FEATURE SUBSET SELECTION PROBLEM ON MICROARRAY DATA

by
NİHAN ÖZŞAMLI

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

SABANCI UNIVERSITY
February 2009

FEATURE SUBSET SELECTION PROBLEM
ON MICROARRAY DATA
APPROVED BY

Assistant Prof Kemal Kılıç ……..

(Thesis Supervisor)

Assoc. Prof. Osman Uğur Sezerman ……..

(Co-advisor)

Assistant Prof. Özgür Gürbüz ……..

Assistant Prof. Gürdal Ertek ……..

Assistant Prof. Tonguç Ünlüyurt ……..

DATE OF APPROVAL: ……

FEATURE SUBSET SELECTION PROBLEM
ON MICROARRAY DATA

Nihan ÖZŞAMLI

MSc Thesis, 2009
Thesis Supervisor: Assist. Prof. Kemal Kılıç
Keywords: feature subset selection, association rule mining, fuzzy logic, pattern classification, gene selection

Abstract

Recent advance of technology gave birth to tools such as microarray chips. The use of microarray chips enabled the scientists to measure the amount of protein production from their genes in a cell, known as the gene expression data. The classification of cell samples by means of their gene expression data is a hot research area. The data used for the analysis is massive and therefore the features, i.e., the genes, must be reduced to a reasonable level due to the computational cost of experiments and the possibility of misleading irrelevant genes. Therefore, usually, the analysis based on the classification of cell samples includes a *feature subset selection* phase. This thesis aims to develop a tool that can be used during the feature subset selection phase of such analyses. Three novel algorithms are proposed for the gene selection problem based on basic association rule mining. The first algorithm starts with fuzzy partitioning of the gene expression data and discovers highly confident IF-THEN rules that enable the classification of sample tissues. The second algorithm search the possible IF-THEN rules based on a heuristic pruning approach which is based on the beam search algorithm. Finally, the third algorithm focuses on the hierarchical information carried through gene expressions by constructing decision trees based on different performance measures. We found satisfactory results in Leukemia Dataset. In addition, in colon cancer dataset, algorithm that is based on construction of decision trees showed good performance.

MICROARRAY VERİSİ ÜZERİNDE
ÖZELLİK ALTKÜMESİ SEÇİMİ PROBLEMİ

Nihan ÖZŞAMLI

Yüksek Lisans Tezi, 2009
Tez danışmanı: Yrd. Doç. Dr. Kemal Kılıç
Anahtar kelimeler: özellik altkümesi seçimi, kural madenciliği, bulanık mantık, patern sınıflandırma

Özet

Teknolojideki son gelişmeler, mikroarray çipleri gibi araçların ortaya çıkmasına önayak olmuştur. Mikroarray çipleri sayesinde bilim insanları, hücredeki genlerden ne kadar miktarda protein üretildiğini ölçme imkanı bulmuşlardır, ölçülen veriler gen ifade verisi olarak adlandırılmaktadır. Gen ifade verisi kullanılarak hücre örneklerinin sınıflandırılması, güncel bir araştırma konusudur. Bu alanda kullanılan veri, çok büyük ölçeklidir; bu nedenle özellikler –genler- sınıflandırma için gerekli ve yeterli sayıya düşürülmelidir. Bu bağlamda, mikroarray gen ifade verileri üzerinde yapılan hücre sınıflandırması çalışmaları özünde bir "özellik altkümesi seçimi" problemi barındırmaktadır. Yapılan çalışmanın amacı, kanserli ve sağlıklı hücre örneklerini, en az sayıda özelliği –geni- kullanarak, başarılı bir şeklide sınıflandırabilecek bir araç geliştirmektir. Çalışmada iki yeni algoritma geliştirilmiştir. Birincisi, verinin bulanık kümelendirilmesinin ardından, bulanık kümelerin oluşturduğu yeni veride yüksek güvenilirlikli EĞER-İSE kurallarını tümünü arama yaklaşımıyla keşfeden bir algoritmadır. İkincisi ise, birincinin prensipleriyle, veri üzerinde, tümünü arama yönteminden ziyade, ışın arama algoritması ile kural keşfeden bir algoritmadır. Son algoritma ise özelliklerin –genlerin- taşıdığı hiyerarşik yapıdaki bilgiye odaklanmaktadır. Bu hedefle karar ağaçları oluşturmada farklı başarı ölçütleri kullanılmıştır. Lösemi veri kümesinde başarılı sonuçlar elde edilmiş, karar ağacı temelli algoritmada ise kolon kanseri veri kümesi ile başarılı sonuçlara ulaşılmıştır.

ACKNOWLEDGEMENTS

First of all I want to thank to my advisor Dr. Kemal Kılıç for his guidance, tolerance and patience, which made this study possible. Also, I would like to thank Dr. Uğur Sezerman for his guidance into the world of bioinformatics. I am thankful to my jury members Dr. Gürdal Ertek, Dr. Özgür Gürbüz and Dr. Tonguç Ünlüyurt for their sincere efforts to bring my studies to a higher levels.

For the financial support I would like to thank to TUBITAK BIDEB, and my advisor Dr. Kemal Kılıç.

My friends Ayfer Başar, Ece Erkol, Mahir Yıldırım and Serkan Çiftlikli started the graduate journey with me. They deserve the most sincere thanks for their support in difficult times.

For the wonderful and unforgettable moments that I had in FENS 1021, I thank my dear friends L. Taner Tunç, Burak Aksu, Dr. Emre Özlü, Figen Öztoprak, Duygu Taş, Elvin Çoban, Umut Kirmit and Lale Tunçyürek. I also would like to thank Dr. Ahu Gümrah Dumanlı for her enjoyable neighbourhood. Even though he thinks he is not able to help me because of his "circumstances", Taner has always made me feel special by sharing his thoughts with me (theory of evolution, politics, girls and boys, Beşiktaş, and many others), he is one of the most outstanding people I will ever meet. Also I shall never forget Burak's fellowship during my life at SU, whose tastes I find the nearest to mine, except for the "green figs". Figen "Figi Hocam" is the most inspiring person for me, whose endurance, intelligence and interest in history impressed me most of the times. Nurşen Aydın, Gamze Belen, Birol Yüceoğlu, Sevilay Gökduman, Ömer Özkırımlı and Gamze Koca are the following members of "the office" who are also wonderful people whom I will always be happy to know.

Sermin Gürel, Deniz Toka and Didem Güven have always been with me, and they always will.

CONTENT

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Since the discovery of DNA structure, the mechanics of life has been revealed to the exploration of humanity. The new era began with the completion of the human genome map in 2003. From DNA sequences to complex protein structures, massive information is carried through nucleotides, which make up the alphabet of the language of life.

The massive information carried through biological molecules is analyzed with the tools of applied mathematics, data mining, artificial intelligence and statistics. The study of biological problems with the help of these tools includes "computational biology" and "bioinformatics". Briefly speaking, the science of developing algorithms with these tools is referred to as "computational biology" and the utilization of these algorithms in order to attain new biological knowledge is referred to as "bioinformatics".

Supervised classification has a significant role in computational biology and bioinformatics research. It is basically the act of classifying a new *sample* in order to acquire certain information about it based on historical data. As an approach to the solution of the classification problem, the "machine learning" concepts have been used. The information attained by the past samples is learned by the help of computers. When there are significantly many features associated with each sample, it is crucial to determine which features actually affect the classification, *i.e.*, the determination of the class label. Too many features might convey irrelevant or redundant information, whereas lack of features might lead to bias during the classification task. Both cases imply high misclassification rates. Hence, determining the subset of features to perform the classification task is crucial and referred to as the "*feature subset selection*" problem. Accurate classification can be achieved with minimum number of features (*i.e.*, with minimum measurement cost) by determining the subset of features that are relevant and necessary.

Recent advance of technology gave birth to tools such as microarray chips. The use of microarray chips enabled the scientists to measure the amount of protein production from their genes in a cell known as the gene expression data. The classification of cell samples by the help of their gene expression data is currently a hot research area. The information obtained from the microarray chips include the information about the amount of proteins that are *transcribed* from the genes (referred to as the expression levels of the genes), and the variation in its level among the cells might be due to the cells typology, *i.e.,* the class labels such as healthy, cancer, etc. The data collected for gene expression level analysis is massive and therefore the features, *i.e.*, the genes, must be reduced to a reasonable level due to the computational cost of experiments and the possibility of misleading irrelevant genes. Therefore, usually, the analysis based on the classification of cell samples includes a *feature subset selection* phase.

Three novel algorithms are proposed for the gene selection problem based on basic association rule mining. The first algorithm starts with fuzzy partitioning of the gene expression data and discovers highly confident IF-THEN rules that enable the classification of sample tissues. The second algorithm search the possible IF-THEN rules based on a heuristic pruning approach which is based on the beam search algorithm. Finally, the third algorithm focuses on the hierarchical information carried through gene expressions by constructing decision trees based on different performance measures.

In Chapter 2, a review of relevant literature regarding the feature subset selection problem and the methods used in the proposed algorithms is presented. In Chapter 3, algorithms that are used for feature selection and colon cancer data classification are proposed. Implementation of the proposed algorithms on colon cancer dataset is given in Chapter 4. Chapter 5 will include conclusions and future work.

CHAPTER 2

LITERATURE REVIEW

In this chapter we will provide the relevant literature regarding both the problem area and the methods that will be utilized in the proposed algorithms. First we will briefly discuss the microarray experiments and the feature subset problems for the readers who might not be familiar with either of the fields. Later, the literature regarding the association rule mining, the decision trees and beam search will be presented since they are the concepts that are utilized in the algorithms that are proposed in this thesis.

2.1. Brief Review of Biology and Microarray Experiments

Proteins are organic molecules that play role in every biological mechanism in a living organism. They make up cells, produce energy, enable oxidation, digestion etc. in molecular level. In addition, proteins carry the characteristics of the species that they belong to. Proteins are made up from *amino acids* and are produced under the management of the deoxyribonucleic acid (*DNA*). DNA, manages the production of proteins by the help of the sequence information it carries. DNA is basically a chain made up of *four nucleotides,* namely *A (adenine), C (cytosine), G (guanine)* and *T (thymine)*. It has a double helix structure in which two strands of nucleotides are bonded with each other. Each nucleotide type can make bonds with a specific nucleotide type at the opposite strand, (*A* with *T* and *G* with *C*), which enables the sequence information to be carried through transcription phase. That is to say, given one of the strands, one can determine the sequence of the other strand easily. This sequential information is carried through several steps that were described under the term of "*central dogma of molecular biology*" [1].

Central dogma begins with the *transcription* of the sequence information on a DNA sequence to mRNA (*Messenger Ribonucleic Acid)*, which is also made up from four

nucleotides like the *DNA* (only difference is *Uracil* replaces *Thymine* in the case of RNA's), and is constructed according to the alignment of DNA. Nucleotides are the tiles that construct the alignment information.

After the sequence information is transcribed from DNA to mRNA, which can pass through nucleus membrane, the information is carried to the ribosome where the proteins are synthesized. This is referred to as *the translation* step where the amino acids are combined to produce proteins. Every sequence of a triple of nucleotide, referred to as *codon,* represents a specific amino acid. Considering the fact that the alphabet of DNA consists of 4 letters (A, C, G, T), a codon can represent $4^3=64$ different amino acids. However, in cells 20 *standard amino acids* are available for protein production which allows several codons to code each amino acid. Furthermore there are also specialized codons such as the start codon and the stop codon which informs the ribosome to start or stop the production process.

Amino acids that are going to take place in the structure of the protein are carried to the ribosome by the transfer RNAs (tRNAs) which combine amino acids according to the sequence information originated at the DNA and carried by *mRNA*. tRNAs that can make bonds with the nucleotide triples on mRNA are aligned, *i.e.*, the amino acids that are carried by those tRNAs come together to produce the protein (Figure 2.1).

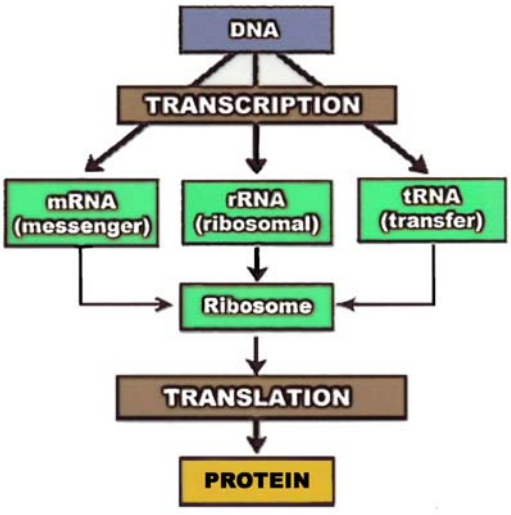

Figure 2.1 Central dogma framework: from DNA to protein

Those regions of DNA which are encoding potentially functional products are referred to as the *genes*. Note that, even though DNA is a very long chain of nucleotides (e.g., human genome is about 3 billion base pair long), only a small portion (about 2%) of it actually is coding proteins. The process of producing a biologically functional molecule by transcription of genes is known as the *expression of a gene,* and the amount of the product is referred to as the *expression levels of genes*. As discussed earlier, the expression level data has invaluable information and recently became a significant tool in biological research areas such as development of better diagnostic tools, drug identification, *etc*. Measurement of gene expression levels can be in different steps of protein production. Amount of *mRNA* or the amount of protein translated from mRNAs can be measured. The measurement instrument is referred to as the *microarray chips* (or shortly *microarrays*) and the process is referred to as the *microarray experiments*.

Microarray experiments enable scientist to measure protein transcription levels of thousands of genes simultaneously. Cell samples are taken; their mRNA is purified and labeled by fluorescent material using *real time polymerase chain reaction* (PCR). Next, the microarray is prepared for the experiment: on each spot of the microarray, there are identical copies of every gene's single stranded DNA structure. Microarray is combined with labeled mRNA samples which originate from different cell types. Afterwards, microarray is washed. Following the washing phase, *only* the mRNA that can synthesize with DNA strands on the spots is left on the microarray. What is left on the microarray reflects the amount of transcription of proteins from the DNA strand initially located on the microarray. Later the microarray is processed by the computer and expression levels of every gene spotted on the microarray is taken as output. The obtained expression level refers to the amount of transcription of information from DNA to mRNA.

There are three types of microarrays according to the material that is spotted on. DNA, cDNA and oligonucleotides can de spotted on the microarrays for the hybridization with fluorescent mRNA of the instances. Note that the oligonucleotides are short single stranded DNAs and are widely used for microarray experiments.

Microarray experiments are used for various research objectives. Firstly, they are used in order to identifying the set of "differentially expressed" genes due to different cell types. Another research area is the exploration of gene sets that behave similarly among the samples (gene expression patterns) [2]. Single Nucleotide Polymorphisms (SNP) identification is also a research field where analyses based on microarray experiments is utilized [3]. The genetic

variation between the individuals of the same species is explained with the presence of SNPs. During evolution, a single nucleotide in DNA strand differs and this differentiation passes to next generation. Studies on SNP data aim to detect subset of SNPs that are associated with genetic diseases related to mutation. DNA Binding Sites are also discovered by the help of microarray experiments. In order to detect locations where protein-DNA interactions take place, i.e. binding sites, are also studied by microarray analysis.

2.2. Feature Subset Selection

Feature subset selection problem deals with the process of selecting the most relevant features in classification problems in order to attain accurate classification. The data set tabulated in Table 2.1 will be used as an example that demonstrates the effect of the feature subset selection on accurate classification. Data yields the people's choices on privacy concern in the internet and contains four features [4].

Table 2.1 Instance data on choice of privacy concern

age	annual income (money unit)	Hours spent online per week	no. of e-mail accounts	privacy concern
26	90	20	4	yes
51	135	10	2	no
29	89	10	3	yes
45	120	15	3	yes
31	95	20	5	yes
25	55	25	5	yes
37	100	10	1	no
41	65	8	2	no
26	85	12	1	no

Based on the information retrieved from four features, it is not clear to determine how different features affect people's concern on privacy. Figure 2. is the scatter plot of the data represented by only the first three features. The red triangles represent people with privacy concern and the blue squares represent people with no concern. It can be observed from the figure that the classification criterion is not clear and the relation of the features with the outcome cannot be stated geometrically.

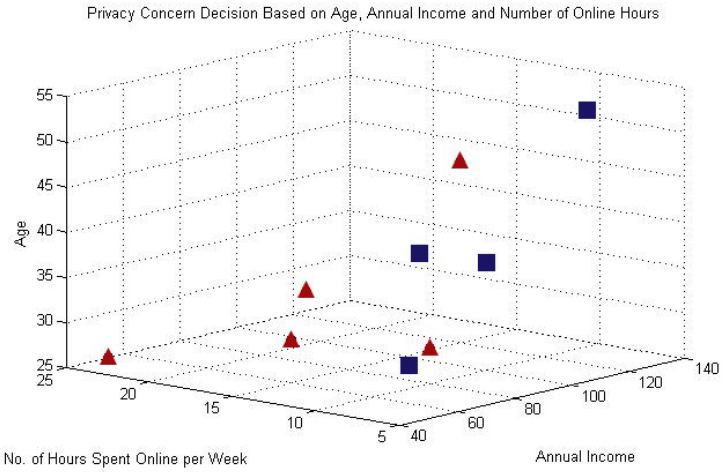

Figure 2.2 Scatter plot on instance data with 3 features

In Figure 2..3, the data points are plotted only on two axes: number of e-mail accounts and number of hours spent online per week. By considering only these features, the classification of the sample data points can be achieved by partitioning them by the red line on the graph. By eliminating two redundant features and only dealing with the features that are relevant, accurate classification can be achieved intuitively.

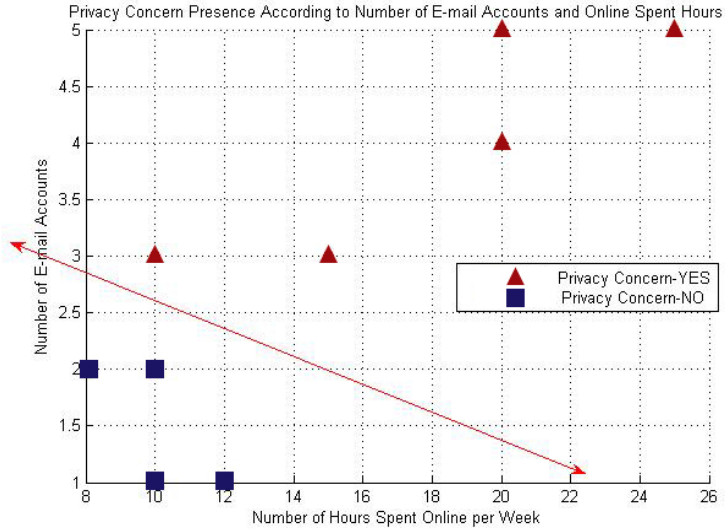

Figure 2.3 Scatter plot of the example data using two features

There is a wide range of feature selection algorithms which are applied in various real life problems such as customer relationship management [5], recommendation systems for web marketing [6], image analysis [7] as well as microarray gene expression analysis [8]. In the literature these approaches are mainly classified as *filters*, *wrappers* and *embedded methods*. A very good literature review of the problem can be found in [9].

Filter methods include utilize some statistics based on the distributions of values of features, such as entropy, correlation, etc during the feature selection process. Best feature subsets that have highest performance measures based on these statistics are selected in order to perform classification [10]. In the *filter methods* the learning stage begins *after* the feature subset is composed.

A widely used measure for filtering the set of features is correlation. Investigating the linear relationships between two variables is the essence of correlation. Both the correlation between class labels and the feature and the correlation within selected features is a point of interest. If this measure is used, the features that are irrelevant can be eliminated and the features with less correlation among each others can be selected. Hall [11] states that the elements of a good feature subset must be highly correlated with class labels and minimally correlated with each other. He used this premise and proposed a filtering method and conducted an experimental analysis in order to explore its performance with respect to the classification accuracy. The algorithms that utilized the proposed filtering methodology outperform those that use Naïve Bayes Algorithm and other procedures as C4.5 (tree generation algorithm that uses impurity when constructing the decision tree) [12] and ID3 (a primitive version of C4.5 introduced by Quinlan in 1986 [13] where decision tree is constructed by entropy value of features on the test instances) without correlation based feature selection.

One of the most famous feature selection algorithm is the RELIEF Algorithm, which has been firstly introduced by Kenji and Rendell [14]. For each feature the algorithm randomly selects *m* instances. For each instance, the nearest miss (the nearest instance which is *not* in the same class) and the nearest hit (the nearest instance which is in the same class) are determined. Next, the difference between the distance of nearest miss to the instance and the distance of the nearest hit to the instance is calculated. The average value of the *m* differences that will be calculated for each one of the *m* randomly selected instances is a measure of the selected feature's ability to distinguish the instances that have different class labels and are near to each other and referred to as the RELIEF measure. The features that have high RELIEF values are selected as significant features.

Many algorithms derived from RELIEF measure are proposed in the literature. Notably, the Relief-F is introduced Kononenko's study [15] which can deal with multiclass and incomplete data. Relief-F deals with nearest *k*-hits and *k*-misses, that is to say, instead of dealing with only the nearest instance with the same and different class labels, this algorithm

determines *k*-instances that is nearest to the selected instance. Average of distances to the hits and misses are taken. The parameter *k* ensures the handling of noise in the data [16]. In addition, the missing data issue is solved by making a probabilistic estimation on the missing value.

Introduced by Almuallim and Dietterich [17], FOCUS algorithm firstly handles each feature individually, then adds features to construct pairs, triples, etc., according to the impurity measure which can be defined as the performance metric that measures how well the selected feature can partition the instances in a way that each partition is composed of instances with the same class.

Another measure group that is used in filter methods includes metrics based on entropy. Entropy measures the randomness of the distributions of feature values. It is calculated as in Equation (2.1) where *c* represents the number of classes, and *x* represents the discrete random variable.

$$Entropy(x_i) = -\sum_{i=0}^{c-1} p(x_i)\log_2 p(x_i), \tag{2.1}$$

As the entropy of a feature decreases, it is more likely that this feature is more successful in classification task. In their paper, Liu *et. al.* [18] introduce a feature selection scheme that filters the features with higher entropy scores. Another performance measure derived from entropy is the Mutual Information [9]. Mutual information determines how much variable *x* is dependent to the target label *y* by Equation (2.2). In Equation (2.2), *p(X)* and *p(Y)* represent probability distribution functions and *p(X,Y)* represents joint probability distribution function of variable *X* and *Y*. Mutual information proposes the amount of knowledge we have about a variable by knowing another variable that is dependent to it.

$$Mutual\_Information(i) = \sum_{x_i}\sum_{y} p(X = x_i, Y = y)\log\frac{p(X = x_i, Y = y)}{p(X = x_i)p(Y = y)} \tag{2.2}$$

Another approach for the feature subset selection problem is the *wrapper approach*. Wrappers are defined as black box structures that select the features according to the learning algorithm itself. The key logic behind the wrapper algorithms is the fact that the aim of feature selection is improving the classification accuracy so why not determine the features that yields best classification algorithm but use other measures such as correlation, entropy, etc. As the selected feature sets are constructed, their performances are measured by training and testing the classification algorithm on selected feature subset.

In wrappers, three main issues arise when constructing the whole algorithm: how to search the feature space, how to guide the search by the help of performance measure, and which performance measure to use in the predictor or the classification algorithm. In order to select the features to be tested, heuristics based on randomized or nonrandomized procedures can be used.

Kohavi and John [19] introduce an extensive study on wrapper approaches for feature selection. Their experiments analyze the impact of search strategy and the learning algorithm on the selection of feature subsets. Liu and Setiono [20] introduce a probabilistic wrapper approach that is based on the Las Vegas Algorithm. This algorithm randomly selects a constant number of features and applies the classification algorithm in order to identify the performance of the feature subset. Those that are in the feature subset with the minimum error rate are selected as the significant features. They used C4.5 which is a decision tree generation method and ID3 for the learning. They measure the accuracy by testing on artificial datasets, and focus on the computational time. The primary issue on the computational time of the algorithm was reported to be the complexity of the learning algorithm.

Following this study, many algorithms that use meta-heuristics for search procedures have been developed. Zhang and Sun used Tabu Search which is an intelligent randomized search procedure that prevents the entrapment in local optima by tracking a *tabu list* that records the moves that will lead to the worse solution sets or to the solution sets discovered before [21].

Another random search algorithm is the *genetic algorithm*, which is widely applied also as a wrapper approach in feature subset selection problem. Every iteration, a new set of chromosomes is identified via *reproduction* and *mutation* processes, which mimic natural evolution. The chromosomes represent a feature subset and the fitness function score is the classification accuracy attained by using those features. The chromosomes that have better fitness function scores are allowed to reproduce and pass to the next generation and the chromosomes that have worst scores are eliminated from the gene pool. As it is the case with genetic algorithms, the drawbacks of wrapper approaches can be stated as the greater computational time to perform the measurement of the feature set's classification accuracy and the model's proneness to overfitting [9].

Randomized search algorithms that are applied with wrappers are widely used for cancer classification of microarray datasets as well. Randomized search based selection algorithms rely on the fact that the measures in selection of feature subsets do not support

monotonicity assumption. Therefore search of every possible subset is required in order to obtain the optimal feature subset. Since, it is infeasible to do this, clever search algorithms that are based on randomness are applied in a wide range of studies.

Yang and Honavar [22] apply genetic algorithm on searching for the best feature subsets and perform induction with artificial neural networks. Artificial neural network that is constructed with selected features from genetic algorithm outperforms the network constructed by the whole features. Another application of genetic algorithms, introduced by Vafaie and DeJong, yield better results when compared to sequential backward selection. They used genetic algorithms to select feature subsets and evaluate their fitness scores by their rule induction algorithm stated in the paper. They state that using genetic algorithms reduces the computational time when compared to sequential search procedures [23]. Handels and Ross [24], utilizes genetic algorithm in order to find the best subset of features that classifies skin tumor from images. By using k-nearest neighbor for the classification accuracy measure, the performance of the chromosomes is evaluated. Their results show that features obtained by genetic algorithm outperform the subsets obtained by greedy search and heuristics. Liu *et. al.* [25] applied genetic algorithm as the feature search procedure, and used SVM method for predicting the classes. By experimenting the classification results of the test instances, SVM provides measuring the performance of features that are selected. Their algorithm was run on microarray dataset to perform cancer classification. Their results show that from different random seeds, different subsets that make almost accurate classification of instances can be evaluated.

One of recent studies that use genetic algorithms in feature subset selection is used on microarray data for cancer classification. Kucukural *et. al.* [26] used genetic algorithms in order to discover the feature subset with minimum elements. Their genetic algorithm initially creates generations i.e. feature subsets, and measure their classification performance by classifying test instances using SVM. Through this step, genes are weighted according to their occurrence and classification score of the subsets they belong. Afterwards these weights are used in order to create a new generation series, applying roulette wheeling approach by weights evaluated. This enabled the algorithm to choose one gene (feature) at multiple times, thus decreasing the number of features selected for the subset, in the chromosome construction phase.

Finally, a third approach to the feature selection problem is classified as the *embedded methods* in which the classification is performed *simultaneously* with the feature selection

phase. In these approaches the classification phase can't be separated from the selection phase which was possible in *wrapper algorithms*. Artificial neural networks and association rule mining are mostly applied embedded methods for feature selection in microarray data. Bloom *et al.* constructed a ANN based framework that combines gene expression data of different types (cDNA and oligonucleotide arrays). Their results show that it is possible to make a system that accurately classifies tumor instances that are taken from different parts of the organism (i.e. breast, colon, ovary etc.) [27].

A study reviewing some techniques for feature subset selection for cancer classification came from Li *et. al.* [28]. They questioned the effectiveness of filter methods and tested symmetrical uncertainty based filter algorithms with SVM, Naïve Bayes and C4.5 in classification phase. They concluded that filters are not efficient on microarray data. Liu *et. al.* also studied microarray data, they [29] used normalized mutual information which is derived from entropy, and minimized entropy among genes. They concluded that their method was strong in classification, but weak in detecting the genes that play role in cancer formation.

2.3. Association Rule Mining

In this thesis we will propose two algorithms for the subset selection and the classification problem for microarray data. One of the algorithms will utilize the association rule mining concept. Association rule mining is used in order to infer frequent relationships between feature behaviors. In the literature, various algorithms are introduced in order to reduce the computational time while gaining more powerful association rules [30]. The concept was firstly used to investigate consumers' behaviors in retail market. The aim was to detect consuming habits by analyzing transactions. Every product was named as an *item* and every transaction belonging to a single consumer was named as an *instance*. Basic concepts and the terminology relating to the rule discovery in association mining are as follows;

- *Item set*: The set of items selected to construct a rule is an item set. A *k item set* refers to an item set that have *k* items.
- *Antecedent*: In general the rules have *IF-THEN* structures. The antecedent of a rule represents the condition in it, i.e., the *IF* part.
- *Consequent*: Consequents are observed if antecedent occurs with estimated confidence of the rule. It is the part that follows "*THEN*" in the rule.

- *Support*: Support is the frequency of occurrence of an item in the instance set. It can be represented as the number or the percentage of instances containing the item. Creighton and Hannah [31] introduced market basket analysis as a rule generation concept to investigate genomic data. In the context of microarray datasets, where the variables (i.e., *gene expressions*) are continuous, items can be the high (or low) expression levels of a gene in the gene set. Every instance represents a transaction, in which highly expressed genes are identified. An item with high support means that it is frequently seen in the instance set and its presence might be characteristic for the class that the support is evaluated. The support of an item set is calculated as number of instances in which all of the items can be observed simultaneously.
- *Confidence*: Confidence is the percentage of occurrence of the consequent *if* the antecedent is observed. The greater the confidence, the more powerful the rule is.

In order to determine the association rules, firstly, the item sets with support values above a certain threshold level should be discovered. Later, the rules with high confidence are determined based on the discovered item sets. There are three basic approaches for discovering the item sets that have support values above the threshold, namely, *A priori*, *Sampling* and *Partitioning*. *A priori* is based on the simple assumption that as new items are added to the item sets their support can't increase. This approach apply breadth-first search and count the items for support calculation. *Sampling* detects association rules by searching through random sets of instances. On the other hand, *Partitioning* divides the dataset into partitions and search for association rules soon to be combined with association rules evaluated from other partitions. These algorithms also use breadth-first search strategy, but rather than counting the supports of item sets, they detect intersection of item sets that belong to different partitions/random instance sets. For a basic survey on discovering association rule we refer the reader to Hipp *et. al.*[32].

Introduced by Agrawal *et. al.* in 1994 [33], *A Priori Algorithm* aims to find rules of pre-determined size. Procedure is based on the following hypothesis: "all subsets of an item set that has a high support level must also have a high support level". At each iteration, algorithm takes (*i-1*) sized item sets that have (*i-2*) item sets in common and constructs a candidate item set by combining them. A priori algorithm applies breadth-first strategy to search through possible item sets. That is to say, in order to discover a *k-item set*, all of the *(k-1)-item sets* should be discovered in this approach.

A sampling algorithm is presented by Toivonnen [34] in which without considering all of the instances in the instance set, only a few representative instances are utilized in order to discover the item set. The sampling algorithm is combined with other searching procedures to discover association rules such as a priori or ECLAT [35] that applies different support count strategies. In sampling approach, random instances are drawn and association rules are derived from this instance subset.

Partitioning algorithm reduces computational time by partitioning the set of instances into subsets. This also reduces the CPU overhead [36]. Other than the computational benefits, by considering every partition as different states, association rules can be generated in an easier way, without looking at each instance, association rules are generated from different partitions are combined –intersection of these sets are determined- in order to discover association rules that represent the whole instance set. Similar to the a priori based algorithms, partition algorithms also use breadth-first strategy to discover item sets. That is to say, in order to discover a k-item set, all of the k-1-item sets should be discovered.

Once an item set is discovered after a searching phase, the support of the discovered item set should be computed. The easiest way in terms of application, yet the most computationally expensive way is to count the occurrence of item sets in the set of instances i.e., transactions.

In a priori based algorithms, the list of instances –transactions- are used for support calculation. Whereas, in some algorithms, such as the ECLAT [35], the list of transaction identifiers (tidlist) structure is used that is constructed as the transposed version of the instance set. Using this approach is an advantage in datasets that have few number of instances and large number of items. In this algorithm rather than counting the support of item sets, intersection of item sets that are representative in instances are used to discover association rules. For each item, the set of transactions that include the item is determined and kept in the tidlist. This methodology uses set of intersections in order to compute the support for item sets. As the two item sets are combined, it is possible to compute the support of new subset by defining the intersection of tidlists of the two combined item sets [32]. One of the main benefits of using tidlists is the low computational requirements: simply detecting intersection of any (k-1) subsets to reach the support of k-item set. That means, all instance set is not scanned during the support calculation process.

Partition algorithm also uses tidlist structure to find the support of frequent item sets, as it also benefits from the intersections of item sets of partitions when discovering item sets

representing the whole instance set. After detecting frequent item sets in the partitions of the instance set, supports for these items are computed through the whole instance set by determining the intersection of item sets.

ECLAT starts with single items and continues constructing item sets in a depth-first manner. New item sets are discovered until an infrequent item set is discovered. When an infrequent item set is discovered, as the depth search strategy offers, the algorithm returns to the earlier levels of the tree and continues constructing item sets in the same fashion. ECLAT uses the same search strategy as the A Priori Algorithm of Agrawal [33], whereas the determination of support is different. The novel part of this algorithm is computation of the support by transforming transactions into tidlists. After this transformation, it is possible to determine the supports by using the intersection of item subsets. Zaki states that ECLAT performs better than A Priori Based Algorithms proposed by Agrawal and Partition Algorithms. In addition, Borgelt points ECLAT's efficiency; however it is stated that this algorithm needs a lot more memory than other algorithms.

We have introduced the basic concepts and approaches regarding association rule mining. In the literature there are various algorithms that explore association rules in large databases like the microarray data. *Top-k Covering Rules Algorithm*, which is introduced by Cong *et. al.* [37], finds the most successful *k* rule groups that are found in every partition of the dataset. Algorithm starts with removing infrequent items. In order to proceed to search for rules, a data matrix in which rows are vectors indicating each item's presence in instances is constructed. Before the depth first search, in every row, rule groups for each row are determined. A rule group covers all the possible subset of items that are common in the given instance set. During the depth first search, in order to evaluate the rule group for the visited node, that is to say, compute support and confidence easily, transposed tables are used. This algorithm does not mine rules in the whole dataset. In order to conduct association rule mining, genes are filtered according to entropy score or the genes and then discretized.

Another algorithm is the FARMER [38]. In this algorithm, data is handled as rows representing instances. FARMER also focuses on rule groups rather than constructing association rules. Items are first discretized for rule mining procedure. FARMER mines association rules according to instance enumeration e.g. finding common genes in an instance set that is represented as a node in the search tree. On this tree FARMER conducts a depth-first search.

Other studies include CLOSET algorithm [39] and its various derivations. In CLOSET, instead of mining all frequent item sets (item sets with supports above the desired threshold), this algorithm mines all frequent *closed* item sets (*largest* item sets with supports above the desired threshold). In order to conduct the searching phase, the algorithm uses FP-Trees (a prefix tree to store the items dataset) which compress data for efficiency increase. Another algorithm named CHARM [40] searches both the item space, i.e. the instances that are covered by given item set, and the instance space, i.e. the items that are common in given instance set.

In the literature there are rare studies that focus on classification based on the association rule mining algorithms. Without conducting any discretization on the data, Georgii *et. al.*'s algorithm [41] mines association rules, as the separation mechanism of the rule includes hyperplanes that minimizes the classification error.

Note that, none of these methods initialize the data with fuzzy partitioning and usually discretize the data to turn it into a data structure more like market transaction databases, which in every transaction items are listed if they are bought from the market. This situation limits the validity of the discovered rules.

2.4. Fuzzy Association Rule Mining

Dubios *et. al.* [42] provide a detailed study on the use of fuzzy logic in association rule mining. They discuss some primal issues e.g. calculation of support and confidence, in fuzzy logic basis and compare the methods that have been used earlier in the literature. Even though they point that using different approaches seems to make no difference in mining association rules, it is left as an open question for further investigation.

A study for association rule mining in a database containing fuzzy features comes from Hong *et. al.* [43]. On the basis of a priori, without starting the mining of association rules, they eliminate the features that are not frequent (i.e. with not satisfactory support). The way support is calculated is based on the membership degrees, as the membership degrees are summed in order to determine the support of a feature. To calculate the support of a feature set, the membership of a feature set to an instance is computed as the minimum of the membership degrees of the features in the set.

De Cock [44] focuses on the methods used to define support and confidence in association rule mining. They classify the relevance of a rule with an instance from positive instance to non-negative instance. They introduce four quality measures that are derived from the outcomes of previous classification in crisp sets. Then, they implement the proposed measures to fuzzy sets. They concluded that the most powerful quality measures, however, are stated as support and confidence.

Wai-Ho *et. al.* [45] studied the change in databases over time with a fuzzy association rule mining perspective. Their proposed framework takes the supports of confidences of the association rules of databases that are constructed in different times. The change in support and confidence of these association rules are observed by linguistic variables i.e. with fuzzy sets. They partition the change in rules as "highly decrease", "fairly decrease" or "more or less the same" etc. and use membership functions to determine the support of each label. Another approach proposed is to build fuzzy decision trees to analyze the change in association rules.

Sudkamp [46] introduces performance measures based on the relevance of instances to the rules. Instances are classified as "examples", "counterexamples" and "irrelevant examples". He states that instances can be defined by association rules "to a degree" if fuzzy rules are used. To determine the relevance of a rule to an instance, it has been found that product is the unique T-norm.

Xiang-Rong *et. al.* [47] introduces a study which mines association rules from microarray gene expression data. They propose FIS-tree mining algorithm, which determines rules from the microarray data under different experimental conditions. They use different data structures e.g. BSC-Tree which is claimed to be a compression format for mining procedures, and state that their mining algorithm outperforms A Priori and Partitioning.

Another study comes from Kaya and Alhajj [48], which searches for association rules using genetic algorithm framework. They use market transaction data and construct chromosomes that represent centroids for all features to be partitioned to fuzzy sets.

Becquet *et. al.* [49] take initial steps towards association rule mining in gene expression data. In their paper, association rule mining is used to determine co-regulated gene expression patterns. They used SAGE data, which stands for "serial analysis of gene expression". Their motivation was to detect genes that are definitely related to the biological state that is being investigated, while they claim that other algorithms were not able to do this.

Their proposed algorithm is an unsupervised learning technique, that is to say, biological state -i.e. outcome- information is not inserted into the algorithm. Their rule prototype is as follows: "IF gene *x* and gene *y* have significantly high level of expression, THEN gene *z* has high level of expression". In order to precede mining association rules, they conduct discretization techniques. Following this step, the binary matrix representing over expression of genes, are transferred to AC-Miner software used to discover association rule mining.

The usage of fuzzy association rule mining for classification (prediction) is studied in Yuanchen *et. al.* [50]. They applied A Priori algorithm to generate fuzzy association rules on the data that is fuzzy partitioned in previous steps. However, they do not use microarray datasets and the datasets include very few features. Their findings are promising and they claim that their method outperforms C4.5 (explained later in the chapter) and SVM (support vector machine). Icev *et. al.* [51] introduces distance-based association rule mining algorithm to determine gene expression patterns based on protein binding sites. The algorithm extends previous research on protein motifs by taking into account the distances between protein motifs when constructing association rules. This contribution improves the accuracy when compared to APriori Algorithm. Rodriguez *et. al.* [52], improves the classic A Priori algorithm by using pruning structures and different data structures, and reduce the time needed to discover item sets.

2.5. Beam Search

First beam search algorithm was reported as HARPY [53] which was used for speech recognition. Ow and Morton [54] give an extensive study on the application of beam search on scheduling problems, where each path in the search tree corresponds to a candidate schedule of jobs. Following Lowerre, beam search has been widely used in speech recognition (Ney, Mergel, 1987[55]), (Alleva, Hwang, 1993[56]), (Nguyen, Schwartz, 1999[57]). In many algorithms related to feature selection, beam search is counted under different type of search strategies, as an alternative to branch & bound, sequential selection/elimination and random search. Carlson *et. al.* [58] used beam search for identifying potential protein binding sites. They claim that limiting the search space to most relevant gene sequences (candidate solutions) enables the algorithm to use more complex and computationally costly objective functions that give more accurate information on sequences i.e. paths in the search tree.

2.6. Classification and Regression Trees – CART

Decision trees are widely used in classification problems due to their effectiveness, relative ease to implement and understand their outcomes [59]. Decision trees handle the features in a hierarchical way. At each node a feature is selected to conduct the classification. The partitioned instances are carried through the paths in the tree to be partitioned by the features that are selected in the lower levels of the decision tree. Decision tree starts with a single node. The number of partitions generated by the feature selected for the starting node determines the number of child nodes for the next level of the decision tree. The growth of a decision tree that is used for classification terminates when there is no use of adding a new node under any leaf of the tree.

Introduced by Breiman in 1984 [60], classification and regression trees approach is based on partitioning the instance set into two subsets at each level of the regression tree. CART uses its own measure for selecting optimal node which is most successful to divide the data into two most *homogenous* parts. The details of this algorithm will be presented later in Chapter 4. Also in other performance measures, at each node, the feature and the splitting point -i.e. data is divided as the instances that are less and greater than this value- that reduces entropy i.e. impurity the most are determined. As a consequence, at each node, it is possible to reduce "impurity" maximally. Mostly used impurity measures are as follows:

$$Entropy(t) = -\sum_{i=0}^{c-1} p(i \mid t) \log_2 p(i \mid t),$$

$$Gini(t) = 1 - \sum_{i=0}^{c-1} \left[p(i / t)\right]^2,$$

$$Classification\_error(t) = 1 - \max_i \left[p(i \mid t)\right]$$

where $p(i|t)$ at node t is the portion of instances that is labeled as class i and c is the number of classes.

In order to make a comparison of the impurity of the child and parent node, the change in impurity measure should be checked. In order to determine the goodness of the chosen feature and the split, delta is calculated. Optimal feature is selected when maximum decrease in impurity -i.e. delta is maximum- is reached. Delta is calculated as follows:

$$\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

where $N(v_j)$ is the number of instances belonging to child node v_j, $I(.)$ is the impurity function and k is the n umber of child nodes.

Decision trees are widely used in machine learning applications. Yoon *et. al.* [61] use decision trees for recommender systems for internet marketing. They introduce a study for a recommender system for users during web-shopping. They use decision trees in order to generate association rules on the buying behaviour of the selected customers and apply this tree structure to test customers.

Another application is on the usage of CART in medicine. Nelson *et. al.* [62] used the CART™ software in order to construct classification trees for determining risk groups of patients. They conclude with rules that identify risks by thresholding attributes that are selected for the construction of regression trees.

A comparison study of different classification methods were introduced by Wu *et. al.* [63], where classification and regression trees was also tested as one of the methods. In their paper, Wu *et. al.* combined boosting with CART, where boosting means random selection of instances according to their outcome of classification, that is to say, if a instance has been classified correctly by previous nodes, the new instance set is less likely include the classified instance. This study is conducted on ovarian cancer mass spectrometry data, and the results found show that constructed CART algorithm combined with boosting can be as powerful as SVM classifier.

One of the issues that is often regarded as a problem of regression and decision trees is the over fitting issue. There are two basic errors that can be defined when constructing a decision tree structure. Training error refers to the degree of misclassification while the algorithm is being trained with the historical data. Whereas, the generalization error, i.e., the test error, is the misclassification error of the testing data, which is not utilized through the learning phase. Hence generalization error is the expected error of misclassified testing instances during the future usage.

It is important to handle the issue of over fitting in order to reach high generalization error. In one sense, over fitting means that the features selected for classification are too specific and takes into consideration only the necessary information required to correctly classify the training instances. In order to reduce over fitting, pruning should be performed.

Pruning strategies incorporate the elimination of nodes that are relatively unnecessary by means of information gain, which is based on the contribution in the impurity decrease of the parent node.

2.7. Decision Trees vs. Association Rules

When an association rule is generated, the instance set is being partitioned simultaneously by all of the features involved in the item set. On the other hand, in a regression tree structure, there is a hierarchical sequence of features that lead to a classification of the instance set. The use of the features in the classification stage differs in the two approaches. Therefore, the two approaches have different ways of handling the search for set of features. Rule mining procedures perform an enumeration on the features that would satisfy desired support and confidence thresholds. However, decision trees make a sequential search on the features, and select the best features only according to the instances that are represented in the current node. Every path in a decision tree corresponds to a hierarchical rule structure i.e. there are multiple if-then rules embedded in each other.

CHAPTER 3

PROBLEM DEFINITION AND PROPOSED ALGORITHMS

In this chapter we will first present the mathematical structure of the feature subset selection problem. Next, the three algorithms that are proposed in this thesis for the feature subset selection problem will be introduced. Two of these algorithms aim to find association rules based on the fuzzy set theory. The first one of these algorithms will be referred to as the Fuzzy Association Rule Mining (F-ARM) based on exhaustive search and the second algorithm also performs association rule mining on the fuzzy partitioned data but will employ a filtered beam search. Finally the third algorithm which is based on the classification and regression trees (CART) will also be presented in this chapter, which discovers hierarchical rule structures based on the decision trees.

3.1. Problem Definition

The main objective of the feature subset selection problem is to maximize the predictive accuracy of the classification while minimizing the number of features used for the classification task.

X : Set of features

n : Number of features

$G(.)$: Function evaluating the classification performance of the selected feature subset.

$$x_i = \begin{cases} 1 & \text{if } x \text{ is selected in the feature subset, from set of all features} \\ 0 & \text{otherwise} \end{cases}$$

Without any constraints added, the feature subset selected problem can be modeled as follows:

$$\max G(x_1, x_2, x_3, ..., x_n)$$
$$\min \sum_{i=1}^{n} x_i \quad x_i \in X$$

3.2. Algorithm for Fuzzy Association Rule Mining (F-ARM)

The F-ARM algorithm aims to discover all fuzzy item sets that satisfy a predetermined threshold support level, with the highest possible confidence. The proposed search strategy is breadth-first, i.e., the algorithm first searches all of the possible item sets that can be generated from all parent item sets that have the same number of elements. The pruning strategy for mining is *a priori* based. That is to say if an item set does not have a support value above the support threshold, any item set that includes that set will be treated as unsatisfactory support level.

Fuzzy Partitioning

In order to mine association rules from data for classification purposes one should first partition the data into clusters. However, it is difficult to strictly label an instance in the data, when the instance is near the boundary of data clusters. Hence, rather than arbitrarily assigning such instances to clusters, a weighting procedure which represents the degree of belonging to clusters would be a better representation of the reality. In this thesis we utilized the Fuzzy C-Means clustering algorithm developed by Bezdek [64]. The FCM algorithm is similar to the well-known k-means clustering procedure which is adapted to the context of fuzzy set theory. In this theory, an instance is a member of every possible cluster with certain "*membership degrees*".

The algorithm tries to minimize the cost function derived from the dissimilarity between data points and cluster centers. The cost function is as follows:

$$J_q(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^{q} d(x_i, \theta_j)$$

where vector θ represents cluster centers, u_{ij} represents the membership degree of point i to cluster j, q is the fuzzification parameter representing the compactness of the clusters; *d(i,j)*

function returns the distance between i and j. As constraints, the properties of the membership degrees are integrated to the model.

The fuzzification parameter determines the membership degree boundaries. As q decreases to 1 ($q>1$), the algorithm resembles traditional k-means clustering algorithm. FCM assumes that the total membership degrees of an instance for each cluster is 1 and the total membership degrees of a cluster for each instance must be strictly greater than zero and strictly less than total number of instances.

In order to determine the cluster center, coordinates that minimize the cost function, gradient of $J(U, \theta)$ with respect to θ is taken. Since the solution cannot be stated in closed form [65], an iteration based procedure is applied to converge to the minimum value of the cost function.

At each iteration, as the cluster centers are updated, $J(U, \theta)$ is recalculated according to the revised membership degrees based on new cluster centers. It is important to note that fuzzy clustering gives information about the degree of belonging to the clusters of any point. It is however more computationally costly than hard clustering procedures.

Membership degrees, i.e., $u_{ij,}$ of each instance, to each cluster are computed via the FCM algorithm. After the fuzzy clustering phase, rule generation step begins. In this phase, instances belonging to the same class are searched. Our approach uses a fast search strategy to generate rules with higher support. For a predetermined threshold for membership levels, our search strategy finds rules that have maximum confidence and are above a predetermined support threshold.

Since the data is clustered with fuzzy sets, we need to define our support and confidence measures for the the fuzzy partitioned data. In this algorithm we stick to the definition of support, and directly apply it to fuzzy membership values. The support of an item i in class c is the sum of all the membership degrees of instances in class c to item i. Equation (3.1) explains the computing, m is the number of instances that belong to class c, $u_{t,i}$ represents the membership degree of instance t to item i.

$$s(i,c) = \sum_{t=1}^{m} u_{ti} \tag{3.1}$$

Note that, one should decide the ANDing operator that will be used when computing the support of item sets with more than one item. There are various different t-norms available in fuzzy set and logic literature for this purpose. A commonly used one is the *min* operator.

Briefly speaking, if an instance belongs to the item *i* to a degree of *a*, and item *j* to a degree of *b*; then that instance belongs to item set (*i* AND *j*) to a degree of $\min\{a,b\}$. Equation (3.2) gives the calculation for the support of an item set, where X refers to the item set.

$$S(X,c_1)=\sum_{t=1}^{m}\min_{i}\{u_{ti}\mid i\in X\} \tag{3.2}$$

The F-ARM algorithm has two input parameters; namely, the *support_threshold* and the *number_of_output_rules*. The first input parameter, namely, the *support_threshold* is a determinant of the number and the size of the discovered item sets, which defines the support level under which an item set is not worth to discover, even if it is highly confident. The value of this parameter is closely related with the desired rule structure. However, as the most representative items are important for discovering rules that define the whole class label, it is important to keep the *support_threshold* as high as possible, which also brings advantages for less computing. The second parameter, namely, the *number_of_output_rules* defines the number of rules that will be selected from all discovered rules at the end, which will be utilized in order to conduct the classification task. These rules will be the most confident ones among all discovered association rules.

The rule structure that is discovered by this algorithm is as follows: the antecedent being the item set discovered and the consequent is the class label of which the item set is mined e.g. IF items *a* AND *b*, THEN class label *c*. Therefore, in order to determine the confidence of the item sets, as the definition proposes, we need to compute the support of the item set in the instance sets defined by other class labels. Equation (3.3) represents the calculation of confidence of an item set under a specific class label c1, where C represents the set of all class labels.

$$C(X,c_1)=\frac{S(X,c_1)}{S(X,c_1)+\sum_{i}S(X,c_i\mid c_i\in C,c_i\neq c_1)} \tag{3.3}$$

Our algorithm separately mines instance sets under the same class label. However, when computing the confidence, the algorithm determines the support of the item sets in other instance sets. After running F-ARM in every instance subset with a different class label, the algorithm outputs *number_of_output_rules* number of rules for each class label defined in the instance set. The decision of the test instance's class label is again based on the fuzzy set theory. This decision is made along two dimensions: the first one is the membership degree of the test instance to the item set, *i.e.*, the antecedent of the rule; the second one, the confidence

of the rule. The membership degree of the test instance is computed in ANDing approach; we take the minimum of the membership degrees of the test instance to the items that belong to the antecedent of the rule. Equation (3.4) explains the determination of the membership degree u_{tr} -the membership to test instance *t* to the rule *r*, where *X* represents the set of items that construct the antecedent part of the rule.

$$u_{tr} = \min_i \{u_{ti} \mid i \in X\} \tag{3.4}$$

The second dimension is the confidence of the rule. Confidence simply measures the fraction where the rule is observed in the whole instance set. Therefore we treat confidence as a possibility measure, define the expected membership of the test instance to the testing rule as follows:

$$E\_u_{tr} = u_{tr} C(X, c_1) \tag{3.5}$$

where X is the antecedent item set, c_1 is the consequent of the rule.

The discovered rules of the same class are connected to each other with the logical operator OR. Therefore, since we have used the MIN function to compute the degree of logical relation AND, this time we take the maximum (MAX) of the expected degree of the membership of the test instance to determine the degree of membership of the test instance to the instance set that is mined. *R* is the set of all rules that is selected to define class c_1.

$$u_{tc_1} = \max_r \{E\_u_{tr} \mid r \in R\} \tag{3.6}$$

The general framework of the algorithm can be given in Figure 3.1

Algorithm for Fuzzy Association Rule Mining	
Step 1	Fuzzy partitioning the data
	Convert gene expression profiles to variables (items) representing high and low gene expression
Step 2	Compute support of every item

	Add membership degrees of each instance
Step 3	Prune items with unsatisfactory support
	As in A Priori Assumption they can never make up satisfactory item sets
Step 4	Construct all possible item sets by combining item sets from previous level and items left from initial pruning
	Perform breadth-first search by inserting items into item sets
Step 5	Determine the support of produced item sets
	Sum of minimum of membership degrees of each instance
Step 6	Prune item sets with unsatisfactory support
	Support values less than input parameter *threshold_support*
Step 7	Determine the confidences of rules derived from item sets
	Store *number_of_output_rules* item sets with maximum confidence (delete item sets with lower confidence from previous iterations, if necessary)
Step 8	Return to Step 4
	Terminate if no item sets with support higher than *threshold_support* could be discovered

Figure 3.1 General Framework for Algorithm F-ARM

The Algorithm starts with a pre-processing step. All training instances are applied the fuzzy c-means algorithm on each gene and the fuzzy partitioned items are generated. Every gene is represented by two items referring to low and high gene expression. According to this membership data, support for each item is calculated. Items with support value above *threshold_support* are selected for the construction of the search tree. Since the A Priori assumption is accepted, any item with lower support cannot make up item sets with satisfactory support (i.e. greater than *threshold_support*).

After that, item sets with 2 items are generated. The supports of these item sets are calculated by (3.2). If an item set has support value lower than *threshold_support*, this item set is omitted from the construction of larger item sets. The confidence of the rules that include discovered item set as the antecedent is calculated as in (3.3). In order to determine confidence, the support of the item set in other instance subsets with different class labels should be computed. Item sets that are included in maximum confident rules are stored in a vector which can keep the *number_of_output_rules* number of item sets.

Item sets that are discovered in the previous iteration are combined with items that have support values above *threshold_support* i.e. initially pruned. Their supports and confidence of rules are calculated. As stated in previous paragraph, if any rule with discovered item set has a confidence value higher than confidence, the minimum confident rule item set stored in the

vector, the item set with the minimum confidence is taken out of the vector, and the discovered item set is inserted. This procedure continues until the most confident item set discovered has less confidence than the confidence of the minimum confident rule item set stored in the vector. Consequently, the number of rules stored in the vector never changes, as any rule is inserted, another rule s taken out. The algorithm iterates in this fashion until no item set can be discovered with support value above *threshold_support*.

An Example on Algorithm F-ARM

Data for F-ARM has 10 items (i.e. 5 genes) and 6 instances Table 3.1.

Table 3.1 Data set for counter example on F-ARM

		instances					
		cancer	**cancer**	**cancer**	**normal**	**normal**	**normal**
Items	**0**	0.8	0.6	0.4	0.4	0.3	0.3
	1	0.2	0.4	0.6	0.6	0.7	0.7
	2	0.4	0.1	0.6	0.6	0.8	0.3
	3	0.6	0.9	0.4	0.4	0.2	0.7
	4	0.2	0.2	0.3	0.4	0.3	0.2
	5	0.8	0.8	0.7	0.6	0.7	0.8
	6	0.4	0.6	0.8	0.3	0.4	0.4
	7	0.6	0.4	0.2	0.7	0.6	0.6
	8	0.5	0.8	0.6	0.2	0.2	0.3
	9	0.5	0.2	0.4	0.8	0.8	0.7

Mining cancer data:

Set *threshold_support*=1.8

Set *number_of_ouput_rules*=2

1. Calculate fuzzy support for each item.

$s(0,C)$= 0.8+0.6+0.4=1.8

$s(1,C)$=0.2+0.4+0.4=1.2

$s(2,C)$=1.1; $s(3,C)$=1.9; $s(4,C)$=0.7; $s(5,C)$=2.3;

$s(6,C)$=1.8; $s(7,C)$=1.2; $s(8,C)$=1.9; $s(9,C)$=1.1

2. For selected threshold, determine the items that have fuzzy supports greater than or equal to the *threshold_support*.

The items above threshold are {0, 3, 5, 6, 8}.

3. Construct the item sets (Table 3.2)

Table 3.2 Constructed item sets with 2 items

	0	**3**	**5**	**6**	**8**
0	X	(0,3)	(0,5)	(0,6)	(0,8)
3	X	X	(3,5)	(3,6)	(3,8)
5	X	X	X	(5,6)	(5,8)
6	X	X	X	X	(6,8)
8	X	X	X	X	X

4. Calculate the fuzzy support of the constructed item sets and eliminate the ones that have support less than the support-threshold. Determine the support of an item set using

Min operator as the t-norm. Take the item sets that have support

greater than or equal to the *threshold_support* for the next iterations.

$S((0,3),C)=Min\{0.8,0.6\}+Min\{0.6,0.9\}+Min\{0.4,0.4\}=1.6$

$S((0,5),C)=1.8$; $S((0,6),C)=1.4$; $S((0,8),C)=1.5$; **$S((3,5),C)=1.8$**;

$S((3,6),C)=1.4$; $S((3,8),C)=1.7$; $S((5,6),C)=1.7$; **$S((5,8),C)=1.9$**; $S((6,8),C)=1.6$.

The item sets above threshold are {(0,5); (3,5); (5,8)}

5. Calculate the confidence of the item sets that are above the threshold

$$C((0,5),C)=\frac{S((0,5),C)}{S((0,5),C)+S((0,5),H)}$$

$$C((3,5),C)=\frac{1.8}{1.8+(0.4,0.3,0.3)}=0.6428$$

$$C((5,8),C)=0.5806$$

$$C((5,8),C)=0.7307$$

6. Record the most 2 confident rules with respect to their confidence levels.

The confident rules are {(5,8), 0.7307; (0,5), 0.6428}

7. Construct the item sets

Table 3.3 New item sets with 3 items

	(0,5)	**(3,5)**	**(5,8)**
0	X	(0,3,5)	(0,5,8)
3	(0,3,5)	X	(3,5,8)
5	X	X	X
6	(0,5,6)	(3,5,6)	(5,6,8)
8	(0,5,8)	(3,5,8)	X

8. Calculate the support for the item sets

$S((0,3,5),C)=1.6$; $S((0,5,8),C)=1.5$; $S((3,5,8),C)=1.7$;

$S((0,5,6),C)=1.0$; $S((3,5,6),C)=1.4$; $S((5,6,8),C)=1.6$.

9. No item sets are identified that have a support level greater than

or equal to the *threshold_support*.

Therefore the most confident two rules discovered up to this point are as follows:

IF item 0 AND item 5, THEN cancer; with confidence 0.6428.

IF item 5 AND item 8, THEN cancer with confidence 0.7307.

Similarly, mining the data regarding to the *normal* instances yields:

IF item 1 AND item 5 AND item 7 AND item 9, THEN NORMAL with confidence 0.75.

These rules can be used to classify an instance that has an unknown label.

Consider the instance tabulated in the following table;

Table 3.4 Test Instance Vector

		test instance
Items	0	0.2
	1	0.8
	2	0.6
	3	0.4
	4	0.3
	5	0.7
	6	0.4
	7	0.6
	8	0.2
	9	0.8

Rule 1: IF item 0 AND item 5, THEN cancer; with confidence 0.6428.

u_{11}=min{0.2, 0.7}=0.2

E_u_{11}=0.2*0.6428=0.1285.

Rule 2: IF item 5 AND item 8, THEN cancer with confidence 0.7307.

u_{12}=min{0.7, 0.2}=0.2

E_u_{12}=0.2*0.7307=0.1461.

Membership of the test instance 1 to class CANCER:

u_{1C}=max{0.1285, 0.1461}=0.1485.

Rule 3:

IF item 1 AND item 5 AND item 7 AND item 9, THEN NORMAL with confidence 0.75.

u_{13}=min{0.8, 0.7,0.6,0.8}=0.6

E_u_{13}=0.6*0.75=0.45.

Membership of the test instance 1 to class NORMAL:

u_{1C}=max{0.45}=0.45

.

Based on these figures, one can conclude that the test instance is more NORMAL than CANCER.

3.3. Filtered Beam Search with Child Width for Association Rule Mining

The second algorithm presented is very much alike the fist algorithm F-ARM. However in this case, rather than an exhaustive search on every possible item set, a greedy pruning strategy will be adopted when mining the item sets.

Filtered beam search is a special case of branch and bound algorithms. Instead of performing and enumeration, *i.e.,* searching every possible solution, by selecting the most promising ones generated from that node, beam search limits the number of generated solutions from a specific node. As determining which nodes should be eliminated is costly, filtered beam search, firstly filters possible solutions according to a local and less computational costly evaluation function, and then selects beam width number of nodes according to a global evaluation function in order to continue iterations i.e. enlarging the search tree.

As proposed by Aktürk and Kılıç [66], our next algorithm searches for association rules by conducting a beam search tree with child-width constraints. As the name implies, child-width brings limits to the number of generated nodes from each parent node.

Algorithm Beam Search with Child Width Constraint	
Step 1	Fuzzy partitioning the data
Step 2	Compute support of every item
Step 3	Prune items with unsatisfactory support
Step 4	Set items with satisfactory support as the initial item set to be enlarged

Step 5	Generate potential child nodes by enlarging each item set with 1 additional item	
		filter item sets that have average support *threshold_support*
	Step 6	calculate support of filtered item sets
	Step 7	Take parent nodes with no potential children emanating, into solution set
	Step 8	If there are no potential children left from filtering, TERMINATE algorithm
		If size of potential children are below beam_width,
		Update beam_width as
	Step 9	min {child_width*number of parent nodes, size of potential children}
	Step 10	take beam_width number of item sets in a way that at most child_width number of item sets can be taken from a parent node. Deterine them as the new parent nodes
Step 11	Return to Step 5	

Figure 3.2 Steps for filtered beam search with child-width constraint

Algorithm Beam Search with Child-width Constraint
Initialization
support={};
candidate_parents={};
support_parents={};
parents={};
selected={};
beam_width;
child_width;
number_of_instances;
number_of_children={};
confidence={};
indexes={0};
solutions={};
End initialization
For each item
    Compute support(item)
    If support(item) ≥ minsup
        Insert item **into** *selected*
    End If
End
Parents=*selected*
While iterations
    For each item set in *parents*
        index=0;
        For each item in *selected*
            Combine (item set+item) = item set
            Compute average support of the item set
            If average_support(item set) ≥ threshold_support
                Insert (item set) **into** *candidate_parents*
                index=index+1;
                Compute support (item set)
                Insert support (item set) **into** *support_candidate*
            End
        End
        If index==0
            Insert *parents*(item set) **into** *solution*
        Insert [index + *indexes*(size(*indexes*)] **into** *indexes*
    End

```
        If size(support_candidate)>0
                For each item set in candidate_parents
                        Compute average_confidence(item set)
                        Insert average_confidence(item set) into confidence
                End
                If size(confidence) < beam_width
                        beam_width=min{child_width*number of parent nodes, size of potential children}
                End
                For i=1: length(parent_support)
                        children(i)=0;
                End
                index=0;
                Clear parents, support_parent,
                While index < beam_width
                Determine item set with maximum average confidence in confidence
                Determine parent of item set using indexes
                                number_of_children(parent(item set))= number_of_children(parent(item
                                        set))+1
                If number_of_children(parent(item set)) < child-width
                        Insert candidate_parents (item set) into parents;
                        Insert support_candidate(support(item set)) into parents;
                End
                Erase item set from candidate_parents;
                Erase support(item set) from support_candidate
                Erase average_confidence(item set) from confidence
                Modify number_of_children according to item set erase
                End
                Clear candidate_parents
                Clear support_candidate
                Clear confidence
                Clear children
                Clear number_of_children
        End If
End While
```

Figure 3.3 Algorithm Filtered Beam Search with Child-width Constraint

In this algorithm we have used the tools for measuring the talent of association rules we applied in algorithm F-ARM. All nodes emanating from the parent node are firstly determined. Then, according to our evaluation of average support –which is computationally less costly compared to other measures (support and confidence) due to not passing the data-nodes that have average support less than *threshold_support* are filtered out.

Lower bound and upper bound for the fuzzy support of an item set

Definitions: Let X be an item set and i be an item where $i \neq j, \forall j \in X$. *S(X)* and *s(i)* represent the previously calculated supports for item set X and item i. The number of instances that are used for support calculation is n.

Let membership degrees of item set X be $u_{1X}, u_{2X}, ..., u_{nX}$ and the membership degrees of item i be $u_{1i}, u_{2i}, ..., u_{ni}$.

Proposition 1: The lowerbound for the value of the support of the union of item set X and item i is:

$$L\_S(X \cup i) = S(X) + S(i) - n$$

Proof: The real support of item set is defined as

$$S(X \cup i) = \sum_{t=1}^{n} \min\{u_{tx}, u_{ti}\}$$

Assume that

$$\sum_{t=1}^{n} u_{tX} + \sum_{t=1}^{n} u_{ti} - n > \sum_{t=1}^{n} \min\{u_{tx}, u_{ti}\}$$

$$\sum_{t=1}^{n} \max\{u_{tx}, u_{ti}\} > n$$

The above inequality is infeasible, since the highest value of a membership degree is 1.

Proposition 2: The upperbound for the value of the union of item set X and item i is:

$$U\_S(X \cup i) = \min\{S(X), S(i)\}$$

Proof: The real support of item set is defined as

$$S(X \cup i) = \sum_{t=1}^{n} \min\{u_{tx}, u_{ti}\}$$

Assume that

$$\min\left\{\sum_{t=1}^{n} u_{Xt}, \sum_{t=1}^{n} u_{it}\right\} < \sum_{t=1}^{n} \min\{u_{Xt}, u_{it}\}$$

Let $\sum_{t=1}^{n} u_{Xt} < \sum_{t=1}^{n} u_{it}$

$$\sum_{t=1}^{n} u_{Xt} < \sum_{t=1}^{n} \min\{u_{Xt}, u_{it}\}$$

$$\sum_{t=1}^{n} u_{Xt} = \sum_{t=1}^{n} \min\{u_{Xt}, u_{it}\} + \sum_{t=1}^{n} f(t)(u_{Xt} - u_{it})$$

where $f(t) = \begin{cases} 1 & u_{Xt} > u_{it} \\ 0 & otherwise \end{cases}$

$$\sum_{t=1}^{n} f(t)(u_{Xt} - u_{it}) < 0$$

The above equation is infeasible since left hand size is always positive. The situation holds for $\sum_{t=1}^{n} u_{Xt} > \sum_{t=1}^{n} u_{it}$ and $\sum_{t=1}^{n} u_{Xt} = \sum_{t=1}^{n} u_{it}$ case.

Definition: Average support of item set $(X \cup i)$ is

$$A\_S(X \cup i) = \frac{L\_S(X \cup i) + U\_S(X \cup i)}{2}$$

From all left (potential) nodes emanating from all parent nodes in the same level of the search tree, beam-width number of nodes should be selected to be the new parent nodes. This selection is done according to the confidence of the rules whose antecedent is the item sets to be selected. Beam-width number of item sets from which generated rules has the maximum confidence is chosen according to "child-width" constraint. This constraint prevents selecting child nodes which are mostly emanating from the same parent node. The application of this constraint is as follows: if only there are less than the child-width number of children nodes emanating from its parent node, a child can be selected as a child node. As the selection is done, and all the children nodes are determined, the children nodes that are generated from the same parent node do not exceed child-width.

As the final step of the beam search algorithm, the last step of F-ARM algorithm has been directly utilized into beam search algorithm. The most confident rules are selected to conduct the classification of the testing instance. The number of the rules selected is determined by the parameter *number_of_output_rules*.

3.4. Decision Tree Construction for Classification

Another rule generation scheme is the hierarchical construction of features for the classification of the data. Decision trees are mostly applied in data mining for classification problems. In spite of the threat of overfitting, Slonim [59] points out the simplicity of logic behind the construction of decision trees complex structures can be modeled easily by decision trees. In order to apply decision tree structure to microarray data, Classification and Regression Trees [60] is used. CART constructs a decision tree, in which each node represents a gene with a split value. For each node, the gene that is most successful in splitting the data in two most possible homogeneous partitions is selected, and two branches of nodes

represent the partitioned instance subsets. Algorithm iteratively "grows" the tree until every path in the tree refers to a decision rule which concludes with a homogeneous instance subset.

For each node, each gene is tested with its optimal split point. For all the values that the gene is taken, the optimal split point that divides the values set into the two most homogeneous must be determined. In this study, candidate split values are calculated as the arithmetic average of all the two consecutive values. The decision for the best split is done when the performance evaluation function takes current node *t* and split value *s* as parameters (3.7). *L* and *R* indicate the left and right branches of the current node. P_L and P_R represent the fraction of instances that belongs to the left and the right branch respectively. For each class *j*<=*m*, the function adds the difference of the fraction of instances that belongs to right and left branches.

$$\Phi(s \mid t) = 2P_L P_R \sum_{j=1}^{m} \left| P(C_j \mid t_L) - P(C_j \mid t_R) \right| \qquad (3.7)$$

Another performance measure for the best split is selected to focus on impurity. For choosing the gene that most homogenously partitions, the data is determined by entropy. Jin and Agrawal [67] used entropy to select split value for the decision tree construction. Their approach focuses on the change on the entropy and looks for the feature and split value that the most decreases impurity.

```
Algorithm CART
Initialize
        Split_values={}
        Genes={}
        Samples_node=All samples in dataset (as a vector)
        Samples_level={}
        Dummy={}
End Initialize
[gene, split, left_samples, right_samples]= find_optimal_splitting (Samples_node)
Insert left_samples into Samples_level
Insert right_samples into Samples_level
Insert gene into genes
Insert split into Split_values
Clear Samples_node
While Samples_level≠{}
        If all vectors in Samples_level include samples of the same class
                Samples_level={}
        else
                For each vector in Samples_level
                        Samples_node= vector in Samples_level
                        [gene, split, left_samples, right_samples]= find_optimal_splitting(Samples_node)
                        Insert left_samples into Dummy
                        Insert right_samples into Dummy
                        Insert gene into genes
                        Insert split into Split_values
                End
                Clear Samples_level
                Samples_level=Dummy
                Clear Dummy
        End
End while
Function find_optimal_splitting
        For each gene that has not been used in any node so far
                For any average value of two successive samples in Samples_node
                        Compute performance_measure;
                End
        End
        Find the gene and the split value with optimal maximum performance_measure:
                gene, split;
        For each sample in Samples_node
                If expression data (gene, sample) < split
                        Assign sample to left_samples;
                Else
                        Assign sample to right_samples;
        End
End Function
```

Figure 3.4 Algorithm CART

For the 2-class case, before the splitting of the instance, entropy is calculated as in (3.8), after the splitting feature splits the instances, the left and right branch instances are determined. In order to find the decrease in impurity by splitting a node, the entropy of the left branch and the right branch is multiplied by their fractions (P_L and P_R) respectively, and

subtracted from the entropy before splitting (3.9). The feature and the split value that maximizes this decrease is selected at the current node.

$$Entropy(s \mid t) = -P(C_1 \mid t\ ) * \log_2 P((C_1 \mid t\ ) - P(C_2 \mid t\ ) * \log_2 P((C_2 \mid t\ ) \quad (3.8)$$

$$I\_Decrease(s \mid t) = Entropy(s \mid t) - P_L * Entropy(s \mid t_L) - P_R * Entropy(s \mid t_R) \quad (3.9)$$

CHAPTER 4

EXPERIMENTAL ANALYSIS

In order to test the performance of the proposed algorithms both in terms of accuracy and computational time an experimental analysis is conducted. For this purpose we utilized three different data sets. These are the publicly available Colon Cancer Data*, Leukemia Data† and the well known Iris Data. First we will provide the detail of the data sets, later we will present the results of the experimental analysis.

4.1. The Colon Cancer Dataset

The colon cancer dataset is composed of 22 normal and 40 cancer instances, with 2000 genes that are determined to be significantly differentially expressed between the normal and cancer classes. This dataset was first introduced in the paper by Alon *et. al.* [68], where they examined the correlation among genes, and utilized a clustering algorithm on the data. In addition, their data included 6500 genes which were reduced to 2000 in the following studies. Their clustering scheme misclassified 8 instances which was explained by the tissue composition of the misclassified instance. Li *et. al.* [79] used genetic algorithms with a k-nn classifier to measure the performance of the chromosome. They misclassified only 1 test instance, when 40 of the instances were used for training while the rest was for testing. Another study conducted by Tan *et. al.*, [80] used the C4.5 algorithm that implemented decision trees and reached 95% predictive accuracy. In their studies, Kucukural et. al. [26] reached 98.38% accuracy with ten fold cross validation and Guyon et. al. [8] misclassified 3 test instances out of 31, by using the SVM classifier.

* Data is available in http://microarray.princeton.edu/oncology/affydata/index.html
† Data is available in http://www.broad.mit.edu/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43

4.2. The Leukemia Dataset

Introduced by Golub *et. al.* in 1999 [69], this dataset consists of different instances that make up separate training and testing sets. The primary objective of the studies related to this dataset is the prediction of the testing instances' type of acute leukemia as ALL (acute lymhoblastic leukemia) or AML (acute myeloid leukemia)[‡].

The training dataset includes 27 ALL and 11 AML bone narrow instances. The number of genes represented in the dataset is 7129. The testing dataset consists of 20 ALL and 14 AML instances. Golub *et. al.*'s study noted that the testing dataset includes a "broader collection of instances" than do the training instances. Golub *et. al.*'s method was able to classify the testing dataset with 100% accuracy using 50 genes.

Jirapech-Umpai *et. al.* [70] investigated the classification performance of evolutionary algorithms on leukemia dataset. On one hand, their algorithm gave high training accuracy; on the other hand, it gave poor performance with the testing dataset. They found 68% at the best case, reaching 98% accuracy in the training phase.

Another study, Yuhang *et. al.* [71], compared many different feature selection and classifier construction methods, and to find 100% accuracy for the leukemia dataset using 5 genes. Bø *et. al.* [72] investigated the performance of their feature selection methodology on leukemia dataset. The prediction accuracy of 2-fold validation scheme stays around 96% as the number of features selected are more than 30 genes. Guyon *et. al.* [8] claimed that the training set and the testing set is significantly different than each other, resulting difference in training leave-1-out and testing accuracy. Their best result was a 100% accurate classification of testing instances with 16 genes.

4.3. The Iris Flower Dataset with Noise

Introduced by Fisher [73] in 1936, the iris flower dataset is one of the primer datasets in the pattern classification literature[§]. It includes 4 features represented with positive real numbers, which carry information on 150 instances. The instances belong to 3 classes: setosa, versicolor, and virginica. The structure in this dataset is different, that is to say, the ratio of the number of instances to the number of features is much higher than the ratio in the microarray

[‡] Data is available in http://www.broad.mit.edu/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43
[§] Data is available in http://en.wikipedia.org/wiki/Iris_flower_data_set

datasets (0.031 in colon dataset, 37.5 in iris dataset). However, most microarray datasets carry information of two classes, often normal and tumor; thus, linear simple classification methods may be successful to obtain higher prediction accuracies. As the scatter diagrams of paired features in Figure 4.1 reveal, it is not possible to separate the data with a single classifier line or labeling the feature's value as "high" or "low". As Ben-Hur [74] states, "setosa" labeled instances can be separated from the others with a single line, whereas, the other labeled instances cannot be linearly separable, showing high overlapping.

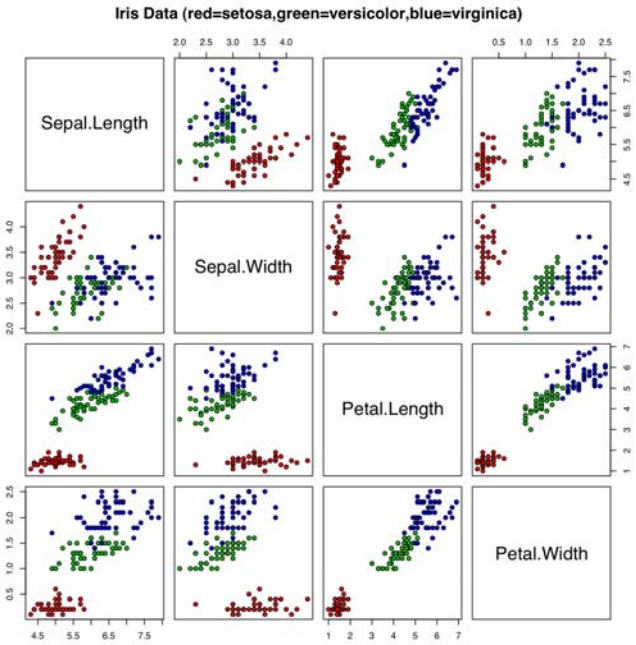

Figure 4.1 Scatter plots of the Iris Flower Dataset[**]

Tsang *et. al.* [75] reached high accuracies with their algorithm that used features named petal length and petal width. Mylonas *et. al.* [76] used k-nn classifier when constructing a hierarchical clustering structure, and found 82% accuracy for iris flower dataset. Another study came from Ahmandian *et. al.* [77] which reached 97% testing accuracy in iris flower dataset. They eliminated the phase of feature subset selection and applied multi-objective optimization principles to evolutionary computing.

In order to test F-ARM about choosing only the relevant features, that is to say, to conduct a successful feature subset selection, randomly distributed 46 features were added to the 4-featured iris flower dataset, resulting in a dataset with 4 relevant features out of 50.

[**] Figure retrieved from http://upload.wikimedia.org/wikipedia/commons/e/ea/Anderson%27s_Iris_data_set.png

4.4. Algorithm F-ARM – Algorithm for Fuzzy Association Rule Mining

Fuzzy Partitioning

In order to apply the fast search of association rules, the colon cancer dataset is first fuzzified by fuzzy partitioning. The expression levels of each gene are partitioned into two fuzzy sets, representing high and low expression levels. Once the expression of a gene is partitioned, the gene is represented in the fuzzy data as two "items".

For fuzzy partitioning of every gene, the general framework for fuzzy clustering introduced in methodology section is applied. Application is done with the fuzzy c-means clustering function of MATLAB. Three different fuzzification parameters are used, namely 1.2, 2 and 2.8. Note that, 2 is widely used as the fuzzification parameter in the literature[78]. To increase the effect of fuzzification, the fuzzzification parameter 2.8 is selected for further operations. Figure 4.2 shows the effect of fuzzification on the membership degrees of instances.

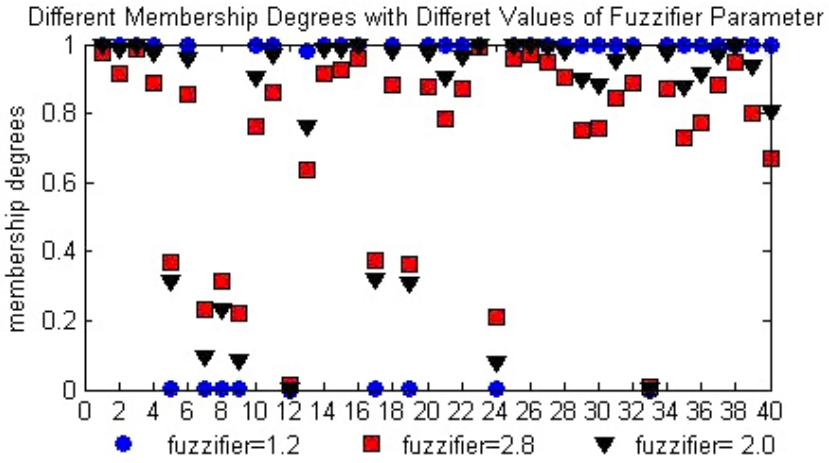

Figure 4.2 Effect of the fuzzification parameter on membership degrees of the same data (taken from colon cancer dataset)

Note that, the clusters converge to a "crisp" set structure if the fuzzification parameter is close to 1 and as the fuzzification parameter increases, membership degrees are more "fuzzified" which means that the membership degrees are more prone to have membership degrees with lower differences among them.

When mining association rules, using the fuzzy partitioning data with a low fuzzification parameter can provide the discovery of a higher number of association rules than

using partitioning data with other fuzzification parameters. However, as the membership degrees are less sensitive to the distances to the cluster centers, it is more likely that the partitioning is prone to risks. With a high fuzzification parameter, it is less likely to discover item sets with a satisfactory level of support and confidence. On the other hand, using a fuzzy partitioning data with a high fuzzification parameter is more appropriate for obtaining the worst case membership degrees.

Changing the ANDing Operator

Taking the product of the elements as the ANDing operator may be expected to increase classification accuracy. The main reason is that multiplication carries the information of every term that is involved; i.e. every element of the multiplication affects the outcome without any specific requirement.

Changing the ANDing operator firstly affects the support values of discovered itemsets. That is to say, the support levels obtained by multiplication are always lower than levels obtained by taking minimum. Following is the proof of this proposition:

Proposition 3:

Let $x_1, x_2, x_3, ..., x_n$ be the membership degrees for itemset $I = \{1,2,3,...,n\}$.

Since $0 \le x_i \le 1 \qquad \forall i \in I$

$$\prod_{i=1}^{n} x_i \le \min_{i \in I}\{x_i\}$$

Proof:

Let $x_{\min}$ be the minimum membership degree of the item set *I*, and the remaining elements of the itemset is represented as $K = \{y \mid y = x_i, y \ne x_{\min}, x_i \in I\}$.

$x_{\min} \prod_{i=1}^{n-1} y_i \le \min\{x_{\min}, K\}$ dividing both sides by $x_{\min}$ will yield

$\prod_{i=1}^{n-1} y_i \le 1$ which holds for every $0 \le y_i \le 1$.

Implementation on Colon Cancer Dataset

Fuzzy c-means clustering was applied to the colon cancer dataset to create 2 fuzzy clusters from each gene. After fuzzification, data with every gene represented with two items,

instances with the same class labels are investigated separately for association rule mining. That is to say, for each class label which are normal class --represented with 22 instances-- and cancer class --represented with 40 instances-- membership degrees for every gene are ready to be mined for association rules.

For each test instance, we omitted each instance as the test instance and conducted fuzzy clustering on the expression data without the test instance. Then for each different class, we mined for rules that are maximally confident within that class. Figure 4.3 and Figure 4.4 represent the results of the initial runs of F-ARM. The first 40 test instances are cancer type and the following 22 instances are normal type. Different lines represent final memberships of test instances to the cancer class and normal class. Figure 4.3 shows memberships of test instances when mining cancer and normal classes separately and simultaneously. Predicting the label of the test instance is done by taking the label to which the test instance has the maximum membership. By this method, the calculated prediction of F-ARM with *support_threshold* 0.9, *number_of_output_rules* 1, and *fuzzification parameter* 2.8 on test instances with leave-1-out validation scheme yields 0.742 accuracy, a worse result when compared to those in the literature.

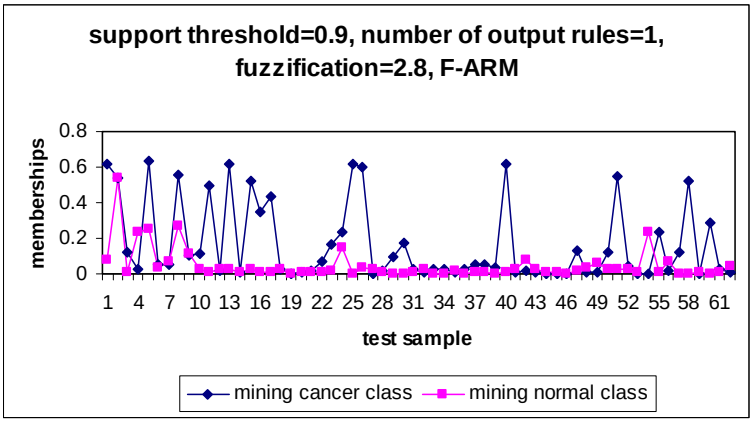

Figure 4.3 Results of F-ARM with given parameters

When the number of output rules increases to 5, the total prediction accuracy decreases to 0.677, memberships can be observed in Figure 4.4. When *number_of_output_rules* are selected as 3, prediction accuracy becomes 0.709.

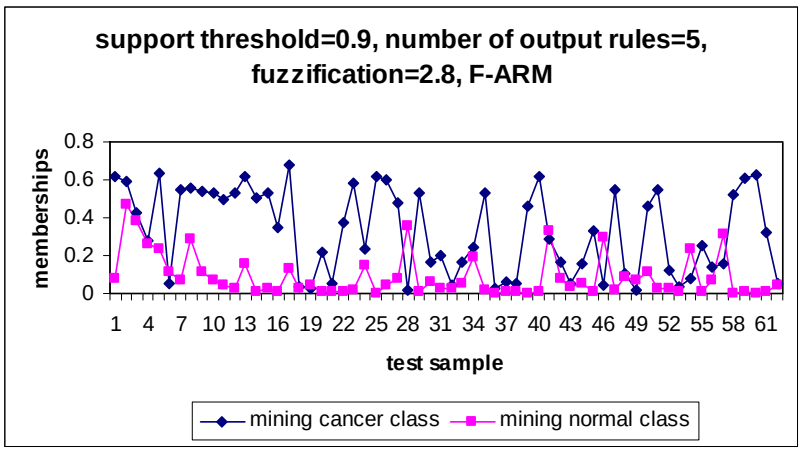

Figure 4.4 Results of F-ARM with given parameters

Prediction accuracy results evaluated by using different parameters can be seen in Table 4.1.

Table 4.1 Results evaluated with different parameter settings

fuzzification parameter=2.8		*number_of_output_rules*		
		1	3	5
threshold_support	0.9	0.742	0.709	0.677
	0.95	0.661	0.581	0.565

When multiplication is used as the ANDing operator, prediction accuracy results evaluated by using different parameters can be seen in Table 4.2.

Table 4.2 Results evaluated with different parameter settings

fuzzification parameter=2.8		*number_of_output_rules*			
		1	3	5	10
threshold_support	0.8	0.5	0.581	0.613	0.71
	0.9	0.806	0.806	0.758	

Implementation on Leukemia Dataset

Fuzzy c-means clustering was applied to the leukemia dataset to create 2 fuzzy clusters from each gene. F-ARM is run on the training dataset and memberships for each instance of the testing set is evaluated using the itemsets discovered by mining the training set. Due to memory limitations, the minimum function as the ANDing operator could only be run with *threshold_support* taken as 0.99; that is to say, taking 0.98 as *threshold_support* resulted with a "lack of memory" error which was declared with an exception in C++

compiler. Memberships evaluated can be seen in Figure 4.5 and accuracy results can be seen in Table 4.3.

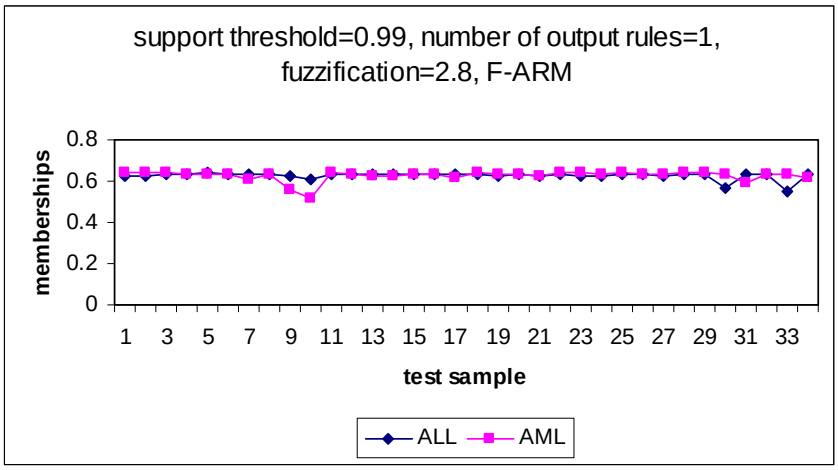

Figure 4.5 Results of F-ARM with given parameters

Table 4.3 Results evaluated with different parameter settings

fuzzification parameter=2.8		*number_of_output_rules*		
		1	3	5
threshold_support	0.99	0.5294	0.4412	0.4706

When ANDing operator is changed to multiplication, the results evaluated with different *threshold_support* values can be seen in Figure 4.6, and accuracies are given in Table 4.4. 100% accuracy is reached with rules generated from 11 genes, when the number of output rules is 1 for each class.

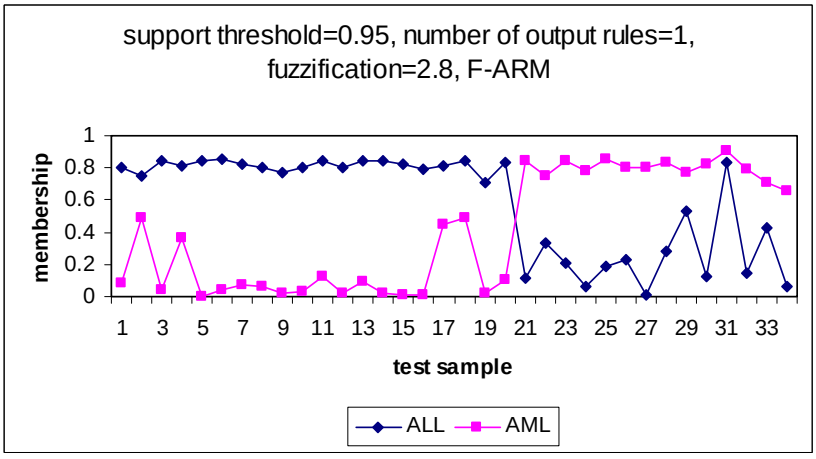

Figure 4.6 Results of F-ARM with given parameters

Table 4.4 Results evaluated with different parameter settings

fuzzification parameter=2.8		*number of output rules*		
		1	3	5
threshold_support	0.95	1.00	1.00	1.00
	0.97	0.9706	1.00	1.00

Implementation on Iris Flower Dataset with Noise

Initially the fuzzification phase that was used for microarray data classification has been applied to the iris dataset with 50 features; that is to say, every feature of the iris dataset was applied fuzzy c-means algorithm and was separated to 2 fuzzy clusters. *Leave-1-out validation* scheme has been applied for accuracy calculations. When F-ARM is applied to this fuzzified data, the memberships appear as in Figure 4.7. This figure reveals that when considering the data having 2 labels, F-ARM is 82% accurate in labeling instances that are "setosa" and "non-setosa". In addition, when "setosa" instances are taken out from the data and F-ARM is applied, accuracy is 94% when labeling "versicolor" and "virginica" instances.

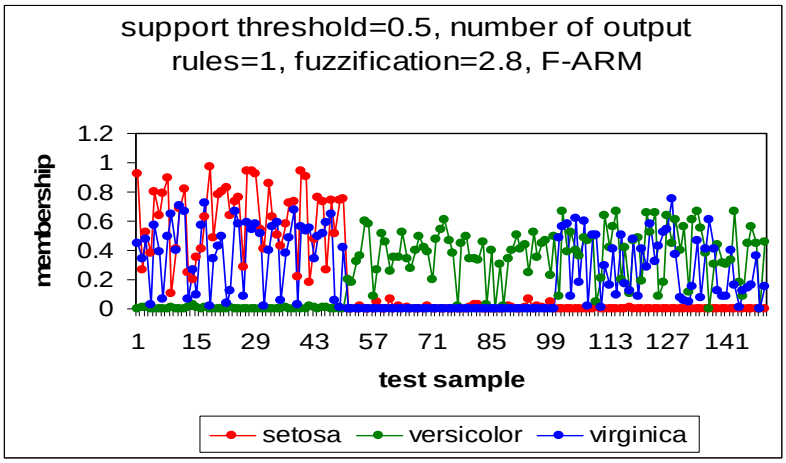

Figure 4.7 Results of F-ARM with given parameters

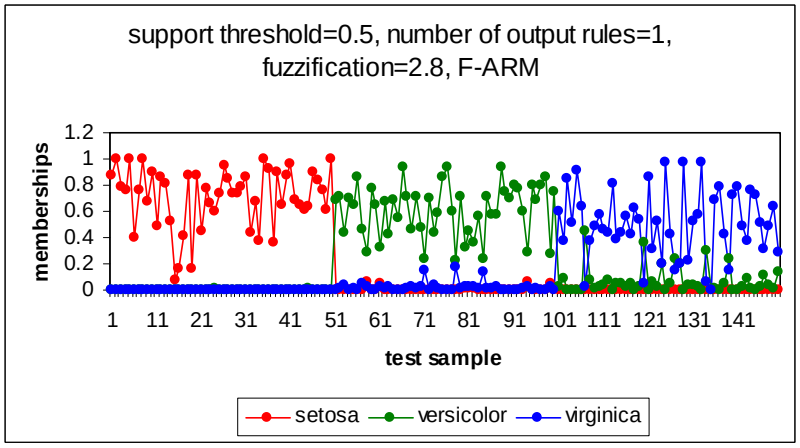

Figure 4.8 Results of F-ARM with given parameters

Secondly, the data was clustered into 3 fuzzy clusters using fuzzy c-means, and F-ARM was applied. Memberships for each class are evaluated with a single run of the algorithm. The memberships evaluated are shown in Figure 4.8. F-ARM selects relevant features that make up high confidence levels. In this data, F-ARM detected features "petal length" and "petal width" to conduct classification. 95.33% accuracy is determined with 2 features of the iris flower data set. Note that experimental study has not been made in this dataset, because item sets that have support greater than *threshold_support* could not be evaluated with support threshold levels greater than 0.5 due to the randomness of the features other than the 4 features from the iris flower dataset. All experiments above have been conducted with computing the product, i.e. multiplication of membership degrees as the ANDing operator.

4.5. Filtered Beam Search with Child-width Constraint for Association Rule Mining

The same fuzzy partitioning scheme is conducted before the mining algorithm. We do not conduct any experiments on the iris dataset with noise because there is a much less number of features than on the other datasets to be tested.

Implementation on Colon Cancer Dataset

Leave-1-out validation scheme has been used to test the accuracy of the beam search algorithm. The parameter set for *beam_width* is {10, 20, 50, 100}, the parameter set for *child_width* is {2, 5, 10, 20}. The parameter set for the number of output rules to be chosen is (*number_of_output_rules*) {1, 3, 5, 10, 20}. Table 4. 5 gives accuracy ratios when support threshold is 0.9. The first and second coloumns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as minimum.

Table 4. 5 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.435	0.435	1.105	50	2	1	0.452	0.452	1.112
10	2	3	0.565	0.516	1.105	50	2	3	0.565	0.532	1.115
10	2	5	0.581	0.468	1.105	50	2	5	0.581	0.468	1.116
10	2	10	0.613	0.532	1.126	50	2	10	0.613	0.532	1.120
10	2	20	0.581	0.532	1.118	50	2	20	0.597	0.565	1.123
10	5	1	0.435	0.435	1.107	50	5	1	0.435	0.435	1.115
10	5	3	0.565	0.516	1.124	50	5	3	0.565	0.532	1.115
10	5	5	0.581	0.468	1.116	50	5	5	0.581	0.468	1.119
10	5	10	0.597	0.516	1.116	50	5	10	0.613	0.532	1.120
10	5	20	0.581	0.532	1.116	50	5	20	0.613	0.581	1.125
10	10	1	0.435	0.435	1.114	50	10	1	0.435	0.435	1.115
10	10	3	0.565	0.516	1.116	50	10	3	0.565	0.516	1.117
10	10	5	0.581	0.468	1.113	50	10	5	0.581	0.468	1.118
10	10	10	0.597	0.516	1.115	50	10	10	0.613	0.532	1.123
10	10	20	0.581	0.532	1.119	50	10	20	0.613	0.581	1.127
10	20	1	0.435	0.435	1.109	50	20	1	0.435	0.435	1.115
10	20	3	0.565	0.516	1.112	50	20	3	0.565	0.516	1.116
10	20	5	0.581	0.468	1.112	50	20	5	0.581	0.468	1.118
10	20	10	0.597	0.516	1.114	50	20	10	0.613	0.532	1.131
10	20	20	0.581	0.532	1.118	50	20	20	0.613	0.581	1.126
20	2	1	0.435	0.435	1.108	100	2	1	0.452	0.452	1.122
20	2	3	0.565	0.532	1.109	100	2	3	0.565	0.516	1.123
20	2	5	0.581	0.468	1.110	100	2	5	0.581	0.468	1.125
20	2	10	0.629	0.548	1.113	100	2	10	0.613	0.532	1.128
20	2	20	0.597	0.581	1.116	100	2	20	0.597	0.565	1.132
20	5	1	0.435	0.435	1.110	100	5	1	0.452	0.452	1.126
20	5	3	0.565	0.516	1.113	100	5	3	0.565	0.532	1.128
20	5	5	0.581	0.468	1.114	100	5	5	0.581	0.468	1.129
20	5	10	0.613	0.532	1.117	100	5	10	0.597	0.516	1.132
20	5	20	0.613	0.565	1.119	100	5	20	0.613	0.565	1.135
20	10	1	0.435	0.435	1.108	100	10	1	0.452	0.452	1.126
20	10	3	0.565	0.516	1.110	100	10	3	0.565	0.532	1.128
20	10	5	0.581	0.468	1.111	100	10	5	0.581	0.468	1.130
20	10	10	0.613	0.532	1.115	100	10	10	0.597	0.516	1.167
20	10	20	0.613	0.565	1.116	100	10	20	0.613	0.565	1.135
20	20	1	0.435	0.435	1.108	100	20	1	0.452	0.452	1.126
20	20	3	0.565	0.516	1.110	100	20	3	0.565	0.516	1.133
20	20	5	0.581	0.468	1.111	100	20	5	0.581	0.468	1.132
20	20	10	0.613	0.532	1.114	100	20	10	0.597	0.516	1.133
20	20	20	0.613	0.565	1.116	100	20	20	0.613	0.565	1.197

Table 4.6 gives accuracy ratios when support threshold is 0.9. The first and second columns represent whether the maximum of the memberships or the average of the

memberships has been used for the computation of the final membership. ANDing operator is selected as multiplication.

Table 4.6 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.516	0.516	1.106	50	2	1	0.516	0.516	1.098
10	2	3	0.468	0.387	1.100	50	2	3	0.468	0.387	1.100
10	2	5	0.468	0.403	1.109	50	2	5	0.468	0.387	1.103
10	2	10	0.548	0.435	1.119	50	2	10	0.565	0.452	1.108
10	2	20	0.516	0.484	1.122	50	2	20	0.548	0.435	1.110
10	5	1	0.516	0.516	1.134	50	5	1	0.516	0.516	1.099
10	5	3	0.468	0.387	1.128	50	5	3	0.468	0.387	1.103
10	5	5	0.468	0.403	1.105	50	5	5	0.468	0.387	1.102
10	5	10	0.548	0.435	1.109	50	5	10	0.548	0.484	1.108
10	5	20	0.516	0.484	1.114	50	5	20	0.548	0.468	1.121
10	10	1	0.516	0.516	1.101	50	10	1	0.516	0.516	1.118
10	10	3	0.468	0.387	1.103	50	10	3	0.468	0.387	1.106
10	10	5	0.468	0.403	1.107	50	10	5	0.468	0.387	1.107
10	10	10	0.548	0.435	1.108	50	10	10	0.548	0.435	1.112
10	10	20	0.516	0.484	1.115	50	10	20	0.548	0.500	1.118
10	20	1	0.516	0.516	1.103	50	20	1	0.516	0.516	1.104
10	20	3	0.468	0.387	1.104	50	20	3	0.468	0.387	1.106
10	20	5	0.468	0.403	1.107	50	20	5	0.468	0.387	1.108
10	20	10	0.548	0.435	1.108	50	20	10	0.548	0.435	1.110
10	20	20	0.516	0.484	1.113	50	20	20	0.516	0.484	1.119
20	2	1	0.516	0.516	1.112	100	2	1	0.516	0.516	1.099
20	2	3	0.468	0.387	1.106	100	2	3	0.468	0.387	1.106
20	2	5	0.468	0.387	1.106	100	2	5	0.468	0.387	1.103
20	2	10	0.548	0.435	1.119	100	2	10	0.565	0.452	1.108
20	2	20	0.548	0.500	1.118	100	2	20	0.548	0.435	1.114
20	5	1	0.516	0.516	1.102	100	5	1	0.516	0.516	1.100
20	5	3	0.468	0.387	1.105	100	5	3	0.468	0.387	1.101
20	5	5	0.468	0.403	1.106	100	5	5	0.468	0.387	1.104
20	5	10	0.548	0.435	1.108	100	5	10	0.565	0.452	1.108
20	5	20	0.516	0.484	1.116	100	5	20	0.548	0.435	1.113
20	10	1	0.516	0.516	1.101	100	10	1	0.516	0.516	1.100
20	10	3	0.468	0.387	1.106	100	10	3	0.468	0.387	1.101
20	10	5	0.468	0.403	1.106	100	10	5	0.468	0.387	1.102
20	10	10	0.548	0.435	1.112	100	10	10	0.565	0.452	1.106
20	10	20	0.516	0.484	1.120	100	10	20	0.548	0.435	1.112
20	20	1	0.516	0.516	1.101	100	20	1	0.516	0.516	1.100
20	20	3	0.468	0.387	1.106	100	20	3	0.468	0.387	1.102
20	20	5	0.468	0.403	1.106	100	20	5	0.468	0.387	1.105
20	20	10	0.548	0.435	1.109	100	20	10	0.565	0.452	1.109
20	20	20	0.516	0.484	1.116	100	20	20	0.548	0.435	1.117

Table 4.7 and Table 4.8 give accuracy ratios when support threshold is 0.8. The first and second columns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as minimum.

Table 4.7 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.629	0.629	18.496	50	2	1	0.581	0.581	18.732
10	2	3	0.548	0.581	18.488	50	2	3	0.516	0.548	18.742
10	2	5	0.532	0.581	18.503	50	2	5	0.500	0.516	18.743
10	2	10	0.581	0.548	18.532	50	2	10	0.484	0.516	18.773
10	2	20	0.645	0.516	18.582	50	2	20	0.500	0.516	18.812
10	5	1	0.613	0.613	19.411	50	5	1	0.581	0.581	19.952
10	5	3	0.581	0.597	19.474	50	5	3	0.565	0.565	19.958
10	5	5	0.581	0.597	19.424	50	5	5	0.532	0.516	19.986
10	5	10	0.532	0.532	19.438	50	5	10	0.516	0.516	20.019
10	5	20	0.548	0.452	19.469	50	5	20	0.565	0.581	20.057
10	10	1	0.581	0.581	20.053	50	10	1	0.597	0.597	20.856
10	10	3	0.581	0.597	20.097	50	10	3	0.581	0.581	20.806
10	10	5	0.565	0.581	20.108	50	10	5	0.597	0.565	20.818
10	10	10	0.532	0.532	20.138	50	10	10	0.516	0.516	20.837
10	10	20	0.565	0.452	20.149	50	10	20	0.548	0.581	20.883
10	20	1	0.581	0.581	20.710	50	20	1	0.597	0.597	21.747
10	20	3	0.581	0.597	20.731	50	20	3	0.581	0.581	21.752
10	20	5	0.565	0.581	20.736	50	20	5	0.597	0.565	21.759
10	20	10	0.532	0.532	20.752	50	20	10	0.516	0.516	21.786
10	20	20	0.565	0.452	20.795	50	20	20	0.548	0.581	21.828
20	2	1	0.548	0.548	18.513	100	2	1	0.548	0.548	18.985
20	2	3	0.516	0.597	18.516	100	2	3	0.500	0.516	18.992
20	2	5	0.565	0.548	18.523	100	2	5	0.468	0.500	19.001
20	2	10	0.516	0.516	18.554	100	2	10	0.468	0.548	19.023
20	2	20	0.613	0.532	18.585	100	2	20	0.468	0.516	19.066
20	5	1	0.565	0.565	19.491	100	5	1	0.548	0.548	20.595
20	5	3	0.516	0.532	19.496	100	5	3	0.532	0.548	20.602
20	5	5	0.532	0.516	19.508	100	5	5	0.516	0.532	20.615
20	5	10	0.500	0.532	19.523	100	5	10	0.548	0.532	20.639
20	5	20	0.516	0.516	19.577	100	5	20	0.532	0.548	20.683
20	10	1	0.548	0.548	20.307	100	10	1	0.581	0.581	21.904
20	10	3	0.516	0.532	20.300	100	10	3	0.581	0.565	21.913
20	10	5	0.532	0.548	20.310	100	10	5	0.581	0.565	21.920
20	10	10	0.516	0.548	20.334	100	10	10	0.565	0.532	21.947
20	10	20	0.468	0.516	20.383	100	10	20	0.516	0.516	21.996
20	20	1	0.548	0.548	21.000	100	20	1	0.565	0.565	23.273
20	20	3	0.516	0.532	21.008	100	20	3	0.581	0.565	23.280

Table 4.8 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	20	5	0.532	0.548	21.011	100	20	5	0.581	0.565	23.292
20	20	10	0.516	0.548	21.059	100	20	10	0.581	0.548	23.314
20	20	20	0.468	0.516	21.091	100	20	20	0.532	0.532	23.363

Table 4.9 and table 4.10 give accuracy ratios when support threshold is 0.8. The first and second columns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as multiplication.

Table 4.9 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.484	0.484	10.591	50	2	1	0.484	0.484	10.622
10	2	3	0.581	0.565	10.594	50	2	3	0.581	0.565	10.648
10	2	5	0.645	0.613	10.611	50	2	5	0.645	0.613	10.671
10	2	10	0.645	0.613	10.675	50	2	10	0.629	0.597	10.736
10	2	20	0.677	0.548	10.804	50	2	20	0.629	0.581	10.863
10	5	1	0.484	0.484	10.745	50	5	1	0.484	0.484	10.842
10	5	3	0.581	0.565	10.767	50	5	3	0.581	0.565	10.868
10	5	5	0.645	0.613	10.787	50	5	5	0.645	0.613	10.891
10	5	10	0.645	0.613	10.816	50	5	10	0.629	0.597	10.955
10	5	20	0.677	0.548	10.966	50	5	20	0.613	0.581	11.083
10	10	1	0.484	0.484	10.862	50	10	1	0.484	0.484	11.034
10	10	3	0.581	0.565	10.890	50	10	3	0.581	0.565	11.060
10	10	5	0.645	0.613	10.913	50	10	5	0.645	0.613	11.082
10	10	10	0.645	0.613	10.977	50	10	10	0.629	0.597	11.145
10	10	20	0.677	0.548	11.108	50	10	20	0.613	0.581	11.272
10	20	1	0.484	0.484	10.948	50	20	1	0.484	0.484	11.177
10	20	3	0.581	0.565	10.976	50	20	3	0.581	0.565	11.203
10	20	5	0.645	0.613	11.002	50	20	5	0.645	0.613	11.224
10	20	10	0.645	0.613	11.063	50	20	10	0.629	0.597	11.286
10	20	20	0.677	0.548	11.185	50	20	20	0.613	0.581	11.413
20	2	1	0.484	0.484	10.571	100	2	1	0.484	0.484	10.709
20	2	3	0.581	0.565	10.593	100	2	3	0.581	0.565	10.737
20	2	5	0.645	0.613	10.625	100	2	5	0.645	0.613	10.758
20	2	10	0.629	0.597	10.684	100	2	10	0.629	0.597	10.826
20	2	20	0.629	0.597	10.807	100	2	20	0.629	0.581	10.949

Table 4.10Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	5	1	0.484	0.484	10.758	100	5	1	0.484	0.484	10.978
20	5	3	0.581	0.565	10.779	100	5	3	0.581	0.565	11.006
20	5	5	0.645	0.613	10.810	100	5	5	0.645	0.613	11.028
20	5	10	0.629	0.597	10.874	100	5	10	0.629	0.597	11.093
20	5	20	0.613	0.581	10.998	100	5	20	0.613	0.581	11.226
20	10	1	0.484	0.484	10.912	100	10	1	0.484	0.484	11.230
20	10	3	0.581	0.565	10.935	100	10	3	0.581	0.565	11.257
20	10	5	0.645	0.613	10.962	100	10	5	0.645	0.613	11.275
20	10	10	0.629	0.597	11.027	100	10	10	0.629	0.597	11.341
20	10	20	0.613	0.581	11.151	100	10	20	0.613	0.581	11.465
20	20	1	0.484	0.484	11.008	100	20	1	0.484	0.484	11.430
20	20	3	0.581	0.565	11.033	100	20	3	0.581	0.565	11.456
20	20	5	0.645	0.613	11.058	100	20	5	0.645	0.613	11.484
20	20	10	0.629	0.597	11.118	100	20	10	0.629	0.597	11.547
20	20	20	0.613	0.581	11.247	100	20	20	0.613	0.581	11.672

The results show the lack of classification performance of this algorithm when compared to F-ARM. This outcome may be related to the non-linearity between average_support values in the algorithm with the real average support levels of itemsets. This will be treated in the discussion section. Another factor that leads to these accuracy results may be the high number of output rules that are discovered by the beam search algorithm. This situation will also be investigated in the discussions section.

Implementation on Leukemia Dataset

Fuzzy c-means clustering was applied to the leukemia dataset to create 2 fuzzy clusters from each gene. Beam search algorithm is run on the training dataset and memberships for each instance of the testing set are evaluated using the itemsets discovered by mining the training set. The parameter set for *beam_width* is {10, 20, 50, 100, 250, 350}, the parameter set for *child_width* is {2, 5, 10, 20, 50, 70}. The parameter set for the number of output rules to be chosen is (*number_of_output_rules*) {1, 3, 5, 10, 20, 50}.

Table 4.11, Table 4.12 and Table 4.13 give accuracy ratios when support threshold is 0.9. The first and second columns represent whether the maximum of the memberships or the

average of the memberships has been used for the computation of the final membership. ANDing operator is selected as minimum.

Table 4.11 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.559	0.559	66.203	100	2	1	0.412	0.412	68.468
10	2	3	0.559	0.559	65.875	100	2	3	0.412	0.412	68.578
10	2	5	0.559	0.559	66.094	100	2	5	0.412	0.412	68.657
10	2	10	0.588	0.588	66.437	100	2	10	0.412	0.588	68.859
10	2	20	0.588	0.588	66.391	100	2	20	0.412	0.588	69.250
10	2	50	0.588	0.588	67.547	100	2	50	0.588	0.588	70.453
10	5	1	0.529	0.529	73.125	100	5	1	0.412	0.412	77.297
10	5	3	0.529	0.529	73.578	100	5	3	0.412	0.412	77.375
10	5	5	0.529	0.559	73.703	100	5	5	0.412	0.412	77.469
10	5	10	0.588	0.588	73.094	100	5	10	0.412	0.412	77.703
10	5	20	0.588	0.588	72.593	100	5	20	0.412	0.412	78.234
10	5	50	0.588	0.588	73.016	100	5	50	0.412	0.412	79.594
10	10	1	0.529	0.529	78.547	100	10	1	0.412	0.412	86.250
10	10	3	0.529	0.529	78.547	100	10	3	0.412	0.412	86.375
10	10	5	0.529	0.559	78.656	100	10	5	0.412	0.412	86.453
10	10	10	0.588	0.588	79.062	100	10	10	0.412	0.412	86.703
10	10	20	0.588	0.588	79.079	100	10	20	0.412	0.412	93.922
10	10	50	0.588	0.588	79.343	100	10	50	0.412	0.412	89.188
10	20	1	0.529	0.529	84.953	100	20	1	0.412	0.412	96.718
10	20	3	0.529	0.529	84.969	100	20	3	0.412	0.412	96.766
10	20	5	0.529	0.559	85.031	100	20	5	0.412	0.412	96.891
10	20	10	0.588	0.588	85.125	100	20	10	0.412	0.412	97.109
10	20	20	0.588	0.588	85.266	100	20	20	0.412	0.412	97.516
10	20	50	0.588	0.588	85.625	100	20	50	0.412	0.412	99.015
10	50	1	0.529	0.529	88.406	100	50	1	0.412	0.412	108.110
10	50	3	0.529	0.529	88.438	100	50	3	0.412	0.412	108.250
10	50	5	0.529	0.559	88.437	100	50	5	0.412	0.412	108.343
10	50	10	0.588	0.588	88.547	100	50	10	0.412	0.412	108.579
10	50	20	0.588	0.588	88.703	100	50	20	0.412	0.412	109.093
10	50	50	0.588	0.588	89.078	100	50	50	0.412	0.412	110.453
10	70	1	0.529	0.529	86.782	100	70	1	0.412	0.412	109.766
10	70	3	0.529	0.529	86.656	100	70	3	0.412	0.412	109.953
10	70	5	0.529	0.559	86.687	100	70	5	0.412	0.412	110.016
10	70	10	0.588	0.588	86.782	100	70	10	0.412	0.412	110.219
10	70	20	0.588	0.588	86.953	100	70	20	0.412	0.412	110.750
10	70	50	0.588	0.588	87.375	100	70	50	0.412	0.412	112.218
20	2	1	0.559	0.559	66.031	250	2	1	0.412	0.412	72.688
20	2	3	0.559	0.559	66.078	250	2	3	0.412	0.412	72.797
20	2	5	0.559	0.559	66.141	250	2	5	0.588	0.588	72.890
20	2	10	0.441	0.588	66.265	250	2	10	0.588	0.588	73.078

Table 4.12 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	2	20	0.588	0.588	66.438	250	2	20	0.588	0.588	73.485
20	2	50	0.588	0.588	66.828	250	2	50	0.588	0.588	74.703
20	5	1	0.559	0.559	72.641	250	5	1	0.412	0.412	86.000
20	5	3	0.559	0.559	72.703	250	5	3	0.412	0.412	86.125
20	5	5	0.559	0.559	72.734	250	5	5	0.412	0.412	86.234
20	5	10	0.588	0.559	72.844	250	5	10	0.412	0.412	86.516
20	5	20	0.588	0.588	73.109	250	5	20	0.588	0.412	86.937
20	5	50	0.588	0.588	73.547	250	5	50	0.588	0.412	88.438
20	10	1	0.559	0.559	79.078	250	10	1	0.676	0.676	100.484
20	10	3	0.559	0.559	79.203	250	10	3	0.676	0.676	100.578
20	10	5	0.559	0.559	79.204	250	10	5	0.676	0.676	100.672
20	10	10	0.588	0.559	79.359	250	10	10	0.676	0.676	100.953
20	10	20	0.588	0.588	79.594	250	10	20	0.676	0.676	101.516
20	10	50	0.588	0.588	80.062	250	10	50	0.676	0.676	103.109
20	20	1	0.559	0.559	85.985	250	20	1	0.676	0.676	118.250
20	20	3	0.559	0.559	86.046	250	20	3	0.676	0.676	118.360
20	20	5	0.559	0.559	86.125	250	20	5	0.676	0.676	118.484
20	20	10	0.588	0.559	86.266	250	20	10	0.676	0.676	118.735
20	20	20	0.588	0.588	86.469	250	20	20	0.676	0.676	119.281
20	20	50	0.588	0.588	86.906	250	20	50	0.676	0.676	120.875
20	50	1	0.559	0.559	90.375	250	50	1	0.676	0.676	143.922
20	50	3	0.559	0.559	90.391	250	50	3	0.676	0.676	144.015
20	50	5	0.559	0.559	90.453	250	50	5	0.676	0.676	144.172
20	50	10	0.588	0.559	90.594	250	50	10	0.676	0.676	144.469
20	50	20	0.588	0.588	90.812	250	50	20	0.676	0.676	144.984
20	50	50	0.588	0.588	91.297	250	50	50	0.676	0.676	146.547
20	70	1	0.559	0.559	89.094	250	70	1	0.676	0.676	151.250
20	70	3	0.559	0.559	89.156	250	70	3	0.676	0.676	151.360
20	70	5	0.559	0.559	89.219	250	70	5	0.676	0.676	151.468
20	70	10	0.588	0.559	89.328	250	70	10	0.676	0.676	151.735
20	70	20	0.588	0.588	89.625	250	70	20	0.676	0.676	152.265
20	70	50	0.588	0.588	90.015	250	70	50	0.676	0.676	153.922
50	2	1	0.412	0.412	67.032	350	2	1	0.647	0.647	75.438
50	2	3	0.412	0.412	67.109	350	2	3	0.647	0.647	75.531
50	2	5	0.412	0.412	67.203	350	2	5	0.647	0.618	75.625
50	2	10	0.412	0.412	67.375	350	2	10	0.588	0.588	75.828
50	2	20	0.588	0.588	67.781	350	2	20	0.588	0.588	76.266
50	2	50	0.588	0.588	68.750	350	2	50	0.588	0.412	77.484
50	5	1	0.412	0.412	74.469	350	5	1	0.412	0.412	91.641
50	5	3	0.412	0.412	74.563	350	5	3	0.412	0.412	91.765
50	5	5	0.412	0.412	74.672	350	5	5	0.412	0.412	91.876
50	5	10	0.412	0.412	74.921	350	5	10	0.412	0.412	92.094
50	5	20	0.412	0.412	75.375	350	5	20	0.412	0.412	92.641

Table 4.13 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
50	5	50	0.412	0.412	76.672	350	5	50	0.412	0.412	94.124
50	10	1	0.412	0.412	81.703	350	10	1	0.676	0.676	109.750
50	10	3	0.412	0.412	81.813	350	10	3	0.676	0.676	109.843
50	10	5	0.412	0.412	81.922	350	10	5	0.676	0.676	110.001
50	10	10	0.412	0.412	82.156	350	10	10	0.676	0.676	110.267
50	10	20	0.412	0.412	82.672	350	10	20	0.676	0.676	110.780
50	10	50	0.412	0.412	83.891	350	10	50	0.676	0.676	112.391
50	20	1	0.412	0.412	89.984	350	20	1	0.676	0.676	132.327
50	20	3	0.412	0.412	90.094	350	20	3	0.676	0.676	132.421
50	20	5	0.412	0.412	90.203	350	20	5	0.676	0.676	132.563
50	20	10	0.412	0.412	90.422	350	20	10	0.676	0.676	132.796
50	20	20	0.412	0.412	90.890	350	20	20	0.676	0.676	133.407
50	20	50	0.412	0.412	92.078	350	20	50	0.676	0.676	134.999
50	50	1	0.412	0.412	97.032	350	50	1	0.676	0.676	167.280
50	50	3	0.412	0.412	97.109	350	50	3	0.676	0.676	167.437
50	50	5	0.412	0.412	97.250	350	50	5	0.676	0.676	167.548
50	50	10	0.412	0.412	97.469	350	50	10	0.676	0.676	167.781
50	50	20	0.412	0.412	97.969	350	50	20	0.676	0.676	168.376
50	50	50	0.412	0.412	99.250	350	50	50	0.676	0.676	169.969
50	70	1	0.412	0.412	96.921	350	70	1	0.676	0.676	178.314
50	70	3	0.412	0.412	97.016	350	70	3	0.676	0.676	178.437
50	70	5	0.412	0.412	97.094	350	70	5	0.676	0.676	178.499
50	70	10	0.412	0.412	97.344	350	70	10	0.676	0.676	178.843
50	70	20	0.412	0.412	97.781	350	70	20	0.676	0.676	179.296
50	70	50	0.412	0.412	99.094	350	70	50	0.676	0.676	180.890

Table 4.14, Table 4.15 and table 4.16 give accuracy ratios when support threshold is 0.9. The first and second columns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as multiplication.

Table 4.14 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.588	0.588	48.239	100	2	1	0.588235	0.588235	49
10	2	3	0.588	0.588	48.287	100	2	3	0.588235	0.588235	48.938
10	2	5	0.588	0.588	48.302	100	2	5	0.588235	0.588235	49.187
10	2	10	0.588	0.588	48.474	100	2	10	0.588235	0.588235	49.375
10	2	20	0.588	0.588	48.958	100	2	20	0.588235	0.588235	49.782
10	2	50	0.588	0.588	50.115	100	2	50	0.588235	0.588235	50.906
10	5	1	0.588	0.588	51.521	100	5	1	0.588235	0.588235	53.406
10	5	3	0.588	0.588	51.615	100	5	3	0.588235	0.588235	53.516
10	5	5	0.588	0.588	51.756	100	5	5	0.588235	0.588235	53.547
10	5	10	0.588	0.588	52.021	100	5	10	0.588235	0.588235	53.843
10	5	20	0.588	0.588	52.412	100	5	20	0.588235	0.588235	54.11
10	5	50	0.588	0.588	53.459	100	5	50	0.588235	0.588235	55.703
10	10	1	0.588	0.588	54.756	100	10	1	0.588235	0.588235	58.094
10	10	3	0.588	0.588	54.787	100	10	3	0.588235	0.588235	58.14
10	10	5	0.588	0.588	54.944	100	10	5	0.588235	0.588235	58.172
10	10	10	0.588	0.588	55.131	100	10	10	0.588235	0.588235	58.391
10	10	20	0.588	0.588	55.584	100	10	20	0.588235	0.588235	58.781
10	10	50	0.588	0.588	56.615	100	10	50	0.588235	0.588235	60.078
10	20	1	0.588	0.588	57.929	100	20	1	0.382353	0.382353	63.516
10	20	3	0.588	0.588	58.006	100	20	3	0.411765	0.382353	63.547
10	20	5	0.588	0.588	58.116	100	20	5	0.411765	0.588235	63.656
10	20	10	0.588	0.588	58.303	100	20	10	0.411765	0.411765	63.953
10	20	20	0.588	0.588	58.741	100	20	20	0.588235	0.588235	64.406
10	20	50	0.588	0.588	59.912	100	20	50	0.588235	0.588235	65.828
10	50	1	0.588	0.588	59.382	100	50	1	0.382353	0.382353	69.141
10	50	3	0.588	0.588	59.366	100	50	3	0.411765	0.411765	69.281
10	50	5	0.588	0.588	59.397	100	50	5	0.411765	0.411765	69.422
10	50	10	0.588	0.588	59.756	100	50	10	0.411765	0.411765	69.641
10	50	20	0.588	0.588	60.163	100	50	20	0.588235	0.588235	70.094
10	50	50	0.588	0.588	61.335	100	50	50	0.588235	0.588235	71.562
10	70	1	0.588	0.588	58.459	100	70	1	0.382353	0.382353	69.828
10	70	3	0.588	0.588	58.554	100	70	3	0.411765	0.411765	70
10	70	5	0.588	0.588	58.631	100	70	5	0.411765	0.411765	70.156
10	70	10	0.588	0.588	58.881	100	70	10	0.411765	0.411765	70.391
10	70	20	0.588	0.588	59.288	100	70	20	0.588235	0.588235	70.859
10	70	50	0.588	0.588	60.429	100	70	50	0.588235	0.588235	72.329
20	2	1	0.588	0.588	48.297	250	2	1	0.588235	0.588235	50.328
20	2	3	0.588	0.588	48.219	250	2	3	0.588235	0.588235	50.406
20	2	5	0.588	0.588	48.453	250	2	5	0.588235	0.588235	50.375
20	2	10	0.588	0.588	48.64	250	2	10	0.588235	0.588235	50.687
20	2	20	0.588	0.588	49.032	250	2	20	0.588235	0.588235	51.11
20	2	50	0.588	0.588	49.89	250	2	50	0.588235	0.588235	52.25
20	5	1	0.588	0.588	51.781	250	5	1	0.588235	0.588235	56.5
20	5	3	0.588	0.588	51.906	250	5	3	0.588235	0.588235	56.437
20	5	5	0.588	0.588	51.969	250	5	5	0.588235	0.588235	56.657

Table 4.15 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	5	10	0.588	0.588	52.094	250	5	10	0.588235	0.588235	56.89
20	5	20	0.588	0.588	52.547	250	5	20	0.588235	0.588235	57.297
20	5	50	0.588	0.588	53.781	250	5	50	0.588235	0.588235	58.609
20	10	1	0.588	0.588	55.109	250	10	1	0.588235	0.588235	63.36
20	10	3	0.588	0.588	55.204	250	10	3	0.588235	0.588235	63.406
20	10	5	0.588	0.588	55.281	250	10	5	0.588235	0.588235	63.547
20	10	10	0.588	0.588	55.375	250	10	10	0.588235	0.588235	71.154
20	10	20	0.588	0.588	55.906	250	10	20	0.588235	0.588235	64.257
20	10	50	0.588	0.588	57	250	10	50	0.588235	0.588235	65.507
20	20	1	0.588	0.588	58.516	250	20	1	0.382353	0.382353	72.961
20	20	3	0.588	0.588	58.593	250	20	3	0.411765	0.441176	73.086
20	20	5	0.588	0.588	58.672	250	20	5	0.411765	0.588235	73.196
20	20	10	0.588	0.588	58.875	250	20	10	0.411765	0.470588	73.477
20	20	20	0.588	0.588	59.36	250	20	20	0.588235	0.588235	73.93
20	20	50	0.588	0.588	60.375	250	20	50	0.588235	0.588235	75.414
20	50	1	0.588	0.588	60.406	250	50	1	0.382353	0.382353	86.025
20	50	3	0.588	0.588	60.484	250	50	3	0.411765	0.441176	85.994
20	50	5	0.588	0.588	60.532	250	50	5	0.411765	0.441176	86.087
20	50	10	0.588	0.588	60.796	250	50	10	0.411765	0.529412	86.447
20	50	20	0.588	0.588	61.235	250	50	20	0.588235	0.588235	86.9
20	50	50	0.588	0.588	62.297	250	50	50	0.588235	0.588235	88.322
20	70	1	0.588	0.588	59.703	250	70	1	0.382353	0.382353	89.494
20	70	3	0.588	0.588	59.719	250	70	3	0.411765	0.441176	89.525
20	70	5	0.588	0.588	59.875	250	70	5	0.411765	0.441176	89.604
20	70	10	0.588	0.588	60.125	250	70	10	0.411765	0.588235	89.994
20	70	20	0.588	0.588	60.453	250	70	20	0.588235	0.588235	90.338
20	70	50	0.588	0.588	61.578	250	70	50	0.588235	0.588235	91.822
50	2	1	0.588	0.588	48.531	350	2	1	0.588235	0.588235	51.225
50	2	3	0.588	0.588	48.625	350	2	3	0.588235	0.588235	51.287
50	2	5	0.588	0.588	48.688	350	2	5	0.588235	0.588235	51.349
50	2	10	0.588	0.588	48.765	350	2	10	0.588235	0.588235	51.412
50	2	20	0.588	0.588	49.328	350	2	20	0.588235	0.588235	51.943
50	2	50	0.588	0.588	50.438	350	2	50	0.588235	0.588235	53.099
50	5	1	0.588	0.588	52.375	350	5	1	0.588235	0.588235	58.444
50	5	3	0.588	0.588	52.312	350	5	3	0.588235	0.588235	58.554
50	5	5	0.588	0.588	52.547	350	5	5	0.588235	0.588235	58.647
50	5	10	0.588	0.588	52.781	350	5	10	0.588235	0.588235	58.834
50	5	20	0.588	0.588	53.25	350	5	20	0.588235	0.588235	59.225
50	5	50	0.588	0.588	54.547	350	5	50	0.588235	0.588235	60.616
50	10	1	0.588	0.588	56.063	350	10	1	0.588235	0.588235	66.851
50	10	3	0.588	0.588	56.125	350	10	3	0.588235	0.588235	66.992
50	10	5	0.588	0.588	56.344	350	10	5	0.588235	0.588235	67.085
50	10	10	0.588	0.588	56.578	350	10	10	0.588235	0.588235	67.305
50	10	20	0.588	0.588	57.015	350	10	20	0.588235	0.588235	67.71
50	10	50	0.588	0.588	58.422	350	10	50	0.588235	0.588235	69.101

Table 4.16 Results evaluated with different parameter settings

Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9						Leukemia data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.9					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
50	20	1	0.588	0.588	60.328	350	20	1	0.382353	0.382353	79.259
50	20	3	0.588	0.588	60.391	350	20	3	0.411765	0.441176	79.227
50	20	5	0.588	0.588	60.469	350	20	5	0.588235	0.588235	79.337
50	20	10	0.588	0.588	60.64	350	20	10	0.588235	0.588235	79.634
50	20	20	0.588	0.588	61.157	350	20	20	0.588235	0.588235	80.133
50	20	50	0.588	0.588	62.562	350	20	50	0.588235	0.588235	81.806
50	50	1	0.588	0.588	63.609	350	50	1	0.382353	0.382353	96.87
50	50	3	0.588	0.588	63.735	350	50	3	0.411765	0.441176	97.151
50	50	5	0.588	0.588	63.812	350	50	5	0.588235	0.588235	97.073
50	50	10	0.588	0.588	64.047	350	50	10	0.588235	0.588235	97.402
50	50	20	0.588	0.588	64.5	350	50	20	0.588235	0.588235	97.917
50	50	50	0.588	0.588	65.891	350	50	50	0.588235	0.588235	99.417
50	70	1	0.588	0.588	63.469	350	70	1	0.382353	0.382353	102.308
50	70	3	0.588	0.588	63.578	350	70	3	0.411765	0.441176	102.495
50	70	5	0.588	0.588	63.672	350	70	5	0.588235	0.588235	102.418
50	70	10	0.588	0.588	63.921	350	70	10	0.588235	0.588235	102.87
50	70	20	0.588	0.588	64.36	350	70	20	0.588235	0.588235	103.214
50	70	50	0.588	0.588	65.765	350	70	50	0.588235	0.588235	104.918

Table 4.17, Table 4.18, Table 4.19, Table 4.20 and Table 4.21 give accuracy ratios when support threshold is 0.8. The first and second columns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as minimum.

Table 4.17 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.588	0.588	1114.6	100	2	1	0.588	0.588	1125.2
10	2	3	0.588	0.588	1113.7	100	2	3	0.588	0.588	1125.7
10	2	5	0.588	0.588	1115.1	100	2	5	0.588	0.588	1125.3
10	2	10	0.588	0.588	1116.5	100	2	10	0.588	0.588	1125.8
10	2	20	0.588	0.588	1114.5	100	2	20	0.588	0.588	1126.4
10	2	50	0.588	0.588	1114.9	100	2	50	0.588	0.588	1127.6
10	5	1	0.588	0.588	1187.7	100	5	1	0.588	0.588	1215.2
10	5	3	0.588	0.588	1188.1	100	5	3	0.588	0.588	1215.7
10	5	5	0.588	0.588	1187.7	100	5	5	0.588	0.588	1215.4

Table 4.18 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	5	10	0.588	0.588	1188.0	100	5	10	0.588	0.588	1215.9
10	5	20	0.588	0.588	1189.3	100	5	20	0.588	0.588	1217.2
10	5	50	0.588	0.588	1189.7	100	5	50	0.588	0.588	1219.3
10	10	1	0.588	0.588	1277.8	100	10	1	0.588	0.588	1325.4
10	10	3	0.588	0.588	1280.5	100	10	3	0.588	0.588	1325.4
10	10	5	0.588	0.588	1290.3	100	10	5	0.588	0.588	1325.9
10	10	10	0.588	0.588	1278.9	100	10	10	0.588	0.588	1326.2
10	10	20	0.588	0.588	1281.1	100	10	20	0.588	0.588	1326.6
10	10	50	0.588	0.588	1279.9	100	10	50	0.588	0.588	1329.8
10	20	1	0.588	0.588	1402.8	100	20	1	0.588	0.588	1482.1
10	20	3	0.588	0.588	1402.3	100	20	3	0.588	0.588	1481.8
10	20	5	0.588	0.588	1402.4	100	20	5	0.588	0.588	1471.94
10	20	10	0.588	0.588	1402.8	100	20	10	0.588	0.588	1472.64
10	20	20	0.588	0.588	1404.0	100	20	20	0.588	0.588	1473.52
10	20	50	0.588	0.588	1404.5	100	20	50	0.588	0.588	1476.08
10	50	1	0.588	0.588	1598.7	100	50	1	0.588	0.588	1754.19
10	50	3	0.588	0.588	1597.6	100	50	3	0.588	0.588	1752.23
10	50	5	0.588	0.588	1598.1	100	50	5	0.588	0.588	1751.92
10	50	10	0.588	0.588	1598.6	100	50	10	0.588	0.588	1754.36
10	50	20	0.588	0.588	1599.4	100	50	20	0.588	0.588	1754.77
10	50	50	0.588	0.588	1599.7	100	50	50	0.588	0.588	1756.16
10	70	1	0.588	0.588	1660.6	100	70	1	0.588	0.588	1860.8
10	70	3	0.588	0.588	1660.0	100	70	3	0.588	0.588	1860.02
10	70	5	0.588	0.588	1660.0	100	70	5	0.588	0.588	1860.48
10	70	10	0.588	0.588	1660.3	100	70	10	0.588	0.588	1860.85
10	70	20	0.588	0.588	1660.8	100	70	20	0.588	0.588	1861.28
10	70	50	0.588	0.588	1662.2	100	70	50	0.588	0.588	1864.34
20	2	1	0.588	0.588	1115.6	250	2	1	0.588	0.588	1132.28
20	2	3	0.588	0.588	1115.2	250	2	3	0.588	0.588	1133.89
20	2	5	0.588	0.588	1115.0	250	2	5	0.588	0.588	1133.11
20	2	10	0.588	0.588	1116.0	250	2	10	0.588	0.588	1133.27
20	2	20	0.588	0.588	1116.6	250	2	20	0.588	0.588	1135.52
20	2	50	0.588	0.588	1117.6	250	2	50	0.588	0.588	1137.42
20	5	1	0.588	0.588	1189.4	250	5	1	0.588	0.588	1249.23
20	5	3	0.588	0.588	1190.2	250	5	3	0.588	0.588	1249.39
20	5	5	0.588	0.588	1190.2	250	5	5	0.588	0.588	1248.09
20	5	10	0.588	0.588	1190.5	250	5	10	0.588	0.588	1247.81
20	5	20	0.588	0.588	1191.0	250	5	20	0.588	0.588	1248.58
20	5	50	0.588	0.588	1192.3	250	5	50	0.588	0.588	1250.69
20	10	1	0.588	0.588	1282.1	250	10	1	0.588	0.588	1391.5
20	10	3	0.588	0.588	1282.8	250	10	3	0.588	0.588	1391.59
20	10	5	0.588	0.588	1282.8	250	10	5	0.588	0.588	1391.66
20	10	10	0.588	0.588	1282.9	250	10	10	0.588	0.588	1391.76
20	10	20	0.588	0.588	1283.7	250	10	20	0.588	0.588	1392.47

Table 4.19 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	10	50	0.588	0.588	1285.4	250	10	50	0.588	0.588	1395.3
20	20	1	0.588	0.588	1410.9	250	20	1	0.588	0.588	1609.09
20	20	3	0.588	0.588	1410.9	250	20	3	0.588	0.588	1607.2
20	20	5	0.588	0.588	1411.1	250	20	5	0.588	0.588	1607.26
20	20	10	0.588	0.588	1411.4	250	20	10	0.588	0.588	1608.09
20	20	20	0.588	0.588	1412.6	250	20	20	0.588	0.588	1608.48
20	20	50	0.588	0.588	1413.5	250	20	50	0.588	0.588	1610.5
20	50	1	0.588	0.588	1615.5	250	50	1	0.588	0.588	2032.84
20	50	3	0.588	0.588	1616.1	250	50	3	0.588	0.588	2031.86
20	50	5	0.588	0.588	1616.8	250	50	5	0.588	0.588	2031.62
20	50	10	0.588	0.588	1616.9	250	50	10	0.588	0.588	2032.22
20	50	20	0.588	0.588	1617.3	250	50	20	0.588	0.588	2032.77
20	50	50	0.588	0.588	1618.8	250	50	50	0.588	0.588	2035.23
20	70	1	0.588	0.588	1683.5	250	70	1	0.588	0.588	2216.41
20	70	3	0.588	0.588	1683.3	250	70	3	0.588	0.588	2216.08
20	70	5	0.588	0.588	1683.8	250	70	5	0.588	0.588	2222.03
20	70	10	0.588	0.588	1684.5	250	70	10	0.588	0.588	2220.81
20	70	20	0.588	0.588	1684.4	250	70	20	0.588	0.588	2222.91
20	70	50	0.588	0.588	1685.9	250	70	50	0.588	0.588	2225.69
50	2	1	0.588	0.588	1118.8	350	2	1	0.588	0.588	1146.11
50	2	3	0.588	0.588	1118.6	350	2	3	0.588	0.588	1146.39
50	2	5	0.588	0.588	1118.6	350	2	5	0.588	0.588	1146.3
50	2	10	0.588	0.588	1118.9	350	2	10	0.588	0.588	1147.3
50	2	20	0.588	0.588	1119.1	350	2	20	0.588	0.588	1148.05
50	2	50	0.588	0.588	1120.9	350	2	50	0.588	0.588	1149.2
50	5	1	0.588	0.588	1198.4	350	5	1	0.588	0.588	1277.55
50	5	3	0.588	0.588	1198.9	350	5	3	0.588	0.588	1279.28
50	5	5	0.588	0.588	1198.8	350	5	5	0.588	0.588	1279.5
50	5	10	0.588	0.588	1198.8	350	5	10	0.588	0.588	1279.35
50	5	20	0.588	0.588	1199.6	350	5	20	0.588	0.588	1280.49
50	5	50	0.588	0.588	1202.2	350	5	50	0.588	0.588	1282.36
50	10	1	0.588	0.588	1298.5	350	10	1	0.588	0.588	1444.94
50	10	3	0.588	0.588	1298.3	350	10	3	0.588	0.588	1446.25
50	10	5	0.588	0.588	1298.7	350	10	5	0.588	0.588	1446.27
50	10	10	0.588	0.588	1299.2	350	10	10	0.588	0.588	1446.93
50	10	20	0.588	0.588	1300.4	350	10	20	0.588	0.588	1447.14
50	10	50	0.588	0.588	1302.5	350	10	50	0.588	0.588	1449.7
50	20	1	0.588	0.588	1437.1	350	20	1	0.588	0.588	1703.88
50	20	3	0.588	0.588	1437.5	350	20	3	0.588	0.588	1703.22
50	20	5	0.588	0.588	1437.0	350	20	5	0.588	0.588	1703.62
50	20	10	0.588	0.588	1437.9	350	20	10	0.588	0.588	1703.91
50	20	20	0.588	0.588	1438.3	350	20	20	0.588	0.588	1705.49
50	20	50	0.588	0.588	1440.9	350	20	50	0.588	0.588	1707.86

Table 4.20 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
50	50	1	0.588	0.588	1670.7	350	50	1	0.588	0.588	2223.8
50	50	3	0.588	0.588	1671.0	350	50	3	0.588	0.588	2223.55
50	50	5	0.588	0.588	1671.5	350	50	5	0.588	0.588	2228.53
50	50	10	0.588	0.588	1671.3	350	50	10	0.588	0.588	2228.07
50	50	20	0.588	0.588	1672.8	350	50	20	0.588	0.588	2230.68
50	50	50	0.588	0.588	1674.9	350	50	50	0.588	0.588	2231.93
50	70	1	0.588	0.588	1753.4	350	70	1	0.588	0.588	2468.5
50	70	3	0.588	0.588	1753.6	350	70	3	0.588	0.588	2464.09
50	70	5	0.588	0.588	1754.6	350	70	5	0.588	0.588	2458.08
50	70	10	0.588	0.588	1754.7	350	70	10	0.588	0.588	2459.99
50	70	20	0.588	0.588	1755.1	350	70	20	0.588	0.588	2458.59
50	70	50	0.588	0.588	1757.8	350	70	50	0.588	0.588	2462.14
20	50	20	0.588	0.588	1617.3	250	50	20	0.588	0.588	2032.77
20	50	50	0.588	0.588	1618.8	250	50	50	0.588	0.588	2035.23
20	70	1	0.588	0.588	1683.5	250	70	1	0.588	0.588	2216.41
20	70	3	0.588	0.588	1683.3	250	70	3	0.588	0.588	2216.08
20	70	5	0.588	0.588	1683.8	250	70	5	0.588	0.588	2222.03
20	70	10	0.588	0.588	1684.5	250	70	10	0.588	0.588	2220.81
20	70	20	0.588	0.588	1684.4	250	70	20	0.588	0.588	2222.91
20	70	50	0.588	0.588	1685.9	250	70	50	0.588	0.588	2225.69
50	2	1	0.588	0.588	1118.8	350	2	1	0.588	0.588	1146.11
50	2	3	0.588	0.588	1118.6	350	2	3	0.588	0.588	1146.39
50	2	5	0.588	0.588	1118.6	350	2	5	0.588	0.588	1146.3
50	2	10	0.588	0.588	1118.9	350	2	10	0.588	0.588	1147.3
50	2	20	0.588	0.588	1119.1	350	2	20	0.588	0.588	1148.05
50	2	50	0.588	0.588	1120.9	350	2	50	0.588	0.588	1149.2
50	5	1	0.588	0.588	1198.4	350	5	1	0.588	0.588	1277.55
50	5	3	0.588	0.588	1198.9	350	5	3	0.588	0.588	1279.28
50	5	5	0.588	0.588	1198.8	350	5	5	0.588	0.588	1279.5
50	5	10	0.588	0.588	1198.8	350	5	10	0.588	0.588	1279.35
50	5	20	0.588	0.588	1199.6	350	5	20	0.588	0.588	1280.49
50	5	50	0.588	0.588	1202.2	350	5	50	0.588	0.588	1282.36
50	10	1	0.588	0.588	1298.5	350	10	1	0.588	0.588	1444.94
50	10	3	0.588	0.588	1298.3	350	10	3	0.588	0.588	1446.25
50	10	5	0.588	0.588	1298.7	350	10	5	0.588	0.588	1446.27
50	10	10	0.588	0.588	1299.2	350	10	10	0.588	0.588	1446.93
50	10	20	0.588	0.588	1300.4	350	10	20	0.588	0.588	1447.14
50	10	50	0.588	0.588	1302.5	350	10	50	0.588	0.588	1449.7
50	20	1	0.588	0.588	1437.1	350	20	1	0.588	0.588	1703.88
50	20	3	0.588	0.588	1437.5	350	20	3	0.588	0.588	1703.22
50	20	5	0.588	0.588	1437.0	350	20	5	0.588	0.588	1703.62
50	20	10	0.588	0.588	1437.9	350	20	10	0.588	0.588	1703.91
50	20	20	0.588	0.588	1438.3	350	20	20	0.588	0.588	1705.49
50	20	50	0.588	0.588	1440.9	350	20	50	0.588	0.588	1707.86

Table 4.21 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=min, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
50	50	1	0.588	0.588	1670.7	350	50	1	0.588	0.588	2223.8
50	50	3	0.588	0.588	1671.0	350	50	3	0.588	0.588	2223.55
50	50	5	0.588	0.588	1671.5	350	50	5	0.588	0.588	2228.53
50	50	10	0.588	0.588	1671.3	350	50	10	0.588	0.588	2228.07
50	50	20	0.588	0.588	1672.8	350	50	20	0.588	0.588	2230.68
50	50	50	0.588	0.588	1674.9	350	50	50	0.588	0.588	2231.93
50	70	1	0.588	0.588	1753.4	350	70	1	0.588	0.588	2468.5
50	70	3	0.588	0.588	1753.6	350	70	3	0.588	0.588	2464.09
50	70	5	0.588	0.588	1754.6	350	70	5	0.588	0.588	2458.08
50	70	10	0.588	0.588	1754.7	350	70	10	0.588	0.588	2459.99
50	70	20	0.588	0.588	1755.1	350	70	20	0.588	0.588	2458.59
50	70	50	0.588	0.588	1757.8	350	70	50	0.588	0.588	2462.14

Table 4.22, Table 4.23, Table 4.24 and Table 4.25 give accuracy ratios when support threshold is 0.8. The first and second columns represent whether the maximum of the memberships or the average of the memberships has been used for the computation of the final membership. ANDing operator is selected as the product.

Table 4.22 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	2	1	0.588	0.588	1019.58	100	2	1	0.588	0.588	900.749
10	2	3	0.588	0.588	1014.73	100	2	3	0.588	0.588	900.991
10	2	5	0.588	0.588	1016.71	100	2	5	0.588	0.588	900.852
10	2	10	0.588	0.588	1030.51	100	2	10	0.588	0.588	901.341
10	2	20	0.588	0.588	980.078	100	2	20	0.588	0.588	902.597
10	2	50	0.588	0.588	1008.33	100	2	50	0.588	0.588	906.646
10	5	1	0.588	0.588	1149.06	100	5	1	0.412	0.412	955.287
10	5	3	0.588	0.588	1576.66	100	5	3	0.412	0.588	954.353
10	5	5	0.588	0.588	997.095	100	5	5	0.412	0.588	954.893
10	5	10	0.588	0.588	935.969	100	5	10	0.412	0.588	953.874
10	5	20	0.588	0.588	936.905	100	5	20	0.412	0.588	957.359
10	5	50	0.588	0.588	939.634	100	5	50	0.412	0.588	959.679
10	10	1	0.588	0.588	987.465	100	10	1	0.588	0.588	1020.36
10	10	3	0.588	0.588	987.152	100	10	3	0.588	0.588	1021.39
10	10	5	0.588	0.588	987.402	100	10	5	0.588	0.588	1020.47
10	10	10	0.588	0.588	989.103	100	10	10	0.588	0.588	1373.15

Table 4.23 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
10	10	20	0.588	0.588	1048.83	100	10	20	0.588	0.588	1514.03
10	10	50	0.588	0.588	1063.63	100	10	50	0.588	0.588	1518.75
10	20	1	0.588	0.588	1137.77	100	20	1	0.588	0.588	1648.98
10	20	3	0.588	0.588	1129.81	100	20	3	0.588	0.588	1649.53
10	20	5	0.588	0.588	1122.89	100	20	5	0.588	0.588	1651.55
10	20	10	0.588	0.588	1112.31	100	20	10	0.588	0.588	1605.7
10	20	20	0.588	0.588	1118.94	100	20	20	0.588	0.588	1267.15
10	20	50	0.588	0.588	1106.43	100	20	50	0.588	0.588	1246.69
10	50	1	0.588	0.588	1247.81	100	50	1	0.588	0.588	1443.28
10	50	3	0.588	0.588	1227.6	100	50	3	0.588	0.588	1475.15
10	50	5	0.588	0.588	1178.81	100	50	5	0.588	0.588	1488.09
10	50	10	0.588	0.588	1166.87	100	50	10	0.588	0.588	1378.78
10	50	20	0.588	0.588	1168.8	100	50	20	0.588	0.588	1392.28
10	50	50	0.588	0.588	1171.55	100	50	50	0.588	0.588	1379.1
10	70	1	0.588	0.588	1198.31	100	70	1	0.588	0.588	1427.57
10	70	3	0.588	0.588	1198.89	100	70	3	0.588	0.588	1424.28
10	70	5	0.588	0.588	1198.93	100	70	5	0.588	0.588	1422.19
10	70	10	0.588	0.588	1199.25	100	70	10	0.588	0.588	1426.07
10	70	20	0.588	0.588	1202.01	100	70	20	0.588	0.588	1422.98
10	70	50	0.588	0.588	1204.24	100	70	50	0.588	0.588	1428.5
20	2	1	0.588	0.588	889.264	250	2	1	0.588	0.588	967.887
20	2	3	0.588	0.588	893.35	250	2	3	0.588	0.588	968.236
20	2	5	0.588	0.588	899.929	250	2	5	0.588	0.588	967.655
20	2	10	0.588	0.588	914.414	250	2	10	0.588	0.588	977.662
20	2	20	0.588	0.588	893.285	250	2	20	0.588	0.588	976.752
20	2	50	0.588	0.588	896.72	250	2	50	0.588	0.588	980.869
20	5	1	0.588	0.588	944.039	250	5	1	0.412	0.412	1035.7
20	5	3	0.588	0.588	1036.96	250	5	3	0.412	0.588	1026.77
20	5	5	0.588	0.588	1070.35	250	5	5	0.412	0.588	1037.44
20	5	10	0.588	0.588	1035.14	250	5	10	0.412	0.588	1025.02
20	5	20	0.588	0.588	1032.35	250	5	20	0.412	0.588	1043.44
20	5	50	0.588	0.588	1043.49	250	5	50	0.412	0.588	1047.97
20	10	1	0.588	0.588	1098.97	250	10	1	0.412	0.412	1149.69
20	10	3	0.588	0.588	1109.14	250	10	3	0.412	0.412	1085.72
20	10	5	0.588	0.588	1045.49	250	10	5	0.412	0.588	1130.37
20	10	10	0.588	0.588	1073.69	250	10	10	0.412	0.588	1092.91
20	10	20	0.588	0.588	1309.55	250	10	20	0.412	0.588	1078.91
20	10	50	0.588	0.588	1081.68	250	10	50	0.412	0.588	1130.13
20	20	1	0.588	0.588	1162.33	250	20	1	0.412	0.412	1306.31
20	20	3	0.588	0.588	1280.4	250	20	3	0.412	0.588	1300.55
20	20	5	0.588	0.588	1580.53	250	20	5	0.412	0.588	1203.08
20	20	10	0.588	0.588	1580.69	250	20	10	0.412	0.588	1197.54
20	20	20	0.588	0.588	1580.71	250	20	20	0.412	0.588	1198.15

Table 4.24 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
20	20	50	0.588	0.588	1588.33	250	20	50	0.412	0.588	1200.8
20	50	1	0.588	0.588	1151.66	250	50	1	0.412	0.412	1454.31
20	50	3	0.588	0.588	1169.08	250	50	3	0.412	0.588	1446.17
20	50	5	0.588	0.588	1168.98	250	50	5	0.412	0.588	1446.44
20	50	10	0.588	0.588	1171.08	250	50	10	0.412	0.588	1450.03
20	50	20	0.588	0.588	1171.87	250	50	20	0.412	0.588	1447.91
20	50	50	0.588	0.588	1175.85	250	50	50	0.412	0.588	1452.15
20	70	1	0.588	0.588	1207.91	250	70	1	0.412	0.412	1555.07
20	70	3	0.588	0.588	1204.1	250	70	3	0.412	0.588	1551.39
20	70	5	0.588	0.588	1203.73	250	70	5	0.412	0.588	1551.12
20	70	10	0.588	0.588	1739.52	250	70	10	0.412	0.588	1552.09
20	70	20	0.588	0.588	1799.24	250	70	20	0.412	0.588	1553.8
20	70	50	0.588	0.588	1805.07	250	70	50	0.412	0.588	1558
50	2	1	0.588	0.588	1320.73	350	2	1	0.588	0.588	914.986
50	2	3	0.588	0.588	1321.04	350	2	3	0.588	0.588	914.95
50	2	5	0.588	0.588	1321.26	350	2	5	0.588	0.588	914.322
50	2	10	0.588	0.588	1320.2	350	2	10	0.588	0.588	915.722
50	2	20	0.588	0.588	1104.62	350	2	20	0.588	0.588	920.722
50	2	50	0.588	0.588	1143.77	350	2	50	0.588	0.588	920.557
50	5	1	0.588	0.588	1094.44	350	5	1	0.412	0.412	990.91
50	5	3	0.588	0.588	1136.98	350	5	3	0.412	0.588	991.035
50	5	5	0.588	0.588	989.019	350	5	5	0.412	0.588	991.749
50	5	10	0.588	0.588	985.658	350	5	10	0.412	0.588	992.196
50	5	20	0.588	0.588	1070.36	350	5	20	0.412	0.588	999.824
50	5	50	0.588	0.588	1218.65	350	5	50	0.412	0.588	1430.77
50	10	1	0.588	0.588	1313.07	350	10	1	0.412	0.412	1618.44
50	10	3	0.588	0.588	1212.98	350	10	3	0.412	0.412	1618.34
50	10	5	0.588	0.588	1306.12	350	10	5	0.412	0.588	1617.81
50	10	10	0.588	0.588	1016.53	350	10	10	0.412	0.588	1626.02
50	10	20	0.588	0.588	1110.07	350	10	20	0.412	0.588	1153.89
50	10	50	0.588	0.588	1176.27	350	10	50	0.412	0.588	1144.72
50	20	1	0.588	0.588	1618.45	350	20	1	0.412	0.412	1315.48
50	20	3	0.588	0.588	1612.95	350	20	3	0.412	0.412	1433.31
50	20	5	0.588	0.588	1678.43	350	20	5	0.412	0.588	1381.07
50	20	10	0.588	0.588	1709.17	350	20	10	0.412	0.588	1347.77
50	20	20	0.588	0.588	1530.54	350	20	20	0.412	0.588	1329.01
50	20	50	0.588	0.588	1201.44	350	20	50	0.412	0.588	1319.24
50	50	1	0.588	0.588	1301.16	350	50	1	0.412	0.412	1669.89
50	50	3	0.588	0.588	1264.84	350	50	3	0.412	0.412	1722.66
50	50	5	0.588	0.588	1271.56	350	50	5	0.412	0.412	1868.26

Table 4.25 Results evaluated with different parameter settings

colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8						colon cancer data, fuzzification parameter=2.8, ANDing operator=product, support threshold=0.8					
beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)	beam width	child width	number of output rules	accuracy when max is used	accuracy when average is used	time (sec)
50	50	10	0.588	0.588	1336.8	350	50	10	0.412	0.588	1776.52
50	50	20	0.588	0.588	1221.28	350	50	20	0.412	0.588	1777.21
50	50	50	0.588	0.588	1225.3	350	50	50	0.412	0.588	1750.55
50	70	1	0.588	0.588	1265.96	350	70	1	0.412	0.412	1869.68
50	70	3	0.588	0.588	1265.04	350	70	3	0.412	0.412	1749.7
50	70	5	0.588	0.588	1264.82	350	70	5	0.412	0.412	1516.21
50	70	10	0.588	0.588	1264.95	350	70	10	0.412	0.588	1753.02
50	70	20	0.588	0.588	1266.66	350	70	20	0.412	0.588	1790.92
50	70	50	0.588	0.588	1269.81	350	70	50	0.412	0.588	1778

Beam search is less successful than F-ARM in leukemia dataset as well as the colon cancer dataset. The reasons for this outcome may be the same as those are stated in the colon cancer dataset outcomes section. It can be observed that the results are mostly 0.588235. This is because the algorithm always finds the memberships for ALL test instances greater than AML test instances, resulting with the ratio 20/34 -i.e. the ratio of ALL test instances in the whole test instances set.

4.6. Algorithm CART

Coding of the CART algorithm has been done in MATLAB. The result for the leave-one-out validation of colon cancer data is shown in Table 4.26.

When constructing nodes in the decision tree, the most successful gene that divides the data into two most homogeneous partitions must be determined. In order to measure the talents of genes, two different performance measures are tested.

The effect of using two different performance measures can be seen in leave-1-out validation results. Before introducing the results it is possible to show the plotting of CART measure and entropy gain measure when CART measure improvement can be observed through the iterations of the algorithm, it is possible to observe that there are some deviations on the curve that represents entropy gain measure levels. These deviations cause the different decision tree structures therefore different accuracy results (Figure 4.9).

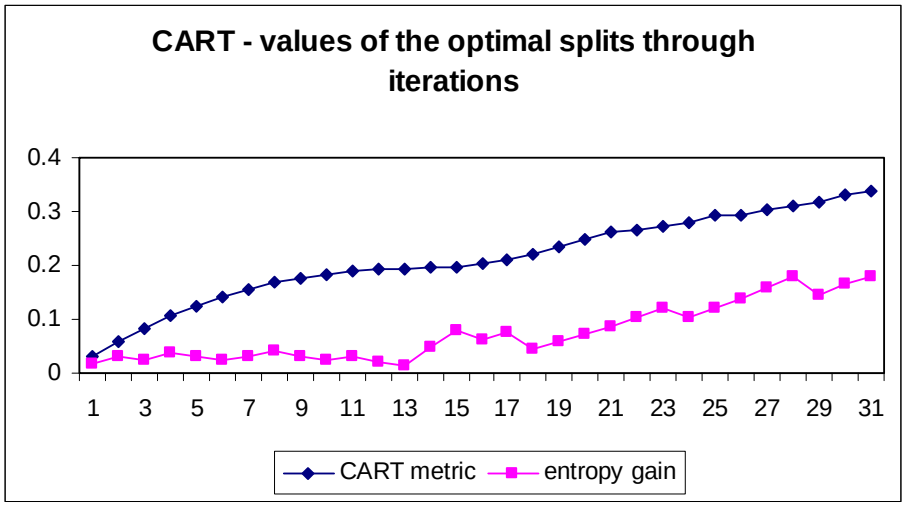

Figure 4.9 Comparison of CART measure and entropy gain

Implementation on Colon Cancer Dataset

Leave-1-out validation has been performed on colon cancer data. Accuracies can be seen in Table 4.26.

Table 4.26 Results for Leave-1-out validation for CART

performance metric used	CART measure	entropy gain
Classification accuracy(%)	98.39	96.77

In leave-1-out validation, for each test instance, a new decision tree is constructed. Different decision trees with two different measures for the same test instance can be observed in Figure 4.10. Numbers in nodes represent the label of gene in the dataset i.e. the row number.

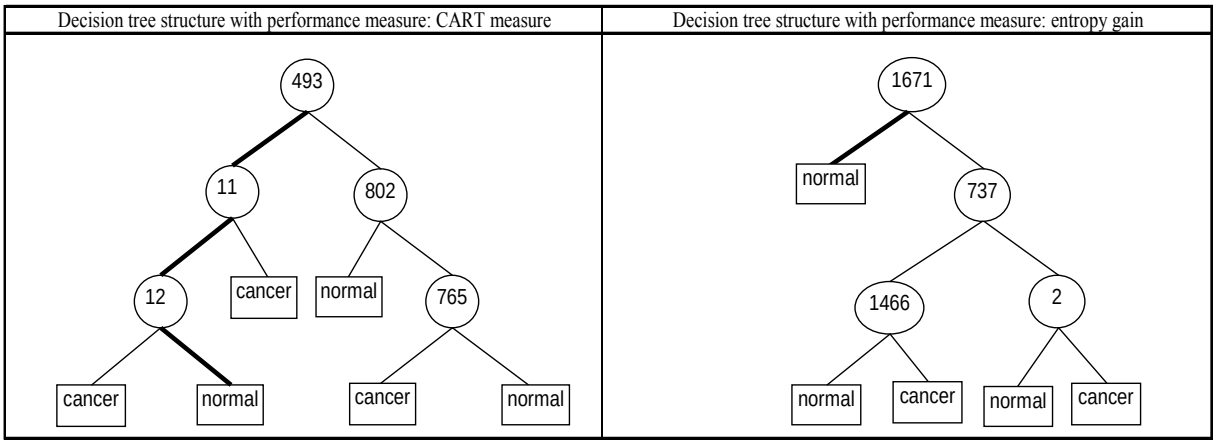

Figure 4.10 Decision trees constructed with different performance measures for selecting nodes, trees for test instance 3: normal tissue.

Implementation on Leukemia Dataset

CART was run on the training leukemia instance set, and the constructed single decision tree was tested on the testing leukemia instance set. Results are shown in Table 4.27.

Table 4.27 Results for testing Leukemia Dataset

performance metric used	CART measure	entropy gain
Classification accuracy(%)	86.84	86.84

The decision tree constructed includes only one node which is gene 4847 and the split point 994, the scatter plot of the expression levels of this gene can be observed in Figure 4.11. This single node is able to classify the training instance set with 100% accuracy, and can classify the testing instance set with 86.84% accuracy with both performance measures. Note that these results are less successful than those of F-ARM.

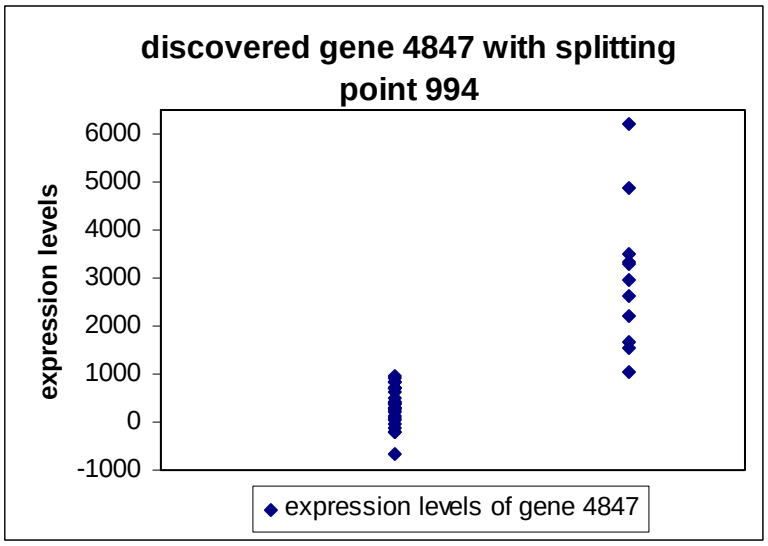

Figure 4.11 Expression level of the gene used in CART

CHAPTER 5

DISCUSSION AND CONCLUSIONS

This study focuses on association rule mining which has been initially constructed on market basket data, where each instance refers to a transaction. In this study we converted cell instances to instances and used fuzzy partitioning. Fuzzy c-means clustering the gene expression data enabled decreasing the risk of overfitting, and a different approach for the normalization of the data.

5.1. Algorithm F-ARM – Algorithm for Fuzzy Association Rule Mining

Results obtained by F-ARM show different performances on different datasets. Figure 4.3 and Figure 4.4 show the different memberships obtained by keeping the number of output rules 1, 3 and 5 respectively. All other parameters do not differ. It can be observed that as the number of rules that are used for classification increases, items that do not play role in the differentiation of two classes are more likely to occur in the rules in colon cancer dataset.

In order to increase classification accuracy in colon cancer dataset, we have also conducted experiments with an initial pruning of the genes, leaving out first 200 and first 700 successful genes, according to signal-to-noise ratio each gene's expression levels. The results did not show any improvements, inferring the conclusion that individually invaluable genes play role in rule generation process. Also the risk of overfitting is investigated with changing the validation scheme into 5-fold. Results gave lower classification accuracies.

The lack of memory problems are due to the APriori Rule Mining procedure. Using more advanced structures in coding and using different strategies for association rule mining, it may be possible to eliminate this problem or observing it in runs with lower support threshold levels.

It is a strong observation that F-ARM is much less successful in colon cancer dataset, when compared to Leukemia dataset. Also it can be observed that we could not make experiments on Leukemia Dataset with *threshold_support* levels as low as the experiments on Colon Cancer Dataset. This was because of the lack of memory problems. This is an evidence for the difference of the two datasets in their structure of the data.

We investigated the structure of genes in colon cancer and leukemia datasets. The first study was conducted with the normalized data of colon cancer and Leukemia training Datasets. We put the two datasets in a scale from 0 to 100, and computed signal-to-noise ratio for every gene. The values for Leukemia data are always greater than the values of colon cancer when put in descending order (Figure 5.1). It can be concluded that genes in the Leukemia Dataset are individually more successful, which can be counted for the success of F-ARM in classifying Leukemia Testing Dataset.

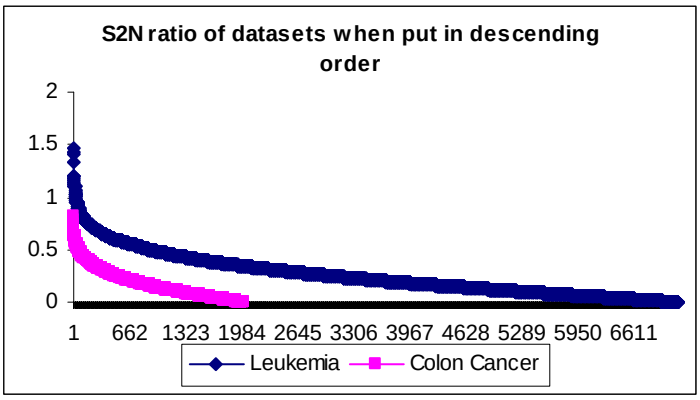

Figure 5.1 S2N ratio values of the datasets

Running F-ARM on the parameters 0.95 for *threshold_support* and 1 for *number_of_output rules* gave 100% accuracy, and Table 5.1 shows the genes that are involved in the rules discovered for classifying Leukemia testing Dataset. It can be concluded that rules generated are dominated by individually successful genes, however the genes that increase the confidence by using only a little portion of the instances are included in the itemset as long as the support of the itemset does not fall below the support threshold.

Table 5.1 Ranks of the genes discovered in F-ARM

	gene # of the items discovered	rank of the gene according to S2N ratio
ALL	2288	7
	6277	410
	7093	2507
AML	758	82
	2701	3281
	4050	355
	5435	6906
	5688	193
	6510	450

5.2. Algorithm Beam Search

The results evaluated with this algorithm in both of the datasets show lack of accuracy, which can be concluded that this algorithm needs serious improvements.

An issue that should be investigated is the *average_support* which is used as an expected support measure of the itemsets that are to be discovered. The relation between this measure with the real support values is investigated statistically. The average of the difference between *average_support* and real support has an average of 0.204 and standard deviation of 0.8786. These parameters yield the 95% confidence interval as [-1.518,1.926]. This deviation may result with filtering out the itemsets that are relevant with the classification or the opposite.

5.3. Algorithm CART

The decision tree constructed by CART on Leukemia Dataset includes only one node. Gene # 4847 with splitting point 994 can classify Leukemia Training Dataset instances with 100% accuracy, however is 86% accurate in classifying the Testing Instances. This gene is ranked as the third among all the genes according to S2N ratio. The rescaled expression levels of the first 3 genes of this ranking are shown in Figure 5.2.

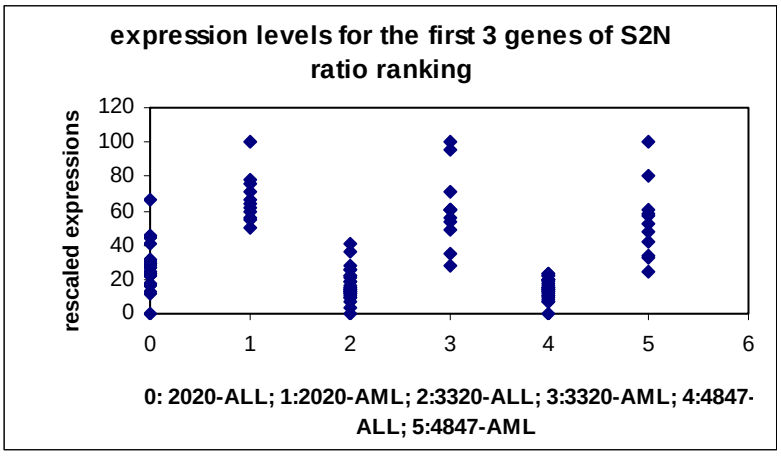

Figure 5.2 The first 3 genes of S2N ranking in Leukemia Data

Even gene# 2020 and gene# 3320 are more compact in expression levels – i.e. low standard deviations among same classes- the outlier instances are the reasons for not being selected by CART for decision tree construction. When these genes are used for the single nodes of decision tree construction, no improvement in the accuracy has been observed. Results are shown in Table 5.2. Results in accuracy did not change with the performance metric used.

Table 5.2 Accuracy results of the first and second genes of S2N ranking

Gene_used	2020	3220
Classification accuracy(%)	86.84	84.21

It is notable that in colon cancer dataset, the genes are of higher rank in the decision trees that are constructed according to CART metric. This may be related to the link between the CART metric and S2N metric.

Different folding schemes can be used for experimenting the power of decision trees in classifying cell instances using gene expression data. In addition, different metrics for identifying the splitting gene can be defined; even these metrics could be related to the biological function of the genes.

It can be observed that the beam search algorithm needs serious improvements, the average support calculation must be optimized in order to reduce the deviation from the real support values.

Algorithm CART is evaluating hierarchical rules, which means that, every element in the decision tree is utilized with only a subset of the whole instance set. Yet the rules mined

by F-ARM and beam search do not have this kind of structure. Items that make up the rule do not have individual meaning, all the items can be utilized on the whole instance set.

It must be considered that CART is a more stable algorithm evaluating good results with a reasonable amount of time. The results show that CART is at least 86% accurate on the whole experimented datasets. Even though F-ARM is more accurate in Leukemia dataset giving 100% accurate classification of testing instances, this algorithm is less accurate in colon cancer dataset. However, the opposite is valid for CART. The structure of the datasets may be the reason for these different accuracies in different algorithms.

The computational time of the algorithms F-ARM and beam search grows exponentially with decreasing support threshold. However, it should be noted that, the computational times can be considered as short when compared to algorithms with high number of iterations.

REFERENCES

[1] Watson, J., D., Crick, F., H., C., "A Structure for Deoxyribose Nucleic Acid," Nature (3), 171, pp. 737-738, 1953.

[2] Jun, L., Getz, G., Miska, E. A., Alvarez-Saavedra, E. *et. al.*, "MicroRNA expression profiles classify human cancers," Nature, Vol: 495-9, pp. 835-838, 2005.

[3] Saeys, Y., Inza, I., Larrañaga, P., "A review of feature selection methods in bioinformatics," Bioinformatics, Vol: 23-19, pp. 2507-2517, 2007.

[4] Pang-Ning, T., Steinbach, M., Kumar, V., *Introduction to Data Mining*. Pearson Education, Boston, 2006.

[5] Yong, S. K., "Toward a Successful CRM: variable selection, sampling and ensemble," Decision Support Systems, Vol:41-2, pp. 542-553.

[6] Schwab, I., Kobsa, A., Koychev, I., "Learning About Users from Observation," Adaptive User Interfaces, 2000.

[7] Law, M., H., C., Figueiredo, M., A., T., Jain, A., K., "Simultaneous Feature Selection and Clustering Using Mixture Models," IEEE Transactions on Pattern Analysis and Machine Inteligence, Vol:26-9, pp. 1154-1116, 2004.

[8] Guyon, I., Weston, J., Barnhill, S., Vapnik, V., "Gene Selection for Cancer Classification Using Support Vector Machines," Machine Learning, Vol: 46-1, pp. 389-422, 2002.

[9] Guyon, I., Elseiff, A., "An Introduction to Variable and Feature Selection," Journal of Machine Learning Research, Vol:3, pp.1157-1182, 2003.

[10] Xing, E., P., Jordan, M., I., Karp, R., I., "Feature selection for high dimensional genomic microarray data," Proceedings of the Eighteenth International Conference on Machine Learning, pp. 601-608, 2001.

[11] Hall, M., A., "Correlation-Based feature selection for discrete and numeric class machine learning," Proceedings of the Seventeenth International Conference on Machine Learning, pp. 359-366, 2000.

[12] Quinlan, J., R., *C.4.5: Programs for Machine Learning*. Morgan Kaufmann, 1992.

[13] Quinlan, J., R., "Induction of decision trees," Machine Learning, Vol:1-1, pp. 81-106. 1986.

[14] Kenji, K., Rendell, L., A., "A practical approach to feature selection," Proceedings of the Ninth International Workshop on Machine Learning, pp. 249-256, 1992.

[15] Kononenko, I., “Estimating Attributes: Analysis and Extensions of RELIEF,” Machine Learning: ECML-94, pp. 171-197, 1994.

[16] Robnik-Sikonja, M., Kononenko, I., “Theoretical and Emprical Analysis of ReliefF and RReliefF,” Machine Learning Journal, Vol: 53, pp. 23-69.

[17] Almuallim, H., Dietterich, T., G., “Learning with many irrelevant features”, Proceedings of the Ninth International Conference on Artificial Intelligence, pp. 547-552, 1991.

[18] Liu, H., Jinyang, L., Wong, L., “A Comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns,” Genome Informatics, Vol: 13, 51-60, 2002.

[19] Kohavi, R., John, G., H., “Wrappers for Feature Subset Selection,” Artificial Intelligence, Vol: 97-1, pp. 273-324, 1997.

[20] Liu, H., Setiono, R., “A Probabilistic approach to feature selection-a filter solution,” Proceedings of the 13th International Conference on Machine Learning, 1996.

[21] Zhang, H., Sun, G., “Feature selection using tabu search method,” Pattern Recognition, Vol: 35, pp. 701-711, 2002.

[22] Yang, J., Honavar, V., “Feature subset selection using a Genetic Algorithm,” IEEE intelligent systems & their applications, Vol: 13-2, pp. 44-49, 1998.

[23] Vafaie, H., De Jong, K., “Genetic algorithms as a tool for feature selection in machine learning,” Proceedings, International Conference on Tools with Artificial Intelligence, pp. 200-203, 1992.

[24] Handels, H., Ross, T., “Feature selection for optimized skin tumor recognition using genetic algorithms,” Artificial Intelligence in Medicine, Vol: 16-3, pp. 283-297, 1999.

[25] Liu, J. J., Cutler, G., Li, W., *et. al.*, “Multiclass cancer classification and biomarker discovery using GA-Based Algorithms,” Genetics and Population Analysis, Vol: 21-11, pp. 2691-2697, 2005.

[26] Küçükural, A., Yeniterzi, R., Yeniterzi, Sezerman, O. U., “Evolutionary Selection of Minimum Number of Features for Classification of Gene Expression Data Using Genetic Algorithms,” Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 401—406, 2007.

[27] Bloom, G., Yang, I., V., Boulware, D., *et. al.*, “Multi-Platform, Multi-Site, Microarray-Based Human Tumor Classification,” American Journal of Pathology, Vol: 164, pp. 9-16, 2004.

[28] Li, Z., Zhang, L., Chen, H., "Are filter methods very effective in gene selection of microarray data?," Bioinformatics and Biomedicine Workshops, pp. 97-100, 2007.

[29] Liu, X., Krishnan, A., Mondry, A., "An Entropy-Based gene selection method for cancer classification using microarray data," BMC Bioinformatics, Vol: 6:76, 2005.

[30] Berry, M., W., Browne, M., Lecture Notes in Data Mining, World Scientific, 2006.

[31] Creighton, C., Hanash, S., "Mining gene expression databases for association rules," Bioinformatics, Vol: 19-2, pp.79-86, 2003.

[32] Hipp, J., Güntzer, U., Nakhaeizadeh, G., "Algorithms for association rule mining-a general survey and comparison," ACM SIGKDD Explorations Newsletter, Vol: 2-1, pp. 58-64, 2000.

[33] Agrawal, R., Srikant, R., "Fast algorithms for mining association rules," Proceedings of the 20th VLDB Conference, 1994.

[34] Toivonnen, H., "Sampling large databases for association rules," Proceedings of the 22nd International Conference on Very Large Data Bases, pp.134-145, 1996.

[35] Zaki, M., J., Partasarathy, S., Ogihara, M. *et. al.* "New algorithms for fast discovery of association rules," 3rd Intl. Conf. on Knowledge Discovery and Data Mining, 1997.

[36] Savasere, A., Omiecinski, E., Navathe, S., "An efficient algorithm for mining association rules in large databases," Proceedings of the 21st International Conference on Very Large Data Bases, pp. 432-444, 1995.

[37] Cong, G., Kian-Lee, T., Tung, A., T., H., *et. al.* "Mining top-k covering rules for gene expression data," Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 670-681, 2005.

[38] Cong, G., Tung, A., T., H., Xu, X., "FARMER: Finding interesting rule groups in microarray datasets," Proceedings of the 2004 ACM SIGMOD international conference on Management of data, pp. 143-154, 2004.

[39] Jian, P., Jiawei, H., Mao, R., "CLOSET: An Efficient Algorithm for Mining Frequent Closed Item sets," ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2000.

[40] Zaki, M., J., Hsiao, C., CHARM: An Efficient Algorithm for Closed Item set Mining, 2002.

[41] Georgii, E., Richter, L., Rückert, U., *et. al.*, "Analyzing microarray data using association rules," Bioinformatics, Vol:21-2, pp. 123-129, 2005.

[42] Dubois, D., Hüllermeier, E., Prade, H., "A systematic approach to the assessment of fuzzy rules," Data Min Knowl Disc, Vol: 13, pp. 167-192, 2006.

[43] Hong, T., Kuo, C., Wang, S., "A fuzzy AprioriTid mining algorithm with reduced computational time," Applied Soft Computing, Vol: 5, 1-10, 2004.

[44] De Cock, M., Cornelis, C., Kerre, E., E., "Elicitation of fuzzy association rules from positive and negative examples," Fuzzy Sets and Systems, Vol: 149, pp. 73-85, 2005.

[45] Wai-Ho, A., Chan, K., C., C., "Mining changes in association rules: a fuzzy approach," Fuzzy Sets and Systems, Vol: 149, pp. 87-104, 2005.

[46] Sudkamp, T., "Examples, counter examples and measuring fuzzy associations," Fuzzy Sets and Systems, Vol: 149, pp. 57-71, 2005.

[47] Xiang-Rong, J., Gruenwald, L., "Microarray gene expression data association rules mining based on BSC-tree and FIS-tree," Data &Knowledge Engineering, Vol: 53, pp. 3-29, 2005.

[48] Kaya, M., Alhajj, R., "Genetic algorithm based framework for mining fuzzy association rules," Fuzzy Sets and Systems, Vol: 152, pp. 587-601, 2005.

[49] Becquet, C., Blachon, S., Jeudy, B., *et. al.*, "Strong-association-rule mining for large-scale gene-expression data analysis: a case study on human SAGE data," Genome Biology, Vol: 3-12, research 0067.1-0067.16, 2002.

[50] Yuanchen, H., Yuchun, T., Yan-Qing Z., *et. al.*, "Adaptive fuzzy association rule mining for effective decision support in biomedical applications," International Journal of Data Mining and Bioinformatics, Vol: 1-1, pp. 3-18, 2006.

[51] Icev, A., Ruiz, C., Ryder, E., F., "Distance-enhanced Association Rules for Gene Expression," BIOKDD03: 3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics, 2003.

[52] Rodriguez, A., Carazo, J., M., Trelles, O., "Mining Association Rules from Biological Databases," Journal of the American Society for Information Science and Technology, Vol: 56-5, pp. 493-504, 2005.

[53] Lowerre, B., "The HARPY speech recognition system," PhD Thesis, Carneige Mellon University, 1976.

[54] Ow, P., S., Morton, T., E., "The Single Machine Early/Tardy Problem," Management Science, Vol: 35-2, pp. 177-191, 1989.

[55] Ney, H., Mergel, D., Noll, A., "A Data-driven organization of the dynamic programming beam search for continuous speech recognition," Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '87, Vol: 12, pp. 833-836, 1987.

[56] Alleva, F., Huang, X., Hwang, M., Y., "An improved search algorithm using incremental knowledge for continuous speech recognition," Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference, Vol:2, pp. 307-310.

[57] Nguyen, L., Schwartz, R., "Single-tree method for grammar-directed search," Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings. IEEE International Conference, Vol: 2, 613-616, 1999.

[58] Carlson, J., M., Chakravarty, A., Gross, R., H., "BEAM: A Beam Search Algorithm for the Identification of Cis-Regulatory Elements in Groups of Genes," Journal of Computational Biology, Vol: 13-3, pp. 686-701, 2006.

[59] Slonim, D., K., "From patterns to pathways: gene expression data analysis comes of age," Nature Genetics Supplement, Vol: 32, pp. 502-508, 2002.

[60] Breiman, L., Freidman, J., Olshen, R., Stone, C., *Classification and regression trees.* Wadsworth International, California, 1984.

[61] Yoon, H., C., Jae, K., K., Soung, H., K., "A personalized recommender system based on web usage mining and decision tree induction," Expert Systems with Applications, Vol: 23-3, pp. 329-342, 2002.

[62] Nelson, L., M., Bloch, D., A., Longstreth Jr., W., T., *et. al.*, "Recursive partitioning for the identification of disease risk subgroups: a case-control study of Subarachnoid Hemorrhage," Journal of clinical epidemiology, Vol: 51-3, pp. 199-209.

[63] Wu, B., Abbott, T., Fishman, D., McMurray, W., *et. al.*, "Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data," Bioinformatics, Vol: 19-13, pp. 1636-1643, 2003.

[64] Bezdek, J., C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers Norwell, MA, USA, 1981.

[65] Thedoridis, S., Koutroumbas, K., *pattern Recognition*, Elsevier Academic Press, San Diego, USA, 2006.

[66] Aktürk, M., S., Kılıç, K., "Generating short-term observation schedules for space mission projects," Journal of Intelligent Manufacturing, Vol: 10-5, pp.387-404, 2004.

[67] Jin, R., Agrawal, G., "Efficient decision tree construction on streaming data," Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 571-576, 2003.

[68] Alon, U., Barkai, N., Notterman, D., A., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide

arrays," Proceedings of the National Academy of Sciences of the United States of America, Vol: 96-12, pp. 6745-6750, 1999.

[69] Golub, T., R., Slonim, D., K., Tamayo, P., *et. al.*, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," Science, Vol: 286-5439, pp. 531-537, 1999.

[70] Jirapech-Umpai, T., Aithen, S., "Feature selection and classification of microarray data analysis: Evolutionary methods for identifying predictive genes," BMC Bioinformatics, Vol: 6-148, 2005.

[71] Yuhang, W., Makedon, F., S., Ford, J., C., Pearlman, J., "HykeGene: A Hybrid approach for selecting marker genes for phenotype classification using microarray gene expression data," Bioinformatics, Vol: 21-8, pp. 1530-1537, 2005.

[72] Bø, T., H., Jonassen, I., "New feature subset selection procedures for classification of expression profiles," Genome Biology, Vol: 3-4, pp. 1-11, 2002.

[73] Fisher, R., A., "The use of multiple measurements in taxonomic problems," Annals of Eugenics, Vol: 7, pp. 179-188, 1936.

[74] Ben-Hur, A., Horn, D., Siegelmann, H., T., *et. al.*, "Support vector clustering," Journal of Machine Learning Research, Vol: 2, pp. 125-137, 2001.

[75] Tsang, E., C., C., Yeung, D., S., Wang, X., Z., "OFSS: Optimal fuzzy valued feature subset selection," IEEE Transactions on Fuzzy Systems, Vol:11-2, pp. 202-213, 2003.

[76] Mylonas, P., Wallace, M., Kollias, S., "Using k-nearest neighbour and feature selection as an implement to hierarchical clustering," Lecture Notes in Computer Science, Vol: 3025, pp. 191-200, 2004.

[77] Ahmandian, K., Golestani, A., Mozayani, N., *et. al.*, "A new multi-objective evolutionary approach for creating ensemble of classifiers," IEEE Interntional Conference on Systems and Cybernetics, pp. 1031-1036, 2007.

[78] Hathaway, R., J., Bezdek, J., C., "Visual cluster validity for prototype generator clustering models," Pattern Recognition Letters, Vol: 24-9-10, pp. 1563-1569, 2002.

[79] Li, L., Weinberg, C., R., Darden, T., A., *et. al.*, "Gene selection for cancer classification based on gene expression data: study of sensitivitiy to choice of parameters of the GA/knn method," Bioinformatics, Vol: 17-12, pp. 1131-1142, 2001.

[80] Tan, A., C., Gilbert, D., "Ensemble machine learning on gene expression data for cancer classification," Applied Bioinformatics, Vol: 2-3, pp. 75-83, 2003.