SEMI-AUTOMATIC EXPLORATORY DATA ANALYTICS FOR ACTIONABLE DISCOVERIES
THROUGH SUBGROUP MINING

A Thesis

Presented to

The Faculty of the Graduate School
At the University of Missouri-Columbia

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By

DANLU LIU

Dr. Chi-Ren Shyu, Thesis Supervisor

JULY 2017

The undersigned, appointed by the dean of the Graduate School, have examined the
thesis entitled

SEMI-AUTOMATIC EXPLORATORY DATA ANALYTICS FOR ACTIONABLE DISCOVERIES
THROUGH SUBGROUP MINING

Presented by Danlu Liu

A candidate for the degree of

Master of Science

And hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Chi-Ren Shyu

Dr. Yunxin Zhao

Dr. Grant Scott

ACKNOWLEDGEMENTS

I would like to thank all the people who help me during the thesis work. Without your encouragement and companion, I cannot finish this task.

Firstly, I would like to appreciate my academic advisor Dr. Shyu, Chi-Ren. He gives me inspiration and advice to begin this research topic. Whenever I encounter some problems, he patiently gives me guidance and suggestions for me to continue this research. He is a knowledgeable and kind advisor; I am honored to learn from him during my graduate process.

Secondly, I would like to thank Dr. Yunxin Zhao and Dr. Grant Scott to serve on my thesis committee.

Thirdly, I want to thank my parents. They always encourage me when I face challenges in life. They give me hope and care. Thanks for their support all the time.

Finally, I want to thank my lab members. They give me invaluable suggestions and help during the experiments. Thanks to Matt Spencer for helping me edit this thesis.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	ii
LIST OF FIGURES.....	v
LIST OF TABLES.....	vi
ABSTRACT	vii
1. INTRODUCTION	1
1.1 Challenges	3
1.2 Thesis Organization	4
2. BACKGROUND AND RELATED WORK.....	6
2.1 Review of frequent pattern mining	6
2.2 Contrast Mining.....	7
2.3 Related Work.....	9
2.3.1 Measurement of contrast patterns	9
2.3.2 Methods to identify groups.....	9
2.3.3 Methods to rank groups	10
2.3.4 Methods to do exploratory mining	11
3. Expert-Guided Feature Partition.....	13
3.1 Terminology	13
3.2 Expert-Guided	15
4. A NEW WORKFLOW OF DATA ANALYSIS	16
4.1 Data Preprocessing.....	19
4.2 Inclusion and Exclusion Process Using SFFS.	21
4.2.1 SFS and SBS feature selection algorithm.	21
4.2.2 Floating algorithm.	22
4.3 Distributed Apriori algorithm and contrast pattern mining.	28
4.4 Criterion Function Design.....	29
4.4.1 Metrics of Contrast Patterns.	29
4.4.2 Good contrast patterns selection.	31

4.4.3	J functions design.....	32
4.5	Tracking Multiple Paths.....	35
4.6	Post-processing of selecting good contrast groups.....	36
5.	EXPERIMENTAL RESULTS.....	38
5.1	Experimental Design.....	38
5.1.1	Synthesized data creation.	38
5.1.2	Different J functions.	40
5.1.3	SFS VS SFFS.	42
5.1.4	Keep multiple results.....	42
5.1.5	The footprint of the branch traverse.....	44
5.2	Real Data Analysis	46
5.2.1	Data Source.	46
5.2.2	Different J functions.	46
5.2.3	Keep multiple results.....	47
6.	CONCLUSIONS	50
7.	FUTURE WORK.....	52
8.	REFERENCES.....	53

LIST OF FIGURES

<i>Figure 1. An Illustration of Contrast Set.</i>	<i>8</i>
<i>Figure 2. Flowchart of Exploratory analysis mining.....</i>	<i>12</i>
<i>Figure 3. An Example of Combination of Smoking and Alcohol.....</i>	<i>17</i>
<i>Figure 4. The architecture of finding clusters.</i>	<i>18</i>
<i>Figure 5. The process of SFS.</i>	<i>22</i>
<i>Figure 6. The process of SFFS.....</i>	<i>23</i>
<i>Figure 7.The Pseudo code for Inclusion and Exclusion.</i>	<i>25</i>
<i>Figure 8. The pseudo code of Algorithm 1.....</i>	<i>27</i>
<i>Figure 9. A travel path of candidate selection.....</i>	<i>37</i>
<i>Figure 10. Synthesized data running time of different search range of SFFS and SFS</i>	<i>41</i>
<i>Figure 11. Traverse Path of SFFS in 15% range.....</i>	<i>45</i>
<i>Figure 12. Real data running time of different search range of SFFS and SFS.....</i>	<i>47</i>

LIST OF TABLES

<i>Table 1. Relation table of features and categories.</i>	<i>14</i>
<i>Table 2. The Transactions of two groups.</i>	<i>32</i>
<i>Table 3. Contrast Pattern of two groups.....</i>	<i>32</i>
<i>Table 4. A comparison of h-index and g-index.</i>	<i>35</i>
<i>Table 5. Synthesized data group 1.</i>	<i>39</i>
<i>Table 6. Synthesized data group 2.</i>	<i>39</i>
<i>Table 7. Synthesized data group 3.</i>	<i>40</i>
<i>Table 8. The accuracy of different index.</i>	<i>41</i>
<i>Table 9. Accuracy of SFS</i>	<i>42</i>
<i>Table 10. Top 5 results of g-index in 5% range.....</i>	<i>43</i>
<i>Table 11. Top 5 results of g-index in 10% range.....</i>	<i>44</i>
<i>Table 12. Top 6 results of g-index in 15% range.....</i>	<i>44</i>
<i>Table 13. Top 10 results of real data.....</i>	<i>48</i>

ABSTRACT

People are born with the curiosity to see differences between groups. These differences are useful for understanding the root causes of certain discrepancies, such as populations and diseases. However, without prior knowledge of the data, it is extremely challenging to identify which groups differ most, let alone to discover what associations contribute to the differences. The challenges are mainly from the large searching space with complex data structure, as well as the lack of efficient quantitative measurements that are closely related to the meaning the differences.

To tackle these issues, we developed a novel exploratory data mining method to identify ranked subgroups that are highly contrasted for further in-depth analyses. The underpinning components of this method include (1) a semi-greedy forward floating selection algorithm to reduce the search space, (2) a deep-exploring approach to aggregate a collection of sizable and creditable candidate feature sets for subgroups identification using in-memory computing techniques, (3) a G-index contrast measurement to guide the exploratory process and to evaluate the patterns of subgroup pairs, and (4) a ranking method to provide mined results from highly contrasted subgroups.

Computational experiments were conducted on both synthesized and real data. The algorithm performed adequately in recognizing known subgroups and discovering new and unexpected subgroups. This exploratory data analysis method will provide a new paradigm to select data-driven hypotheses that will produce potentially successful actionable outcomes to tailor to subpopulations of individuals, such as consumers in E-commerce and patients in clinical trials.

1. INTRODUCTION

Nowadays, data plays a critical role in everyone's daily lives. As of 2012, about 2.5 exabytes of data are created each day, and that number is doubling every 40 months or so [1]. Many data scientists are trying to find the non-trivial information in the data. However, the significant amount of data brings not only interesting information but also computational challenges. The volume, velocity, and variety of data means that a single computer is not able to handle Big Data. As a result, the Big Data ecosystem has emerged to store, process, analyze, and visualize data. Through the analysis of data, users can access hidden relationships and properties that are invisible to ordinary observation.

Besides, Big Data is changing the research method ubiquitously. A well-known and often used research method is the hypothesis-driven approach. Researchers usually raise a hypothesis and then test it using deductive reasoning. The hypothesis-driven approach is easy to understand and use, but it requires that researchers understand the problem in advance to come up with a meaningful solution. It also often leads to confirmation bias, when researchers sometimes interpret data unfairly to confirm their hypothesis. Moreover, the researchers' experience and knowledge of the area may restrict the range of hypothesis. Exploratory data analysis is more suitable in the field of data mining because it lets data guide the formulation of hypotheses. In state-of-the-art exploratory data analysis, the data are used to form many hypotheses, which researchers can use to collect additional data and do new experiments. Those new findings could be previously unknown but truly useful for future research. The challenge of exploratory analysis is in

the interpretation of the results of the data mining. Thus, it is necessary to utilize statistical models that provide clear answers and are powerful enough to accomplish such exploratory research.

Contrast mining is a branch of data mining. It is an exploratory procedure that compares separate groups to discover major differences between them. These differences are called contrast patterns. Applications utilizing contrast mining appear in many areas [2-4]. In medicine, clinicians might want to discover the differences between healthy people and people with diabetes. In finance, economists might ask why one company has a better revenue than others have. This type of comparison between two groups is useful because examining a group in isolation would make it difficult to find the interesting features. While comparing groups, features that are found in both groups are less interesting and can be discarded, leaving major group differences. The Contrast mining approach identifies these differences between groups which reveal how these groups are separated.

The examples above are hypothesis-driven, so the hypothesis makes the groups choices clear. However, in many cases the groups are poorly defined or unknown. For example, autism patients are believed to have clinically relevant subgroups, but it is not known what these subgroups are. Exploratory mining can help by finding subgroups using data-driven algorithms. Subgroup discovery, the process of proposing useful groups to compare, is a common challenge in the data mining area. It is also necessary to determine how to represent a population group suitably. A definition of an induced subgroup description is in N. Lavrač, et al [5]. It provides a classification model to divide subgroups

by using the conjunction of attribute-value pairs selected from the features describing the training instances. Several applications are developed for subgroup discovery [6, 7]. It is time-consuming to exhaustively search for all combinations of population features. Besides, there are still some questions to be answered. Firstly, which attributes can be contrasted and are the compared groups meaningful or not. Due to the variety of population features, the pool of potential population groups is enormous. Identifying all the groups and their contrast patterns is computationally expensive. Secondly, are the contrast patterns significant or not. Thirdly, is there any method to evaluate the overall quality of the groups, to determine whether their differences are large or not. These three questions must be answered elegantly to create an effective subgroup discovery method.

1.1 Challenges

As stated above, many researchers currently only focus on mining the similarities and differences between predefined populations. This will ignore relationships between hidden populations. Subgroup discovery is needed to discover the unobvious groups so that contrast patterns between the groups can be found.

The first problem of this research is how to define the populations and subgroups. Some comparisons between populations are more meaningful than others with broader impacts. Therefore, a robust and interpretable model is required to select population groups, as well as pattern groups as an initial condition for an automatic algorithm. The guide of the expert is vital to have valuable mining results.

The second problem is how to reduce the search space and the search time. Contrast

Mining takes a significant amount of time even if the algorithm starts with all candidate populations. Combining contrast mining and subgroup discovery is expected to grow the search space and time exponentially. A distributed approach of the algorithm is needed to streamline the exploratory mining process.

The third problem is how to implement contrast mining and evaluate contrast patterns. In the contrast mining step, we need to use some statistical measures to determine which contrast patterns are useful. A function that evaluates all contrast patterns of the compared groups is needed for us to obtain an overall evaluation for populations. This thesis also discusses the ability to keep track on multiple candidate high-scoring subgroups.

The research aims at finding the useful groups to highlight for further study. The contrast patterns between two groups tell the differences between two population groups. By mining the data, we hope to find unexpected, non-trivial groups that have interesting and meaningful differences with potential actionable plans.

1.2 Thesis Organization

The rest of this thesis is organized in the following manner. Chapter 2 discusses traditional frequent set mining algorithms and contrast mining methods. Several contrast pattern metrics are also introduced in this chapter, along with related researches. Chapter 3 defines the features of populations and patterns. The process of subgroup forming is also outlined. Chapter 4 presents a new workflow for subgroup discovery including a novel evaluation function and a method for tracking multiple notable subgroups. Chapter 5

reports the experimental and analytic results. Finally, Chapters 6 and 7 summarize conclusions about our discoveries and briefly discuss future work.

The rest of this thesis is organized in the following manner. Chapter 2 discusses the background and related work about frequent pattern mining, contrast mining, identifying groups and exploratory mining. Chapter 3 defines the features of populations and patterns. The process of subgroup forming is also outlined. Chapter 4 presents a new workflow for data analysis including a novel evaluation function and a method for tracking multiple notable subgroups. Chapter 5 reports the experimental and analytic results. Finally, Chapters 6 and 7 summarize conclusions about our discoveries and briefly discuss future work.

2. BACKGROUND AND RELATED WORK

2.1 Review of frequent pattern mining

Frequent patterns and association rules are often used to discover the relationships between items in datasets. Association rules are based on frequent patterns discovered in the database. Those frequent patterns are the items and sets of items which appear frequently in the database. The definition of frequent items and association rules are first proposed by Agrawal, et al [8]. The Apriori algorithm is used to identify the items and combinations of items that are considered frequent, and association rules are generated based on those frequent items. The main idea of Apriori is listed as follows. Step 1: assume the algorithms has identified the n -dimension frequent item sets. Step 2: form the $(n+1)$ -dimension frequent item sets based on those n -dimensional frequent itemsets. The $(n+1)$ -dimension frequent item sets must follow two conditions: (a) If any sub-itemset of the $(n+1)$ -dimension itemset is not included in the n -dimension frequent itemsets, then the $(n+1)$ -dimension itemset is not a frequent item set. (b) The $(n+1)$ -dimensional item set itself must be a frequent itemset. Step 3: add the $(n+1)$ -dimensional frequent item sets to the collection of frequent itemsets and then return to Step 1. The algorithm begins to scan the whole database to identify the one-dimensional frequent item sets and then proceeds with Step 1. The advantage of Apriori is that it is easy to understand and implement. However, it needs to scan the database for the generation of each larger dimension of itemset to find the frequent patterns and the generation of candidate itemsets is very slow. To overcome the drawbacks of Apriori, another algorithm called FP-growth was

proposed by Han, et al [9]. It decreases the complexity of Apriori vastly by avoiding generating candidates of frequent pattern and repeatedly scanning the whole database. Frequent patterns can be generated from this tree structure by using a bottom-up algorithm called FP-growth. The advantage of this frequent pattern mining method is that the tree structure compresses data efficiently. There is also an elegant top-down frequent pattern mining method called FHPGrowth [10]. It also generates a tree structure called FHPTree to compress the data and takes a top-down traversal to generate frequent patterns. Unlike the FP-tree, that has a structure dependent on the data distribution, the number of nodes in FHPTree is only relative to the unique items in the whole database. Besides, the FHPGrowth is efficient at finding the maximal patterns.

2.2 Contrast Mining

People often hope to tell the story about the differences between subgroups. The interesting patterns can guide people to focus on certain subgroups and deepen their understanding about the causes that lead to the different situations. Figure 1 shows an example where group1 has more triangles than group2, but group2 has more stars conversely. Their difference is obvious because the dimension and data size are small. Fortunately, Contrast Mining excels at finding the differences between two or more classes. The objects that describe the differences across the classes are called contrast patterns. There are many synonyms used to describe the contrast patterns, including emerging patterns, contrast sets, discriminating patterns. For example, people who smoke are more likely to get lung cancer. In this example, the two population groups would be

people with lung cancer and people without lung cancer; the contrast pattern is smoking. There are multiple contrast patterns between groups, and those patterns can be a single item or the combination of several items.

Figure 1. An Illustration of Contrast Set.

Researchers have been developing computational methods in contrast mining. Bay and Pazzani [11] proposed the STUCCO approach to mine contrast patterns between the groups. They also proposed a pruning method to reduce the search time for contrast patterns. They treated the contrast patterns mining as a tree search problem and examined the quality of contrast patterns using support differences and the Chi-Square test. Emerging patterns mining was proposed by Dong [12] to solve runtime issues in the simple algorithms. Several measures are used in contrast pattern mining, such as growth rate, odds ratio, and gain [13, 14]. These measures describe whether the contrast patterns are relevant enough to the groups of interest. The simplest methods evaluate the overall quality of the contrast patterns by computing the average value of contrast measurements. Two other metrics proposed by Q. Liu, et al [15] calculate the overlap of the data instances to represent the diversity of the pattern set.

2.3 Related Work

2.3.1 Measurement of contrast patterns

There are several metrics to evaluate the relevance of contrast patterns. The first one is *confidence*, which is widely used in finding association rules by estimating the probability of the pattern appearing in each group. In addition, *support difference* is also used to evaluate the absolute difference of the frequency of item appears in each group. Moreover, *growth rate* is the support ratio of the two groups. *Length* simply represents the number of items in the contrast patterns. Also, there are some other measurements to attempt to quantify the value of the contrast patterns. For instance, *Chi-Square* is often used to calculate the significance of the contrast patterns.

2.3.2 Methods to identify groups

In the traditional subgroup mining method, the form of the subgroup is $X \Rightarrow Y$ (target class), where X is the conjunction of attributes and Y is the target class. The main goal of subgroup mining is to perform descriptive analysis and classification. A good subgroup should be easy to interpret. Besides, it also needs to contain as many transactions as possible to avoid being mistaken for noises. The main challenges of subgroup discovery are converting continuous features into classes, selecting features, designing the quality function and post-processing of the rules. Apriori-C algorithm is proposed by V. Jovanoski, et.al [16] and this algorithm is an improved version of the Apriori algorithm. The advantage of this algorithm is that it removes some itemsets and useless rules to reduce the search space. Due to the large space of the Apriori, it uses the

Odds Ratio or other statistical methods to select the subset of features first. The main problem of many subgroup discovery methods is that too many rules are generated. However, Apriori-C recommends the 10-15 highest-quality rules. It uses two association rules to classify the classes. One is N-best rules, and the other is N-best rules for each class. The later one is good for imbalance data. Klösgen and May [17] introduced SubgroupMiner which uses beam search and quality function to rank the rules. The classical binomial test is used to verify the difference in the entire population. Subgroup suppression is used to eliminate redundant subgroups. An expert is heavily involved in the selection of subgroups. Another method called CN2-SD was introduced by N. Lavra, et al [18] who used two weighted covering algorithms to weight the association rules. B. Kavšek, et al [19] developed Apriori-SD method based on Apriori-C method. It improves the covering algorithm, and the voting mechanism is invoked during the classification process. These subgroup discovery methods are good at finding the subgroups with target class label. However, they were considered “supervised” versions of subgroup discovery which may not be as powerful as “unsupervised” approach to automatically identify subgroups that have highly contrasted patterns.

2.3.3 Methods to rank groups

It is important to rank the top k interesting subgroups after identifying subgroups. Apriori-C algorithm ranks the found subgroups based on the rule covering and recommends 10-15 useful subgroups for users. SubgroupMiner uses a quality function to rank groups and an expert is involved in selecting the meaningful and interesting rules. It

also uses the subgroup suppression method during the searching procedure to filter redundant rules. $wWRAcc'$ is employed in Apriori-SD to evaluate the subgroups. Also, Stefan et al. [20] proposes an approach based on their work on ranking SVM.

In conclusion, one goal of these subgroup discovery algorithms is to find the combination of attributes X to describe the target subgroup Y . Thus, they use a rule covering method to evaluate how many instances a rule $X \Rightarrow Y$ contain. The other goal of these subgroup discovery algorithms is to use the combination of attributes X to classify the target class Y . Thus, the classification measurements are involved in this process. However, these ranking measurements focus on the number on the instances a subgroup contain and the classification accuracy of the rule. The measurement of the rule is supervised with the target label Y . These methods don't give the difference measurement between subgroups in an unsupervised way.

2.3.4 Methods to do exploratory mining

Exploratory data analysis (EDA) can describe the data in a general way instead of formal modeling or hypothesis testing task. EDA does not require the user to make assumptions and instead utilizes observed facts to give the direction for further data collection and research. The function of the EDA in the data mining is shown in Figure 2.

John T. Behrens gives an introduction and guide for effective Exploratory data analysis [21]. The exploratory data analysis is derived from statistics and has recently been applied to data mining and Big Data area [22, 23].

Our approach is an exploratory analysis. It doesn't have a target class label at the

beginning. The mining groups can give a direction for further study to find the deep reasons for the group differences.

Figure 2. Flowchart of Exploratory analysis mining.

3. Expert-Guided Feature Partition

3.1 Terminology

In our method, we assume that all the datasets are in the form of a relational table as shown in Table 1. A relational table has several features P and each feature has multiple categories C . The features can be divided into two categories. One is population features, meaning our method can use these features to form the subgroups. The other is pattern features. These features are used to describe the main characteristics of the subgroups. During the mining process, depending on the application domains, some pattern features might be considered as population features.

To form the subgroups, we only consider the population features. For each population feature P_i , we extract its values: any two values can form a pair $(P_i \rightarrow C_{im}, P_i \rightarrow C_{in})$. We can take these two subpopulations as a contrasting pair. The first item $P_i \rightarrow C_{im}$ describes one population which has the trait C_{im} and the second item $P_i \rightarrow C_{in}$ describes the other population with the trait C_{in} . By adding a new category $(P_j \rightarrow C_{jm}, P_j \rightarrow C_{jn})$ to the previous pair, it forms a new population pair $(P_i \rightarrow C_{im} \wedge P_j \rightarrow C_{jm}, P_i \rightarrow C_{in} \wedge P_j \rightarrow C_{jn})$. The two elements in this tuple each describe one of the new populations. By repeatedly adding or removing categories in the tuple, we can get various combinations, where each combination represents a different population pair to compare $(P_i \rightarrow C_{im} \wedge P_j \rightarrow C_{jm} \wedge \dots, P_i \rightarrow C_{in} \wedge P_j \rightarrow C_{jn} \wedge \dots)$.

Definition: The population pair in the form $(P_i \rightarrow C_{im} \wedge \dots \wedge P_k \rightarrow C_{km}, P_i \rightarrow C_{in} \wedge \dots \wedge P_k \rightarrow C_{kn})$,

where each feature P_k is a population feature and C_{kn} is a category of P_k . The population pair must satisfy these conditions:

- (a) the features in the first and second item are the same.
- (b) there are no two same features in the first item; nor in the second item.
- (c) these two populations must have at least one common base. i.e. there exists $P_i \rightarrow C_{ij}$ appearing in both populations.

Condition (a) guarantees that the two populations are comparable. Condition (b) will prevent there from being any overlap of certain population features. For example, when P represents “gender”, we assume a person cannot be both male and female. To make the comparison between two populations more meaningful, our algorithm requires that the population pair must have one feature and category in common, which is stated in Condition (c). Note that we only consider the contrast between two populations in this research. To compare a group of populations, one can expand this idea from pairs to n-tuples.

<i>Population features</i>			<i>Pattern features</i>		
Feature 1	Feature 2		Feature k		Feature n
$C_{1,1}$	$C_{2,1}$		$C_{k,1}$		$C_{N,1}$
$C_{1,2}$	$C_{2,2}$		$C_{k,2}$		$C_{N,2}$
...	...		...		...
$C_{1,K1}$	$C_{2,K2}$		$C_{k,Kk}$		$C_{N,KN}$

Table 1. Relation table of features and categories.

3.2 Expert-Guided

In traditional subgroup mining, a single feature is often assigned as a target class. This target class is what the user is interested in, or what the user wants to classify. However, in our model, we put multiple features as a target instead of single one. The challenge is deciding which features are target patterns and which are population features. Some experts are interested in certain subgroups, so the features for these subgroups must be included in the mining process. For example, experts might be interested in comparing the differences between young people and old people. In this case, the subgroup is in the form $(Age \rightarrow young \wedge \dots, Age \rightarrow old \wedge \dots)$.

- Division of population features and pattern features.

In our model, we invite experts and users to decide what they want to examine within the dataset. For example, if they want to understand how smoking habits affect blood pressure, we will include the blood pressure as part of pattern features and smoking habits as part of population features which provide a basis for our exploratory mining process. In general, we will put the inherent subgroup characteristics as population features and other information of interest as pattern features.

- Modifiable features and Unmodifiable features.

Domain experts should decide which features are modifiable features and which are not. Features are labelled as unmodifiable when the experts decide that they must be included as part of the subgroup definition. Modifiable features mean that the algorithm has the options to include or exclude those features.

4. A NEW WORKFLOW OF DATA ANALYSIS

Different population subgroups often have different patterns of attributes that represent the intrinsic nature of the groups. The differences in the patterns between the groups can be used to evaluate the significance of the mining results. Some population clusters may have trivial contrast patterns, such as extremely sick patients having a much higher cost of care. In this example, the relationship between sickness and cost is a contrast pattern. However, this example is a well-known relationship, and we are particularly interested in novel contrast patterns are not obvious and which contain new knowledge that gives hints for further interventions.

It is computationally challenging to perform an exhaustive search for the top k interesting population clusters. A research dataset about the combined effect of smoking tobacco and drinking alcohol was published by Hart CL, et al [24]. In his research, the people who both smoke and drink have higher risk of death than other populations, as shown in Figure 3. However, when examining subgroups of this dataset that have different diseases, drinking and smoking habits have a different effect on the death rate. *“Relative rates for CHD mortality were high for current smokers, with a possible protective effect of some alcohol consumption in never smokers. Stroke mortality increased with both smoking and alcohol consumption.”* Thus, if the researchers try to consider more diseases into the study, the number of disease subgroups will increase. In addition, if the researchers also consider the impact of other habits, the feature space is getting larger. Due to the large number of features and categories, their combination space is huge. It is time-consuming

to identify the subgroups manually or just depend on statistical methods.

Figure 3. An Example of Combination of Smoking and Alcohol.

The architecture of our exploratory data analysis is depicted in Figure 4. First, our method categorizes data into discrete values. Then experts are asked to classify features into population features and pattern features. If the experts have target subgroups or wish to require the inclusion of some features, they can define them in the “Prioritization of population grouping” step. After preprocessing the data, we use a sequential floating forward selection approach [25] to choose categories from population features and form the population pairs. Each population pair represents a contrast subgroup. Then we use the J function to evaluate each subgroup and pick the subgroup with highest J value. Here, the J function is used to evaluate the quality of the contrast group by examining their contrast patterns. After picking a contrast subgroup, we alternate executing the inclusion and exclusion steps. In the inclusion step, we add a new category with the

highest contribution to the contrast subgroup. In the exclusion step, we drop a category with the lowest contribution to the contrast subgroup. At any point, if the subgroup satisfies the stopping criterion, the search algorithm stops. The stop criterion is defined using the rate of growth of the J function. Throughout the whole process, we track the cluster pairs with high J values for each floating step and use a depth-first search method to thoroughly explore promising paths. Finally, we provide the top k highest scoring pairs for closer examination.

The methods section describes the whole process of the approach. It is divided into four sections. 4.1 presents the data preprocessing and data storage using SparkSQL; 4.2 introduces the Inclusion and Exclusion Process Using SFFS. 4.3 describes the use of the distributed Apriori algorithm to mine the frequent items and evaluation method of contrast patterns. 4.4 presents the path-tracking method of our algorithm.

Figure 4. The architecture of finding clusters.

4.1 Data Preprocessing

We use a Big Data ecosystem to store and process the data. This allows for scalability when processing data of various sizes. The majority of data storage utilizes the Hadoop Distributed File System (HDFS) [26] and data processing and calculations occur in Apache Spark [27]. When the raw dataset is small, it can also be stored locally and read into Spark later, avoiding the need for HDFS. Our use of a floating algorithm for incremental feature selection requires the frequent repartitioning of the data. To avoid the inconvenience of splitting data, we use the SparkSQL to split them. SparkSQL [28] is Apache Spark's module for managing structured data. SparkSQL can query structured data inside Spark programs, using either SQL or a common DataFrame API.

In our application, the dataset contains multiple features which can be divided into two types. One is the population features $P_1, P_2, \dots, P_n$ which defines the groups of people. The other is pattern features $I_1, I_2, \dots, I_n$ as we previously introduced. We assume all the features in the dataset only contain categorical attributes. If not, it is necessary to discretize any continuous values into categories. One simple discretization method is to divide the possible range into the several buckets using a uniform interval. For example, the feature X is in the range $[x_{\min}, x_{\max}]$ and the user wants to divide the feature into k buckets. Then the discrete values are in the form of $[x_{\min}, x_{\min} + d], [x_{\min}, x_{\min} + 2d], \dots, [x_{\max} - d, x_{\max}]$, where $d = (x_{\max} - x_{\min})/k$. This method is called the equal-width method. Another method to discretize the continuous numbers is called the equal-density method. This force buckets to have a similar number of rows.

Finally, the entropy-based binning method is also introduced by U. M. Fayyad and K. B. Irani [29]. D. Gamberger and N. Lavrac [6] gives another method to discretize the continuous attributes. For an integer P_i , features of the form $P_i = C_{in}$ and $P_i = C_{im}$ are generated. A continuous feature P_i can discretized based on adjacent pairs. They create the form $P_i \leq (C_{in} + D_{im}) / 2$ for all adjacent value pairs (C_{in}, D_{im}) , and $P_i > (C_{in} + D_{im}) / 2$ for all adjacent pairs (D_{im}, C_{in}) . In our situation, we use the following discretization method.

If the feature P_i has a well-known scale, we usually categorize the feature by the default scale. For example, when considering age, the World Health Organization (WHO) recommends dividing into three groups: 5-17 years old, 18-64 years old and 65 years old and above. Blood pressure literature also has a chart to divide the numeric value into different categorical levels. We often discretion these features by their default method.

If the feature P_i does not have a well-known scale, there are a couple of other categorization methods. The simplest method is discretizing the feature into two categories based on a comparison to the median value. Any values above the median are considered high values and those below the median are considered low values. Another method is to analyze the distribution of the feature values. If they are distributed equally, we use the equal-width method. If they are distributed unequally, we use the equal-density method instead. Integer value attributes P_i are of the form $P_i = C_{in}$ and $P_i = C_{im}$.

4.2 Inclusion and Exclusion Process Using SFFS.

4.2.1 SFS and SBS feature selection algorithm.

A feature selection algorithm is used to select a subset of features d from original feature sets D , where $d \subseteq D$. Thus, feature selection will reduce the search space and avoid the exhaustive search. The naïve method of selecting d features is to evaluate each feature and then select top d best features. The drawback of this approach is that it does not consider the dependency between features. There are two nominal methods in this area. One is SBS and the other is SFS, both of which are greedy algorithms. The Sequential Backward Selection (SBS) was proposed by Marill and Green [30] and Sequential Forward Selection (SFS) was proposed by Whitney [31]. They are similar, though one is a bottom-up approach (SFS) and the other is a top-down approach (SBS). The SBS algorithm includes all the available features at the beginning, and low-impact features are iteratively dropped. The SFS starts with an empty set and adds a high-impact feature at each step. Both use a criterion function to determine the feature to add or remove. The search process of SFS is shown in Figure 5. It continues adding new features until the number of features satisfy the requirement. However, recently developed algorithms are less likely to fall into local optima. Plus-L minus-R selection (LRS) minimizes the drawback of the sequential feature selection algorithm [32]. If $L > R$, the algorithm repeatedly adds L features first and then drops R features. If $L < R$, the algorithm repeatedly adds R features first and then drops L features. The main drawback of this search algorithm is that the user must define two parameters. An even more recently developed floating selection algorithm will also avoid local optimal by adding and

removing features and improves the LRS algorithm by reducing the number of required parameters.

Figure 5. The process of SFS.

4.2.2 Floating algorithm.

SFS (SBS) is a greedy feature-selection algorithm, as it keeps moving forward by adding (or removing) new features. The main drawback of SFS (SBS) lies in the addition (or removal) of features that appear to be beneficial selections, but are no longer optimal once future additions (or removals) are known. To avoid this local optimal drawback, we use Sequential Floating Forward Selection (SFFS), a floating feature selection approach based on traditional SFS algorithm. It revises suboptimal choices by occasionally removing features [25]. The state-of-the-art of SFFS algorithms review the included features after each inclusion step, and if the criterion function J increases after excluding some features, then it drops those features. The advantage of the SFFS algorithm is shown in Figure 6. In Step 2, Feature 1, Feature 2 and Feature 3 are selected because the

combination of them have a high J value. After the selecting and removing part, the final selected features are Feature 1, Feature 3 and Feature 4.

Although SFFS is still greedy and the result is locally optimal, it reduces the time needed to search a huge space with a closer to true optimal solution than SFS. The advantage of the SFFS is that it achieves a balance between efficiency and finding a closer to globally optimal solution. The pseudo code of the SFFS Algorithm is described in Figure 8.

Figure 6. The process of SFFS.

In the beginning, the algorithm chooses two features as a base using SFS. Then the main structure of the algorithm is divided into three steps. The algorithm chooses the feature with the highest J score to add it, as described in the Inclusion Function in Figure 7. Then, the algorithm may remove the previously selected feature with the lowest J score if the J score falls below a threshold, as described in the Exclusion Function in Figure 7. If there are more features with J scores below the threshold, then more low-scoring features will be dropped. This continued dropping step is called the Continued

Exclusion step. The whole process repeats until the stop criterion is achieved. For each feature in the remaining feature set, category pairs are formed. For each pair, Line 4 of the Inclusion Function represents adding a new feature to the previous pair and using it as part of the description of the contrast group. Lines 5-9 of Algorithm 1 chooses two population features as a base. To evaluate the current state, SFFS applies the J function to calculate the current J value. Line 5 of the inclusion step represents the process of splitting data by SparkSQL. The full raw data is divided into two groups to compare. Some researchers call the resulting groups the positive class and the negative class. Line 6 represents running the contrast algorithm on these two classes. In this step, the contrast mining algorithm will identify the patterns that distinguish the two classes. We designed a criterion function J that uses these mined patterns to evaluate the impact of the group division. Lines 9-11 choose the best one among the available features and the algorithm adds the best category pair of that new feature into the previous pair set. Finally, it deletes the selected feature from the remaining feature set. Similarly, the SFFS chooses a feature to remove when the exclusion would not decrease the J value. Lines 1-7 of the exclusion function represents the step of choosing the feature to exclude. Line 8 describes the mining process of contrast patterns and calculation of the new J value. If the category pair is the worst one among those old features, the algorithm will record that old feature and drop it when the following situation is satisfied: the cardinality of the selected feature set must be larger than or equal to two (the base number of features). Since the J value doesn't decrease with the removal of this feature, it won't have a negative impact to the current feature sets.

Function Inclusion
1: for fea in Remain_fea 2: Compose pairs of fea's category list $List(C_1, C_2, \dots, C_n)$ 3: for p in paris 4: tempMap=SelectMap+ ($fea \rightarrow p$) 5: Separate dataset based on tempMap to form two population clusters(P_1, P_2) 6: $J(tempMap)=contrast_mining(P_1, P_2)$ 7: end 8: end 9: Select bestfea with highest J value 10: Select_fea+= bestfea 11: SelectMap+= ($bestfea \rightarrow p$) 12: Remain_fea-=bestfea
Function Exclusion
1: for fea in Select_fea 2: Compose <i>pairs</i> of fea's category list $List(C_1, C_2, \dots, C_n)$ 3: for p in paris 4: tempMap=SelectMap- ($fea \rightarrow p$) 5: Separate dataset based on tempMap to form two population clusters(P_1, P_2) 6: end 7: end 8: $J(tempMap)=contrast_mining(P_1, P_2)$ 9: Select <i>worstfea</i> with the highest J value

Figure 7. The Pseudo code for Inclusion and Exclusion.

We have applied the advantage of the SFFS algorithm to our process of selecting subgroups. The whole process of the subgroup formation is described in the Algorithm 1 module. Lines 5-9 indicate the use of the inclusion process to select the base for later floating feature selection process. We can simply explain why two features are initially chosen as a base. If it picks only one feature P_i with the highest J value and put it into the selected_fature= $\{P_i\}$, we know that feature P_i is the best one in the feature group

$\{P_1, P_2, \dots, P_n\}$. When there is only one feature in the selected_feature = $\{P_i\}$, it cannot be dropped. Otherwise, the remaining selected_feature set is the empty set $\{\}$, which is not interesting to the user. If you pick two features P_i and P_j into the selected_feature = $\{P_i, P_j\}$ and then initiate a dropping step, dropping P_j would leave only P_i in the selected_feature. P_j was chosen since it was the best feature to add at that point, so the algorithm will not be able to find a better P_k that is better than P_j to form the selected_feature set with a cardinality of two. In this case, the algorithm would lock into the process of repeatedly adding P_j and then dropping it. We do not want to allow this situation, so we start the adding and dropping process from the two-feature base. Line 10 indicates the stop criterion of the floating sequential algorithm. In the original version of SFFS, the user determines the number of the features that must be present when the algorithm stops. Here we have a flexible algorithm that will determine when to stop. Also, if the remaining feature set is empty, the algorithm also stops. Lines 11-12 of Algorithm 1 call the inclusion function. Here the algorithm tries to add a new category pair into the previous one. Line 14 calls the feature exclusion process. Line 15-17 determines whether the worst feature is the one just added before or not. If the feature to be dropped is the same as the added one before, the algorithm will not remove this bad feature. This is because dropping it would cause the algorithm to return to step 1, then add this feature again. The algorithm repeatedly adds and removes the same feature indefinitely. Alternatively, if the feature to be dropped is not the one just added, it goes to the continue dropping step. Line 20-24 of Algorithm 1 keeps dropping the harmful features in the same

way as step 2 until removing more features will not increase the J value. This whole process is continued until some criterion is not satisfied.

Algorithm 1 Population Cluster Searching

```

1:   Input dataset and create SparkSQL DataFrame
2:    $PList = \{(P_1, (C_{11}, \dots, C_{1n})), (P_2, (C_{21}, \dots, C_{2n})), \dots, (P_n, (C_{n1}, \dots, C_{nn}))\}$ 
3:    $Remain\_fea = \{P_1, P_2, \dots, P_n\}; Select\_fea = null;$ 
4:    $SelectMap = null; k = 0;$ 
5:   while  $k < 2$ 
6:     inclusion:
7:       Call  $Inclusion(Remain\_fea, Select\_fea, SelectMap, PList)$ 
8:        $k = k + 1$ 
9:   end
10:  while  $(J(k) - J(k-1)) / J(k) > \alpha$  and  $Remain\_fea \neq null$ 
11:    step1 inclusion:
12:      Call  $Inclusion(Remain\_fea, Select\_fea, SelectMap, PList)$ 
13:    step2 exclusion:
14:      Call  $Exclusion(Select\_fea, PList)$ 
15:      if  $(worstfea == bestfea)$  then
16:         $k = k + 1$ 
17:         $J(k) = SelectMap' J \text{ value}$ 
18:      else
19:        step3 continue exclusion:
20:          Call  $Exclusion(Select\_fea, PList)$ 
21:          if  $(worstfea == bestfea)$  then
22:             $k = k + 1$ 
23:             $J(k) = SelectMap' J \text{ value}$ 
24:          else repeat step3
25:  end

```

Figure 8. The pseudo code of Algorithm 1.

In our research, our aim is to find the population pairs with high J values. We need to mine contrast patterns between the population pair and define an appropriate J function to evaluate the pair.

4.3 Distributed Apriori algorithm and contrast pattern mining.

To get the contrast patterns, the method finds the frequent patterns in the two classes first. The Apriori algorithm is a widely used frequent pattern mining method. It is easy to implement in the distributed system because of the repetitiveness of the process. The Apriori algorithm utilizes an iterative candidate generation, a Cartesian product, to grow itemsets of increasingly large size. For each iteration k , our goal is to construct the $(k + 1)$ -itemsets using the frequent k -itemsets. This is accomplished by performing a full pairwise comparison between the k -itemsets. This poses a challenge in a distributed computing environment, as k -itemsets may exist on different physical machines, and thus, data must be shuffled between machines to complete the Cartesian product. Utilizing the Cartesian Scheduler, this candidate generation step can be executed on the Apache Spark framework [33].

First, raw data is processed and converted into $\langle \text{item}, \text{transactionSet} \rangle$ key-value pairs. Each key-value pair has a unique item and for each item i the transactionSet tracks which transactions contained i . Next, this collection of key-value pairs is passed through the Cartesian Scheduler. An additional function is passed along with the key-value pairs, defining the comparison to be performed between each data element. Each comparison between two k -itemsets begins with the union of the k -itemsets. If the union yields an

itemset with $k+1$ items, it is a candidate $(k+1)$ -itemset. The next step of each comparison is to evaluate the support for each $(k+1)$ -itemset by considering the intersection between the k -itemsets' transactionSets. The itemsets that pass the minimum support threshold are frequent $(k+1)$ -itemsets. This process is repeated until the candidate generation step yields no larger itemsets.

When Apriori is performed on two distinct populations or collections of items, a post-processing contrast mining analysis may be conducted to compare the two sets of results. Itemsets with support values that differ greatly between populations are considered contrast sets. The process involves performing a groupByKey operation to bring together the transaction sets for an itemset from the different populations. The result is the following $\langle \text{itemset}, [\text{transactionSet1}, \text{transactionSet2}] \rangle$ where transactionSet1 and transactionSet2 were acquired from population1 and population 2, respectively. This format simplifies the process of performing arbitrary contrast metrics. The metrics of contrast patterns are discussed in the next section.

4.4 Criterion Function Design.

4.4.1 Metrics of Contrast Patterns.

There are lots of metrics to measure the impact of contrast patterns. Assume we have two groups called D_1 and D_2 . Both contain some transactions respectively [34].

- **Confidence:** This is used to predict the probability of Item I appearing in each class. Higher confidence indicates a higher probability that item I belongs to class D_1 .

$$Confidence = count(I, D_1) / count(I, D_1 \cup D_2)$$

- Support Difference: This is the absolute difference between the supports of D_1, D_2 .

The larger support difference means the frequency differs between two groups. It can be used to evaluate differences of patterns in two classes.

$$Support_Difference = |Support(I, D_1) - Support(I, D_2)|.$$

- Growth: This measures the support ratio of an item I between the two groups. The larger growth is preferable in contrast mining. The range of the growth is $[0, Infinity)$.

$$Growth = Support(I, D_1) / Support(I, D_2).$$

- Length: This simply measures the number of items contained in the contrast pattern.

$$Length(I, D_1, D_2) = |I|$$

These above methods each only measure one certain pattern at a time. However, if two groups have many different patterns, it is still an open question to determine how to evaluate them in aggregate. There is some work that attempts to assess the diversity of the pattern set. The simplest method to calculate the overall contrast between two classes is to calculate the average of any of the above metrics. However, this approach does not consider the number of contrast patterns. Some groups contain many contrast patterns which mean the differences between these two classes is large. Instead of calculating the average metric values, another method evaluates the diversity of the contrast patterns. Two metrics of evaluating the overall contrast patterns are proposed in [35]. A higher diversity of contrast patterns indicates a larger difference between the classes. One metric measures the overlap of the items between several contrast patterns, a lower value means the overlap of contrast items is low. This corresponds to a high diversity of contrast

patterns. The other metric is similar to the previous one, but it measures the overlap of records the items contains. Similarly, a lower value indicates a higher diversity of the patterns, which is more favorable to the overall evaluation.

4.4.2 Good contrast patterns selection.

To assess the overall difference of those contrast patterns, we cannot use all of them to determine the quality of the contrast groups. We need a metric that can be quickly applied to calculate the contrast between pattern sets repeatedly. Some patterns are the subset of other patterns and their support in the two groups is equal. An example of the contrast patterns is shown in Table 2. These patterns were used to calculate the statistics shown in Table 3. From Table 2, we know that the patterns $\{A, E\}, \{C, E\}$ are the subsets of $\{A, C, E\}$ and their supports are the same (75%), meaning that $\{A, E\}, \{C, E\}, \{A, C, E\}$ contain the same transactions as well. Calculating the growth of these three patterns would be redundant. Here, we give the definition of the subset of the contrast pattern.

Definition: For a contrast pattern set V , if there doesn't exist a contrast pattern set U which is a superset of a contrast pattern set V , i.e. $V \subseteq U$ and the support of V in group 1 is near the support of U in group 1, so is in group 2. i.e. $Support_{group1}(V) - Support_{group1}(U) \leq \varepsilon$ and $Support_{group2}(V) - Support_{group2}(U) \leq \varepsilon$. We call the contrast pattern set V as a closed contrast pattern.

In our architecture, we only consider the closed contrast pattern. The user determines the parameter ε . The larger ε is, the smaller the closed contrast pattern

is. Lower values of ε will include more patterns to evaluate the overall quality of contrast.

	I1	I2	I3
Group1_trans1	A	C	E
Group1_trans2	A	C	E
Group1_trans3	A	C	E
Group1_trans4	A	C	F
Group2_trans1	B	D	F
Group2_trans2	B	D	F
Group2_trans3	B	D	F
Group2_trans4	B	D	E

Table 2. The Transactions of two groups.

	Support in Group1	Support in Group2	Growth
{A, C, E}	75%	0%	Infinity
{A, C}	100%	0%	Infinity
{A, E}	75%	0%	Infinity
{C, E}	75%	0%	Infinity
{A}	100%	0%	Infinity
{C}	100%	0%	Infinity
{E}	75%	0%	Infinity
{B, D, F}	0%	75%	Infinity
...	...	...	...

Table 3. Contrast Pattern of two groups.

4.4.3 J functions design.

Different population pairs have different quantities of contrast patterns.

Furthermore, some contrast patterns have a very high support difference between the population pair; others do not. The job can be abstracted into the problem of maintaining a balance between the quantity and quality of the contrast patterns. Our goal of this thesis is to find the “most interesting” contrast groups. To evaluate the overall contrast patterns between two groups, we need to design a proper J function to use in the SFFS algorithm. The idea of this work is similar to the process of evaluating the reputation of a scholar who has various publications. We cannot say the scholar is a good scholar if he publishes many papers which get very few citations. Meanwhile, it is also hard to say that the scholar who only publishes one very high cited paper is a good scholar. This problem is not a new issue in the scholar evaluating the area. There are many methods to assess the impact of a scholar in the research area and many factors have the influences of assessing a scholar, such as age, patents, reputation and the area the scholar is in. However, the most ways to evaluate a good scholar involves considering their number of publications and their quality. Jorge [36] proposed a widely used metric called the h-index to measure the contributions of a scholar. H-index is defined as a such: a scientist has index h if h of his or her N papers have at least h citations each, and the other $(N-h)$ papers have no more than h citations each. Although h-index is widely used, the criticism of h-index never settles. The main drawback of the h-index is that it does not consider the quality of the papers. For example, a scientist published an excellent paper, and the citation is about 1000, but he has only published this only one paper. The h-index of that scientist is only 1. The disadvantage of the h-index is easy to observe in this case. The M-index is proposed [37] to deal with the situation when the distribution of citation counts is skewed. It argues that the median and

arithmetic average should be used as the measure of the central tendency. Besides these, there are also some other indexes to evaluate the scholar. Google introduced the i10-index in Google Scholar. It only counts the number of publications with at least ten citations. The advantage of the i10-index is it is simple and easy to calculate. However, it is only used by Google, and it is hard to justify the choice of ten for the threshold. There are also some other metrics to evaluate the scholars, such as o-index which calculates the geometric mean of the h-index and the most cited paper of a researcher [38]. Also, the e-index [39, 40], c-index [41], and s-index [42] are also introduced to address the weakness of the h-index.

To evaluate the overall quality of contrast patterns, we use the quantitative indicator g index [43]. The g-Index was originally used to quantify productivity based on publication record. Given a set of articles ranked in decreasing order of the number of citations that they received, the g-index is the unique largest number such that the top g articles cumulatively received at least g^2 citations. It avoids the drawback of h index which only considers the quantity regardless of the quality. A comparison between h-index and g-index are shown in Table 4. In our method, we use g-index as our J function. To evaluate the contrast degree of the population pair, we consider the number of contrast patterns and the quality of these patterns, an easy and efficient way is to calculate the growth of contrast patterns and then sort them in a decreasing order based on growth. We take the growth as the value to calculate g index. For example, if the $\text{growth}(x)=5$, the value of the x pattern is 5. Also, the growth can be the infinity value. To transfer the Infinity value into the real number for calculating the g-index. We convert values of infinity to a large number

beyond the normal value. For each population pair, we calculate its J value. Line 9 in the inclusion function represents the iterative calculation of the J function and choosing the high J values as candidates. We store the contrast pair with “high” J value during each iteration.

Patterns	Growth	
	1st Groups	2 nd Groups
1	51	6
2	34	5
3	29	4
4	22	4
5	3	3
6	1	3
7	0	2
8	-	2
9	-	1
10	-	0
11	-	0
H-index	4	4
G-index	11	4

Table 4. A comparison of h-index and g-index.

4.5 Tracking Multiple Paths.

We use Depth-first search to get top k valuable results. Assume the start point is a node and we call it root node. Run the SFFS algorithm from the beginning; the algorithm will give back multiple population pairs. Sort the pairs in decreasing order. We set the first

category pair k_1 's value as a base. The candidate pairs are the ones satisfy the condition $k_1 - k_n < threshold$, indicating that this candidate pairs' J value is near the top one k_1 . For each candidate pair, we add it to the root node's population cluster pair to form the new cluster pair. The algorithm takes these good pairs as children nodes of the root and adds them to the root's children list. Then the algorithm starts from the first child and continues running the inclusion and exclusion step of the SFFS algorithm. The algorithm does the same job for each iteration. When the child node's value is equal or less than its parent's value, this means the value will not increase anymore, so this branch is terminated. Then the algorithm traces back to the parent node to continue the job. The whole process here is a Depth-first search; it is shown in Figure 9. After traversing all the possible paths, the algorithm stops. All the branches are the paths to the various results. The leaf nodes store the final population cluster pair and its J value. We display the results and sort their J value in decreasing order. If two population cluster pairs' values are the same, we prioritize the pair with a smaller category length because having more categories different between the two populations will usually result in more contrast patterns.

4.6 Post-processing of selecting good contrast groups.

Usually, the floating algorithm results in many good contrast groups to provide to the user. However, some contrast groups with high scores could be "trivial" groups that could be easily predicted by the expert and not of interest. To pick up "good" and "significant" groups from the candidate pool, we need to involve the expert into the system.

Figure 9. A travel path of candidate selection.

5. EXPERIMENTAL RESULTS

5.1 Experimental Design

We designed experiments to demonstrate the ability of this exploratory method. Because there is no existing method to solve the contrast group exploratory mining, performance comparisons between our results and others' work on common data sets are not possible. Thus, we test our methods on synthesized data and real data with the following three main goals:

1. Compare the search time needed as the search space increases. At the same time, we will compare the search time needed for the SFS and SFFS algorithms.
2. Create some subgroups intentionally while constructing a synthesized data set and test if the algorithm can identify those subgroups.
3. Use exploratory analysis on the real data to see if there exists some useful information and compare the results with others' public work.

5.1.1 Synthesized data creation.

When we create the synthesized data, it should follow some constraints. The data should contain some inherent subgroups, and these subgroups need to contrast, meaning these groups should have many distinguishing patterns. In our synthesized data, we create three contrast subgroups, and we introduce obvious differences into those groups. The synthesized data contains ten population attributes P_1 - P_{10} and nine pattern features P_{11} - P_{19} . The first group is shown in Table 5, the second group is given in Table 6, and the third

group is provided in table 7. Their patterns are largely different.

P1	P2	P3	P4	P11	P12	P13	P14	P15	P16	P17	P18	P19
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	good	good	good	good	Good	good	good	good
high	high	high	good	good	bad	good	good	good	Good	good	good	good
high	high	high	good	good	bad	good	good	good	Good	good	good	good
high	high	high	good	good	bad	good	good	good	Good	good	good	good
low	low	low	bad	bad	good	bad	bad	good	Bad	bad	bad	bad
low	low	low	bad	bad	good	bad	bad	good	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	good	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	good	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad
low	low	low	bad	bad	bad	bad	bad	bad	Bad	bad	bad	bad

Table 5. Synthesized data group 1.

P5	P6	P11	P12	P13	P14	P15	P16	P17	P18	P19
male	male	bad	good	bad	good	good	good	good	bad	good
male	male	bad	good	bad	good	good	good	good	bad	good
male	male	bad	good	bad	good	good	good	good	bad	good
male	male	bad	good	bad	good	good	good	bad	bad	bad
male	male	bad	good	bad	good	good	good	bad	bad	bad
male	male	bad	good	bad	good	good	bad	bad	bad	bad
male	male	bad	good	bad	good	good	bad	bad	bad	bad
male	male	bad	good	bad	good	good	bad	bad	bad	bad
male	male	bad	good	bad	good	good	bad	bad	bad	bad
female	female	good	bad	good	bad	bad	bad	bad	good	bad
female	female	good	bad	good	bad	bad	bad	bad	good	bad
female	female	good	bad	good	bad	bad	bad	bad	good	bad
female	female	good	bad	good	bad	bad	bad	good	good	good
female	female	good	bad	good	bad	bad	bad	good	good	good
female	female	good	bad	good	bad	bad	good	good	good	good
female	female	good	bad	good	bad	bad	good	good	good	good
female	female	good	bad	good	bad	bad	good	good	good	good
female	female	good	bad	good	bad	bad	good	good	good	good
female	female	good	bad	good	bad	bad	good	good	good	good

Table 6. Synthesized data group 2.

P7	P8	P11	P12	P13	P14	P15	P16	P17	P18	P19
suzhe	suzhe	good	bad	good	good	bad	good	good	bad	good
suzhe	suzhe	good	bad	good	good	bad	good	good	bad	good
suzhe	suzhe	good	good	good	good	bad	good	good	bad	good
suzhe	suzhe	good	good	good	good	bad	good	good	bad	good
suzhe	suzhe	good	good	good	good	bad	good	good	good	good
suzhe	suzhe	good	good	good	good	good	good	good	good	good
suzhe	suzhe	good	good	good	good	good	good	good	good	good
suzhe	suzhe	good	good	good	good	good	good	good	good	good
suzhe	suzhe	good	good	good	good	good	good	good	good	good
suzhe	suzhe	good	good	good	good	good	good	good	good	good
xibei	xibei	bad	good	bad	good	good	good	good	good	good
xibei	xibei	bad	good	bad	good	good	good	good	good	good
xibei	xibei	bad	good	bad	good	good	good	good	good	good
xibei	xibei	bad	good	bad	good	good	good	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	bad	good	bad	good	good	bad	good	good	good
xibei	xibei	good	good	good	good	good	good	good	good	good

Table 7. Synthesized data group 3.

5.1.2 Different J functions.

We perform the experiments using different J functions to compare their accuracy and to determine if g-index is better than the h-index and if the g-index is better than only considering the number of the patterns. Table 8 shows that the floating algorithm using g-index is better than both of these other options. We see that the g-index can find two of the three synthesized subgroups. According to this accuracy metric, the h-index is the same as only considering the number of contrast patterns. The h-index could, in fact, be better than only considering the number of contrast patterns when there are more than three subgroups in the data. We would predict that the SFS is usually faster than the floating algorithm because it contains does not contain the dropping steps used in SFFS.

Accuracy %	g-index	h-index	# of patterns
5%	0	1/3	0
10%	1/3	1/3	1/3
15%	2/3	1/3	1/3
20%	2/3	1/3	1/3

Table 8. The accuracy of different index.

Figure 10. Synthesized data running time of different search range of SFFS and SFS

When comparing how quickly the algorithm completes using the different selection algorithms, the floating algorithm is slower than the traditional sequential algorithm. In addition, the search time will increase when expanding the search range. However, the result of floating algorithm is better than the traditional sequential algorithm and

enlarging the search space will improve the performance of the algorithm. For the floating algorithm, the search time of g-index is slower than the h-index. However, the search time of g-index is quicker than the h-index for the traditional sequential algorithm. The reason of this phenomenon is floating algorithm can traverse more paths. The evaluation values are different for different indexes. The g-index searches more paths because the value of them are similar. Thus, the time is longer when search more paths. The running time comparison of the three indexes when using SFFS and SFS is shown in Figure 10.

5.1.3 SFS VS SFFS.

SFS always considers the addition of features and never deletions, so some wrong steps could influence the results of the rest of the algorithm. Here is the comparison of accuracy using the SFS selection algorithm and the three indexes. From Table 9, we see that even we use the g-index, the accuracy of finding the subgroups is lower than the floating algorithm (Table 8), since SFS can only find one of the synthesized subgroups.

Accuracy %	g-index	h-index	# of patterns
5%	0	1/3	0
10%	1/3	1/3	1/3
15%	1/3	1/3	1/3
20%	1/3	1/3	1/3

Table 9. Accuracy of SFS

5.1.4 Keep multiple results.

In the experiment, we track a different range of branches: the top 5%, 10%, 15%, 20%

and 25% highest-scoring groups. The higher the value is, the more branches are tracked. After the algorithm completes, we sort the results by the g-index score, where a higher score indicates a more valuable result. Table 10 shows the top 5% of groups and their g-index. It did not find the contrast groups we intentionally constructed in the synthesized dataset, but these discovered groups are new and useful. Reexamining the original data, we see that these contrasts are useful even though we were not previously aware of them.

5%	Groups	g-index
1	P1 -> (low,high) P6 -> (male,female)	50.20458
2	P1 -> (low,high) P6 -> (male,female) P3 -> (low,high)	44.45597
3	P4 -> (bad,good) P5 -> (female,male)	38.2372
4	P1 -> (low,high) P5 -> (female,male)	32.86335
5	P1 -> (low,high) P4 -> (bad,good) P5 -> (female,male)	32.53844

Table 10. Top 5 results of g-index in 5% range.

Also, when we track the top 10% of branches, this includes more contrast groups. In addition to the previous contrast subgroups retained in the top 5% of branches, some more new subgroups are also included. The ranked subgroups are listed in Table 11. The 4th group is the group we intentionally constructed, and the 1st, 2nd, 3rd, and 5th groups are all subsets of this group. All desired groups are marked with star on their ID.

Also, when we enlarge the branches the algorithm searches, tracking 15% branches in each step, it includes more desired groups. In the Table 12, the 6th group is what we designed before. Besides, it also includes some groups that are also useful.

In conclusion, when including more branches, it contains more useful groups. However, the searching time increases as we track a larger percent of branches. Thus, it is

best to find a balance between searching time and better result.

10%	Groups	g-index
1*	P1 -> (low,high) P3 -> (low,high) P2 -> (low,high)	46.72169
2*	P1 -> (low,high) P4 -> (bad,good) P2 -> (low,high)	44.54773
3*	P4 -> (bad,good) P2 -> (low,high)	43.75557
4*	P1 -> (low,high) P4 -> (bad,good) P3 -> (low,high) P2 -> (low,high)	40.66633
5*	P1 -> (low,high) P2 -> (low,high)	31.52142

Table 11. Top 5 results of g-index in 10% range.

15%	Groups	g-index
1	P3 -> (high,low) P5 -> (female,male)	45.96194
2	P6 -> (female,male) P3 -> (high,low) P5 -> (female,male)	42.59577
3	P1 -> (low,high) P5 -> (male,male) P2 -> (low,high)	40.47715
4	P5 -> (male,male) P2 -> (low,high)	40.30509
5	P6 -> (male,female) P2 -> (low,high)	39.59798
6*	P6 -> (female,male) P5 -> (female,male)	38.9953

Table 12. Top 6 results of g-index in 15% range.

5.1.5 The footprint of the branch traverse.

The below path tree shows the searching path while tracking the top 15% of branches. It indicates that the floating algorithm can drop some bad features when the result is not that good. However, it still has the problem avoiding repeating a previously taken path. The red paths in Figure 11 represent that the search algorithm has gone through this path before and it is a redundancy.

5.2 Real Data Analysis

5.2.1 Data Source.

The real data examined in this work was obtained from the Kaggle website and the data source is in [44]. The dataset has 13 population features and three pattern features. Those pattern features are the score levels. We select some features to be our population feature and math score features as the contrast features. Population features include Sex, Mother's education level, Father's education level, Study time, Extra school support, Extra paid classes, Willing to take higher education, With a romantic relationship, family relationship condition, Workday alcohol assumption, Absences times, the number of past class failures. Math scores are divided into three-period grades, and we use these as a pattern features. The data has been classified and we discretize the scores into four levels: A, B, C, and D.

5.2.2 Different J functions.

As stated previously, the SFS is expected to be faster than the floating algorithm (SFFS) in general because it does not involve a series of inclusion and exclusion processes that create more branches. For different indexes, the values of contrast group vary. The use of different indexes makes the algorithm stopping at different point. This reason introduces the different running time for different indexes no matter floating or traditional algorithm. In addition, Different groups can have different patterns and the number of contrast patterns was difficult to predict. The different paths the algorithm takes can introduce differences in running time. The running time of SFFS and SFS is shown in Figure 12. It

compares the three indexes. From Figure 12 we can see, SFFS-h-index and SFFS-number take more time on adding and dropping features. Their running time differ largely from their traditional algorithms. SFFS-g-index also takes more time than SFS-g-index, but its running time is not much longer than the traditional one. We can know that SFFS-h-index and SFFS-number walks more paths than SFFS-g-index and those paths could include more repetitive paths.

Figure 12. Real data running time of different search range of SFFS and SFS

5.2.3 Keep multiple results.

Table 13 shows the top 10 groups from the g-index floating algorithm, where it traverses the top 15% of branches. From here we can find some relationships:

1. The students with more failures in the past classes and whose mother's education is

median, even their family relationship is good, their math scores are still worse than the students whose mother's education is high, and who did not fail in past classes.

2. The students whose father's education is high and who have many absences often also have bad grades. However, their grades can be a little bit higher than the students whose father's education is lower.

3. The students who are willing to have a higher education but drinking a lot every day are more likely to get lower scores in the second and final test.

4. The students who take more drinks daily and who did not fail much in past classes are likely to get lower scores in the first test.

5. The students who drink daily and weekly are more likely to fail in the three tests. Their score is more likely to be C or D than those who do not drink.

Group_Id	Groups
1	famrel -> (good,bad) Failures -> (high,low) Medu -> (median,high)
2	Absences -> (low,median) Fedu -> (median,high)
3	Higher_edu -> (yes,yes) Dalc -> (low,high) school_support -> (no,no)
4	Dalc -> (low,high) Failures -> (low,low)
5	Dalc -> (low,high) school_support -> (no,no)
6	Walc -> (low,high) Dalc -> (low,high) Failures -> (low,low)
7	studytime -> (low,low) Dalc -> (low,high) Failures -> (low,low) school_support -> (no,no)
8	Walc -> (low,high) Dalc -> (low,high) school_support -> (no,no)
9	Sex -> (M,M) Fedu -> (median,high)
10	Dalc -> (low,high)

Table 13. Top 10 results of real data.

There are some other results shown on Kaggle from other researchers that also analyzed this problem. Here are two popular conclusions: *“the two strongest predictors of average student grade are the willingness to pursue higher education and mother's education”* and *“The median average grade is visually higher among those students who*

had very low levels of daily alcohol consumption. However, the median grade of the students with a medium, high, and very high levels of daily drinking does not seem to be very different.” Their results are similar to our work. In addition to their conclusions, we found that the students with highly educated mothers tend to have better grades. Also, we believe that the father’s education level and regular alcohol assumption and absences are related to students’ grades. These novel findings demonstrate the power of exploratory mining using our approach.

6. CONCLUSIONS

In this thesis, we identified new research opportunities from the existing method of subgroup discovery, contrast pattern mining, and g-index. The current subgroup discovery methods only mine the class of a per-determined target value. They were not designed to discover subgroups with high contrasts. In our research, we design a semi-automatic architecture for choosing subgroups in the data pool and mining their contrast patterns.

For choosing the subgroups, the sequential floating forward algorithm is applied to choose attributes to form subgroups. This semi-greedy algorithm can achieve better mining results than what the traditional algorithm can do. However, the trade-off is on the longer search time in comparison to the traditional methods. To overcome that, our distributed method is deployed on Spark to make up the drawback of running time.

For the evaluation of the contrast patterns, our innovative approach dose not simply sum up their growth values or use all contrast patterns to assess the contrast group. First, we select those contrast patterns which have high differences. Second, we choose those closed contrast pattern, and this avoids the repeated calculation of their impact. Additionally, the measurement of contrast patterns is used in our approach. We apply a metric typically used in scholastic author judgement to assess our subgroups and rank them by the g-index.

In addition, data analytics results should present multiple options to the users for the creation of actionable plans. In our approach, we design a depth-first search to track multiple paths and rank their relevance to the application domains. More results are

mined by determine a larger search range.

The described experiments demonstrate the power of this new architecture in exploratory mining which is an important utility for the broad data mining and knowledge discovery research community and industry, such as healthcare and E-Commerce.

7. FUTURE WORK

For the future work, we plan to optimize the current algorithm. First, we hope to implement pruning steps during the path searching. As we stated above, some paths are traversed multiple times during the floating procedure. We can store the previously traversed paths in a stack. When other branches approach a previously traversed path, we can force it to stop. This pruning method can save time and search space. Second, reducing the involvement of the specialist is also important. It could be more automatic and accurate for the algorithm to find the principle components and reduce the dimensionality of the data.

Furthermore, we hope to adjust the algorithm to allow temporal mining and spatial mining. For example, when we want to find the differences between two or more time-series, we can use this contrast method. When we want to compare snippets of time and find the most different ones, we can run this exploratory algorithm to guide us. In the spatial application, someone may be curious to compare the structure of the space. For example, comparing protein structures or geoinformation structures can provide more knowledge about their differences.

Lastly, the extension of this algorithm to handle more than two contrast groups is needed to tackle real-world applications

8. REFERENCES

- [1] A. McAfee and E. Brynjolfsson, "Big data: the management revolution," (in eng), *Harv Bus Rev*, vol. 90, no. 10, pp. 60-6, 68, 128, Oct 2012.
- [2] J. Li, H. Liu, J. R. Downing, A. E.-J. Yeoh, and L. Wong, "Simple rules underlying gene expression profiles of more than six subtypes of acute lymphoblastic leukemia (ALL) patients," *Bioinformatics*, vol. 19, no. 1, pp. 71-78, 2003.
- [3] J. Li and L. Wong, "Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns," *Bioinformatics*, vol. 18, no. 10, pp. 1406-1407, 2002.
- [4] J. Li and Q. Yang, "Strong Compound-Risk Factors: Efficient Discovery Through Emerging Patterns and Contrast Sets," *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, no. 5, pp. 544-552, 2007.
- [5] N. Lavrač, B. Cestnik, D. Gamberger, and P. Flach, "Decision Support Through Subgroup Discovery: Three Case Studies and the Lessons Learned," *Machine Learning*, journal article vol. 57, no. 1, pp. 115-143, 2004.
- [6] D. Gamberger and N. Lavrac, "Expert-guided subgroup discovery: methodology and application," *J. Artif. Int. Res.*, vol. 17, no. 1, pp. 501-527, 2002.
- [7] W. Klösgen and M. May, "Census Data Mining – An Application."
- [8] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," presented at the Proceedings of the 20th International Conference on Very Large Data Bases, 1994.
- [9] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *SIGMOD Rec.*, vol. 29, no. 2, pp. 1-12, 2000.
- [10] M. A. Phinney, "Distributed Frequent Hierarchical Pattern Mining for Robust and Efficient Large-Scale Association," Computer Science, University of Missouri-Columbia, 2017.
- [11] S. D. Bay and M. J. Pazzani, "Detecting Group Differences: Mining Contrast Sets," *Data Mining and Knowledge Discovery*, journal article vol. 5, no. 3, pp. 213-246, 2001.
- [12] G. Dong and J. Li, "Efficient mining of emerging patterns: discovering trends and differences," presented at the Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, San Diego, California, USA, 1999.
- [13] J. Li and Q. Yang, "Strong Compound-Risk Factors: Efficient Discovery Through Emerging Patterns and Contrast Sets," *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, pp. 544-552, 2007.
- [14] X. Yin and J. Han, "CPAR: Classification based on Predictive Association Rules," in *Proceedings of the 2003 SIAM International Conference on Data Mining*, pp. 331-335.

- [15] Q. Liu and G. Dong, "A Contrast Pattern Based Clustering Quality Index for Categorical Data," in *2009 Ninth IEEE International Conference on Data Mining*, 2009, pp. 860-865.
- [16] V. Jovanoski and N. Lavrač, "Classification Rule Learning with APRIORI-C," in *Progress in Artificial Intelligence: Knowledge Extraction, Multi-agent Systems, Logic Programming, and Constraint Solving 10th Portuguese Conference on Artificial Intelligence, EPIA 2001 Porto, Portugal, December 17–20, 2001 Proceedings*, P. Brazdil and A. Jorge, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 44-51.
- [17] W. Klösigen and M. May, "Spatial Subgroup Mining Integrated in an Object-Relational Spatial Database," in *Principles of Data Mining and Knowledge Discovery: 6th European Conference, PKDD 2002 Helsinki, Finland, August 19–23, 2002 Proceedings*, T. Elomaa, H. Mannila, and H. Toivonen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 275-286.
- [18] N. Lavra *et al.*, "Subgroup Discovery with CN2-SD," *J. Mach. Learn. Res.*, vol. 5, pp. 153-188, 2004.
- [19] B. Kavšek, N. Lavrač, and V. Jovanoski, "APRIORI-SD: Adapting Association Rule Learning to Subgroup Discovery," in *Advances in Intelligent Data Analysis V: 5th International Symposium on Intelligent Data Analysis, IDA 2003, Berlin, Germany, August 28-30, 2003. Proceedings*, M. R. Berthold, H.-J. Lenz, E. Bradley, R. Kruse, and C. Borgelt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 230-241.
- [20] S. Rueping, "Ranking interesting subgroups," presented at the Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Quebec, Canada, 2009.
- [21] J. T. Behrens, "Principles and procedures of exploratory data analysis," *Psychological Methods*, vol. 2, no. 2, pp. 131-160, 1997.
- [22] N. Andrienko and G. Andrienko, *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag New York, Inc., 2005.
- [23] M. Vega, R. Pardo, E. Barrado, and L. Debán, "Assessment of seasonal and polluting effects on the quality of river water by exploratory data analysis," *Water research*, vol. 32, no. 12, pp. 3581-3592, 1998.
- [24] C. L. Hart, G. Davey Smith, L. Gruer, and G. C. Watt, "The combined effect of smoking tobacco and drinking alcohol on cause-specific mortality: a 30 year cohort study," (in eng), *BMC Public Health*, vol. 10, p. 789, Dec 24 2010.
- [25] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119-1125, 1994/11/01 1994.
- [26] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop Distributed File System," presented at the Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010.
- [27] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," presented at the Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, Boston, MA, 2010.

- [28] M. Armbrust *et al.*, "Spark SQL: Relational Data Processing in Spark," presented at the Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, 2015.
- [29] U. M. Fayyad and K. B. Irani, "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning," in *13th International Joint Conference on Uncertainty in Artificial Intelligence(IJCAI93)*, 1993, pp. 1022-1029.
- [30] T. Marill and D. Green, "On the effectiveness of receptors in recognition systems," *IEEE Transactions on Information Theory*, vol. 9, no. 1, pp. 11-17, 1963.
- [31] A. W. Whitney, "A Direct Method of Nonparametric Measurement Selection," *IEEE Transactions on Computers*, vol. C-20, no. 9, pp. 1100-1103, 1971.
- [32] S. D. Stearns, "On selecting features for pattern classifiers," in *Proceedings of the 3rd International Conference on Pattern Recognition (ICPR 1976)*, 1976, pp. 71-75.
- [33] M. Phinney, S. Lander, M. Spencer, and C. R. Shyu, "Cartesian Operations on Distributed Datasets Using Virtual Partitioning," in *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, 2016, pp. 1-9.
- [34] G. Dong and J. Bailey, *Contrast Data Mining: Concepts, Algorithms, and Applications*. Chapman & Hall/CRC, 2012, p. 434.
- [35] Q. Liu and G. Dong, "CPCQ: Contrast pattern based clustering quality index for categorical data," *Pattern Recognition*, vol. 45, no. 4, pp. 1739-1748, 4// 2012.
- [36] J. E. Hirsch, "An index to quantify an individual's scientific research output," (in eng), *Proc Natl Acad Sci U S A*, vol. 102, no. 46, pp. 16569-72, Nov 15 2005.
- [37] L. Bornmann, R. Mutz, and H.-D. Daniel, "Are there better indices for evaluation purposes than the h index? A comparison of nine different variants of the h index using data from biomedicine."
- [38] S. N. Dorogovtsev and J. F. F. Mendes, "Ranking scientists," *Nat Phys*, Correspondence vol. 11, no. 11, pp. 882-883, 11//print 2015.
- [39] M. V. Dodson, "Citation analysis: Maintenance of h-index and use of e-index," *Biochemical and Biophysical Research Communications*, vol. 387, no. 4, pp. 625-626, 10/2/ 2009.
- [40] C.-T. Zhang, "The e-Index, Complementing the h-Index for Excess Citations," *PLoS ONE*, vol. 4, no. 5, p. e5429, 2009.
- [41] M. Bras-Amorós, J. Domingo-Ferrer, and V. Torra, "A bibliometric index based on the collaboration distance between cited and citing authors," *Journal of Informetrics*, vol. 5, no. 2, pp. 248-264, 2011.
- [42] Z. K. Silagadze, "Citation entropy and research impact estimation," 2009.
- [43] L. Egghe, "Theory and practise of the g-index," *Scientometrics*, journal article vol. 69, no. 1, pp. 131-152, 2006.
- [44] P. Cortez and A. Silva, "Using Data Mining to Predict Secondary School Student Performance," 2008.