. (2005). Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of the 11th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'05).*

Leskovec, J., Lang, K., Dasgupta, A., & Mahoney, M. (2008). Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. In *arxiv.org:0810.1355.*

Leung, H., Rao Tummala, V., & Chuah, K. (1998). A knowledge-based system for identifying potential project risks. *Omega*, *26*(5), 623–638.

Li, Z., Xiong, H., Liu, Y., & Zhou, A. (2010). Detecting blackhole and volcano patterns in directed networks. In *Proceedings of the 10th ieee international conference on data mining (ieee icdm'10)* (p. 294-303). Australia.

Lin, J. W., Hwang, M. I., & Becker, J. D. (2003). A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal*, *18*(8), 657–665.

Liu, Y., Li, Z., Xiong, H., Gao, X., Wu, J., & Wu, S. (2013). Understanding and enhancement of internal clustering validation measures. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *43*(3), 982–994.

Loebbecke, J. K., Eining, M. M., & Willingham, J. J. (1989). Auditors experience with material irregularities-frequency, nature, and detectability. *AUDITING-A JOURNAL OF PRACTICE & THEORY*, *9*(1), 1–28.

McGurr, P., & DeVaney, S. (1998). Predicting business failure of retail firms: an analysis using mixed industry models. *Journal of Business Research*, *43*(3), 169–176.

McNeil, A. J., Frey, R., & Embrechts, P. (2015). *Quantitative risk management: Concepts, techniques and tools: Concepts, techniques and tools*. Princeton university press.

Mehlhorn, K., & Naher, S. (1999). *The leda platform of combinatorial and geometric computing*. Cambridge University Press.

Mojsilović, A., Ray, B., Lawrence, R., & Takriti, S. (2007). A logistic regression framework for information technology outsourcing lifecycle management. *Computers & operations research*, *34*(12), 3609–3627.

Moonesinghe, H. D. K., & Tan, P.-N. (2008). Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools*, *17(1)*.

Newman, M. E. J. (2004). Detecting community structure in networks. *The European Physical Journal B*, *38*, 321-330.

Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E 69 026113*.

Noble, C. C., & Cook, D. J. (2003). Graph-based anomaly detection. In *Proceedings of the 9th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'03)* (p. 631-636).

Papadimitriou, S., Kitagawa, H., Gibbons, P. B., & Faloutsos, C. (2003). Loci: Fast outlier detection using the local correlation integral. In *Proceedings of the 19th international conference on data engineering (icde'03)* (p. 315-326).

Pathak, N., DeLong, C., Banerjee, A., & Erickson, K. (2008). Social topic models for community extraction. In *The 2nd sna-kdd workshop on social network mining and analysis (sna-kdd'08)*.

Peng, J., Heisterkamp, D., & Dai, H. (2002). Adaptive kernel metric nearest neighbor classification. In *Proceedings of the 16th international conference on pattern recognition* (pp. 33–36).

Persons, O. S. (2011). Using financial statement data to identify factors associated with fraudulent financial reporting. *Journal of Applied Business Research (JABR)*, *11*(3), 38–46.

Sasisekharan, R., Seshadri, V., & Weiss, S. M. (1996). Data mining and forecasting in large-scale telecommunication networks. *IEEE Intelligent Systems*(1), 37–43.

Saul, L., & Roweis, S. (2003). Think globally, fit locally: unsupervised learning of low dimensional manifolds. *The Journal of Machine Learning Research*, *4*, 119–155.

Saunders, A., Cornett, M. M., & McGraw, P. A. (2006). *Financial institutions management: A risk management approach* (Vol. 8). McGraw-Hill/Irwin.

Spathis, C. T. (2002). Detecting false financial statements using published data: some evidence from greece. *Managerial Auditing Journal*, *17*(4), 179–191.

Steyvers, M., Smyth, P., Rosen-Zvi, M., & Griffiths, T. (2004). Probabilistic author-topic models for information discovery. In *Proceedings of the 10th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'04).*

Stice, J. D. (1991). Using financial and market information to identify pre-engagement factors associated with lawsuits against auditors. *Accounting Review*, 516–533.

Stice, J. D., Albrecht, S., & Brown, L. (1991). Lessons to be learnedzzzz best, regina, and lincoln savings. *The CPA Journal*, *61*(4), 52–53.

Sun, J., Qu, H., Chakrabarti, D., & Faloutsos, C. (2005). Neighborhood formation and anomaly detection in bipartite graph. In *Proceedings of the 5th ieee international conference on data mining (ieee icdm'05)* (p. 418-425).

Tan, P.-N., Steinbach, M., & Kumar, V. (2005). *Introduction to data mining.* Addison Wesley.

Tenenbaum, J., De Silva, V., & Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Thiprungsri, S., & Vasarhelyi, M. A. (2011). Cluster analysis for anomaly detection in accounting data: An audit approach. *The International journal of digital accounting research*, *11*, 69–84.

Wang, C., Wang, W., Pei, J., Zhu, Y., & Shi, B. (2004). Scalable mining of large disk-based graph databases. In *Proceedings of the 10th acm sigkdd international conference on knowledge discovery and data mining (acm sigkdd'04).*

Wang, J., Hsu, W., Lee, M., & Sheng, C. (2006). A partition-based approach to graph mining. In *Proceedings of the 22nd international conference on data engineering (icde'06)* (p. 74).

Xing, E., Ng, A., Jordan, M., & Russell, S. (2002). Distance metric learning, with application to clustering with side-information. In *Advances in neural information processing systems (nips)* (pp. 505–512).

Xiong, H., Wu, J., & Chen, J. (2009). K-means clustering versus validation measures: a data-distribution perspective. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, *39*(2), 318–331.

Yan, X., & Han, J. (2002). gspan: Graph-based substructure pattern mining. In *Proceedings of the 2nd ieee international conference on data mining (ieee icdm'02).*

Yang, L., & Jin, R. (2006). Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 1–51.

Yang, L., Jin, R., & Sukthankar, R. (2007). Bayesian active distance metric learning. In *Proceedings of the 23rd conference on uncertainty in artificial intelligence* (p. 442-449).

Yue, D., Wu, X., Wang, Y., Li, Y., & Chu, C.-H. (2007). A review of data mining-based financial fraud detection research. In *Wireless communications, networking and mobile computing, 2007. wicom 2007. international conference on* (pp. 5519–5522).

Zhang, K., Tang, M., & Kwok, J. (2005). Applying neighborhood consistency for fast clustering and kernel density estimation. In *Ieee computer society conference on computer vision and pattern recognition (cvpr)* (pp. 1001–1007).

Zhou, D., Manavoglu, E., Li, J., Giles, C. L., & Zha, H. (2006). Probabilistic models for discovering e-communities. In *Proceedings of the 15th international world wide web conference (www'06).*

Zmijewski, M. E. (1984). Methodological issues related to the estimation of financial distress prediction models. *Journal of Accounting Research*, 59–82.

APPENDIX

APPENDIX A

FINANCIAL STATEMENT INDICATORS DETAILS

Table A.1: Detailed formulas and descriptions of selected indicators

Indicator	**Formula**	**Description**
Z-SCORE	$T_1 = \frac{Working\ Capital}{Total\ Assets}$ $T_2 = \frac{Retained\ Earnings}{Total\ Assets}$ $T_3 = \frac{Earnings\ Before\ Interest\ and\ Taxes}{Total\ Assets}$ $T_4 = \frac{Market\ Value\ of\ Equity}{Total\ Liabilities}$ $T_5 = \frac{Sales}{Total\ Assets}$ $Z = 1.2 * T_1 + 1.4 * T_2 + 3.3 * T_3 +$ $0.6 * T_4 + 0.99 * T_5$	Insolvency Predictor - A predictor which predicts a company's probability of failure using 8 variables from a company's financial statements
LOGDEBT	$log(Total\ Debt)$	Logarithm of Total Debt

Continued on next page

Table A.1 – continued from previous page

Indicator	Formula	Description
DEBTEQ	$\frac{Liabilities}{Shareholders' \ Equity}$	A measure of a company's financial leverage calculated by dividing its total liabilities by stockholders' equity. It indicates what proportion of equity and debt the company is using to finance its assets
TDTA	$\frac{Total \ Debt}{Total \ Assets}$	A metric used to measure a company's financial risk by determining how much of the company's assets have been financed by debt

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
SALGRTH	*Sales Growth*	The ratio of sales growth. Measures the continuing growth of a company
RECSAL	$\frac{Accounts\ Receivable}{Sales}$	A ratio helps to identify recent increases in accounts receivable. When computes this ratio each month and then look at the changes that occur as the months pass, the accounts receivable to sales ratio can signal potential problems in the cash flow

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
RECA	$\frac{Accounts\ Receivable}{Total\ Assets}$	The portion of accounts receivable made of total assets
INVSAL	$\frac{Inventory}{Sales}$	Ratios to predict inventory related cash flow problems
INVTA	$\frac{Inventory}{Total\ Assets}$	
COSAL	$Sales - Gross\ Margin$	Ratios to decide whether gross margin is manipulated
GPTA	$\frac{Gross\ Profit}{Total\ Assets}$	
NPTA	$\frac{Net\ Profit}{Total\ Assets}$	An indicator of how effectively a company is using its assets to generate profit

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
RETA	$\frac{Retained\ Earnings}{Total\ Assets}$	A ratio indicates the extent to which assets have been paid for by company profits. A ratio near 100% indicates that growth has been financed through profits, not increased debt, vice versa
ROS	$\frac{Net\ Income\ before\ Taxes}{Sales}$	Variables indicate the effectiveness of company in generating profit
ROE	$\frac{Net\ Income\ before\ Taxes}{Shareholders'\ Equity}$	
ROA	$\frac{Net\ Income\ before\ Taxes}{Total\ Assets}$	
NPSAL	$\frac{Net\ Profit}{Sales}$	
LTA	$log(Total\ Assets)$	Logarithm of Total Assets

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
WCAP	$Current\ Assets - Current\ Liabilities$	Working Capital
NFATA	$\frac{Property\ Plant\ \&\ Equipment}{Total\ Assets}$	PP&E is the asset cannot be easily liquidated
SALTA	$\frac{Sales}{Total\ Assets}$	A ratio indicates how well using business assets to generate revenue. A higher ratio means a higher return on assets, which can compensate for a low profit margin
CACL	$\frac{Current\ Assets}{Current\ Liabilities}$	A measure of both a company's efficiency and its short-term financial health
NIFA	$\frac{Net\ Income}{Fixed\ Assets}$	The return ratio of the total investment

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
CASHTA	$\frac{Cash}{Total\ Assets}$	The liquidity of the Total Assets
QACL	$\frac{Quick\ Assets}{Current\ Liabilities}$	Quick Assets = Current Assets - Inventory
Continued on next page		

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
EBIT	$Earnings\ before\ Interest\ and\ Taxes$	A variable which nulls the effects of the different capital structures and tax rates used by different companies. By excluding both taxes and interest expenses, the figure hones in on the company's ability to profit and thus makes cross-company comparisons easier
LTDTA	$\frac{Long\text{-}term\ Debt}{Total\ Assets}$	Long-term Debt is loans and financial obligations lasting over one year

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
ACID- TEST RATIO	$\frac{Cash + Short\text{-}term\ Inverestment + REC}{Current\ Liabilities}$ $REC = Accounts\ Receivable$	A stringent test to determine whether a firm has enough short-term assets to cover its immediate liabilities without selling inventory. Companies with ratios less than 1 cannot pay their current liabilities and thus should be viewed with extreme caution

Continued on next page

Table A.1 – continued from previous page

Indicator	**Formula**	**Description**
SOLVENCY RATIO	$\frac{After\text{-}Tax\ Net\ Profit + Depreciation}{LT\ Liabilities + ST\ Liabilities}$	A ratio used to measure a companys ability to meet its long-term obligations. The lower a companys solvency ratio is, the greater is the likelihood that the company will default on its debt obligations
MKVALT	$Share\ Price * Number\ of\ Shares$	Market Value