5-2016

Global Entropy Based Greedy Algorithm for discretization

Sai Jyothsna Jonnalagadda
The University of Texas Rio Grande Valley

GLOBAL ENTROPY BASED GREEDY ALGORITHM FOR DISCRETIZATION

A Thesis

by

SAI JYOTHSNA JONNALAGADDA

Submitted to the Graduate College of
The University of Texas Rio Grande Valley
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2016

Major Subject: Computer Science

GLOBAL ENTROPY BASED GREEDY ALGORITHM FOR DISCRETIZATION

A Thesis
by
SAI JYOTHSNA JONNALAGADDA

COMMITTEE MEMBERS

Dr. Dongchul Kim
Chair of Committee

Dr. Bin Fu
Committee Member

Dr. Xiang Lian
Committee Member

Dr. Timothy Wylie
Committee Member

May 2016

ABSTRACT

Jonnalagadda, Sai Jyothsna. Global Entropy Based Greedy Algorithm for Discretization. Master of Science (MS), May, 2016, 48 pp., 15 tables, 6 figures, 60 references, 30 titles.

Discretization algorithm is a crucial step to not only achieve summarization of continuous attributes but also better performance in classification that requires discrete values as input. In this thesis, I propose a supervised discretization method, Global Entropy Based Greedy algorithm, which is based on the Information Entropy Minimization. Experimental results show that the proposed method outperforms state of the art methods with well-known benchmarking datasets. To further improve the proposed method, a new approach for stop criterion that is based on the change rate of entropy was also explored. From the experimental analysis, it is noticed that the threshold based on the decreasing rate of entropy could be more effective than a constant number of intervals in the classification such as C5.0.

DEDICATION

The Completion of my Master's would not have been possible without the blessings and Divine support of Shri Shirdi Sai Baba. I would like to dedicate my work to my family. My mother, Vema Narayanamma, my father, Ravichandraiah, my brother, Krishna Prasad and my sister-in-law, Ioana Lazarescu and my nephew, Ishan Peter Jonnalagadda.

ACKNOWLEDGEMENTS

I would like to express my truest and sincere thanks to Dr. Dongchul Kim, Department of Computer Science, UTRGV, for his reverence supervision, invaluable guidance, inspiring discussions, care and encouragement throughout this Master's work. His ideas, stimulating comments, interpretations and suggestions increased my cognitive awareness and have helped me considerably to reach my future goals. I remain obliged to him for his help and guidance throughout all stages of this. His constant inspiration and encouragement towards my efforts shall always be acknowledged.

I would like to thank Dr. Dongchul Kim who helped me throughout the Experimentation analysis and results.

I am also grateful to all committee members Dr. Bin Fu, Dr. Xiang Lian, Dr. Timothy Wylie for evaluating my thesis.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ACRONYMS

AI	Artificial Intelligence
ML	Machine Learning
IEM	Information Entropy Minimization
MDLP	Minimum Description Length Principle
CAIM	Class Attribute Interdependent Maximization
CACC	Class Attribute Contingency Coefficient
GA	Genetic Algorithm
GE	Global Entropy
NBC	Naïve Bayes Classifier
CV	Cross Validation
AQ	A^q (quasi-optimal) Algorithm

CHAPTER 1

INTRODUCTION

Artificial Intelligence (AI) is a system by which computers can think and act like humans and make intelligent decisions using a given set of data. Machine Learning (ML), a prominent and growing subdomain of AI, is the process by which a system can efficiently recognize patterns in a given dataset by virtue of computational learning techniques, predict decision-making. Essentially, ML is based on a conventional AI approach but focusses more on self-learning from data without explicit predefined rules or instructions. Several scientific advancements such as robotics, computer vision, bioinformatics, etc. are underpinned by ML, which has resulted in its prominence as a key research topic both in academia and industry. Among the various research topics associated with ML, classification is the most important and serves as the backbone of ML process.

In ML, classification is the process that groups the output into specific classes. An algorithm which implements the classification process is called a classifier. Classification has been the focus of many researchers and has been applied to a variety of applications such as hand written digit recognition [HTA2004], malicious code detection [ZSJ2006], face detection and recognition and biomarker discovery [GDM2006]. Some classification algorithms such as AQ [KAR1999], CN2 [CPT1989], CLIP4 [CKL2004] can only handle numerical or nominal data, while some others can handle continuous attributes but perform well with discrete valued attributes [CTJ1991], [TCL2008]. To deal with this problem, several discretization algorithms have been developed [FUI1992], [DKS1995].

Discretization is a pre-processing mechanism that is used with ML algorithms that can only handle discrete data. In ML, it enhances the learning process and makes it more accurate and faster. The primary goal of a discretization algorithm is to transform continuous datasets into discrete ones by creating a set of intervals that span across the continuous datasets. For example, many discretization algorithms based on information entropy such as maximum entropy, which discretizes using the criterion of minimum information loss and Information Entropy Minimization (IEM) [KIR1995], [IKB1993], focus on efficiency and good performance. IEM consists of two parts, splitting criterion to divide an interval into two intervals and the other is the stopping criterion which terminates the iterative splitting algorithm.

In this thesis, we propose a greedy discretization algorithm and explore a heuristic stopping criterion to improve the algorithm. As a performance evaluation, we first compare state of the art methods such as IEM, Class Attribute Interdependency Maximization (CAIM) [KLJ2004] and Class Attribute Contingency Coefficient (CACC) [TCL2008] with well-known bench marking datasets. We then compare these methods with the proposed methods, using a different number of intervals, to show that the proposed methods can outperform the state-of-the-art methods when the number of intervals is properly defined. In addition, our experiment validated that the proposed heuristic stop criterion with decreasing rate of entropy could provide better performance than a number of intervals approach.

This thesis is organized into different chapters each dealing with a specific task of the research. Chapter 1 deals with introduction, research goals of the project, and the organization of the thesis. Chapter 2 gives an overview of Discretization and Classification processes. Chapter 3 analyzes different types of discretization methods. It gives detailed explanations of different types of methods and their criterion and also compares existing methods with newly implemented methods. Chapter 4 explores the idea of a new stopping criterion using a greedy approach for state of the art discretization algorithms. Chapter 5 presents experimental analysis by comparing performance and accuracy with different types of discretization algorithms and discusses the best results. Chapter 6 concludes with a summary of the thesis.

CHAPTER II

AN OVERVIEW OF DISCRETIZATION AND CLASSIFICATION PROCESS

Data discretization is the process of simplifying large amounts of raw data by reducing the number of continuous values (or variables) by partitioning them into specific intervals. It is generally used in data-preprocessing for ML algorithms that are limited to the use of discrete data. Careful selection of an effective discretization method is extremely important to produce new and more accurate patterns. Effective discretization improves the efficiency of the machine learning algorithm and enhances the knowledge extracted from discretized dataset easy to understand and more useful. Discrete features also reduce memory usage and enhance representation of the knowledge. Discretization can significantly impact the performance of classification algorithms and has important implications in analyzing large and highly complex datasets. A well-developed discretization algorithm can not only reduce the continuous attributes into discrete ones for better understanding but can also make classification effective and efficient. Most importantly, a carefully developed discretization algorithm can significantly improve the accuracy and performance in classification.

2.1 Classification

Classification is a process that classifies given data based on the training set and the values in a classifying attribute and uses it in classifying new data. Classification is a supervised learning technique used to assign instances to pre-defined classes. Generally, a classification model is created from the training data and is used to classify new instances or unknown cases and also estimate the accuracy of the model. However, features (or attributes) for classification

data are often numerical (or continuous). Some of the classification algorithms can take nominal data as input and others discretize numeric data into nominal data during the learning process. Methods used to evaluate classification are accuracy, speed, robustness, scalability, interpretability and other measures such as goodness of rules, or compactness of the classification rules.

Numerous methods have been proposed for classification and data mining in the literature, some examples include Decision trees [CRM2007], Genetic algorithms [KSV2007], Neural networks [CFG2007] and Bayesian classification [RRY2006]. Among them, decision trees like C5.0 and Naïve Bayes Classifier (NBC) are widely used, efficient and also display a good classification when compared with other techniques. A decision tree like C5.0 [RQN2005] is a flowchart that resembles a tree structure, which can be built by using a recursive divide and conquer algorithm that partitions the data. Alternatively, the Naïve Bayes classifier (NBC) is based on Bayes theorem with strong independence assumptions between attributes. The main advantage of NBC is that it requires a small amount of training data to estimate the parameters and uses the maximum likelihood method. Therefore, both C5.0 and NBC have been used in many unsupervised and supervised discretization algorithms such as Equal Width and Equal Frequency [CWG1991], IEM [IKB1993], CAIM [KLJ2004], CACC [TCL2008], to evaluate accuracies and the number of intervals with different datasets.

2.1.1 Naïve Bayes Classifier

A Naïve Bayes Classifier (NBC) is a simple probabilistic classifier based on applying Bayes theorem with strong or naïve independence assumptions. Bayes theorem was named after Thomas Bayes (1702 –1761), who invented a method to compute the distribution for the probability parameter of a binomial distribution. It is normally used in machine learning and is

typically a collection of classification algorithms based on Bayes theorem. Naïve Bayesian model is easy to build and is useful in analyzing large and complex datasets. Despite its simplicity, the Naïve Bayesian Classifier often outperforms more sophisticated classification methods. Essentially, NBC uses Bayes theorem for calculating the posterior probability. The Naïve Bayes classifier assumes that the effect of the value of a predictor (X) on a given class (C) is independent of the values of other predictors; this assumption is also called class conditional independence. Bayes theorem is as follows:

:

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)} \qquad (1)$$

where, $P(C|X)$is the posterior probability of class (target) given predictor (attribute), $P(C)$ is the prior probability of class, $P(X|C)$ is the likelihood, which is the probability of a predictor given a class, and $P(X)$ is the prior probability of predictor.

The Naïve Bayes algorithm is a classification algorithm, which is based on applying Bayes theorem with the naïve assumption of independence between every pair of features. Given a class variable *C* and a dependent feature vector x_1 through x_n, Bayes theorem states the following relationship:

$$P(C|x_1, \dots, x_n) = \frac{P(C)P(x_1, \dots, x_n|C)}{P(x_1, \dots, x_n)} \qquad (2)$$

Using the naïve independence assumption, equation (2) can be written as

$$P\left(x_i\middle|C, x_1, \dots, x_{i-1,}\, x_{i+1}, \dots, x_n\right) = P(x_i|C) \qquad (3)$$

for all i, this relationship is simplified to

$$P(C|x_1, \dots, x_n) = \frac{P(C)\prod_{i=1}^{n} P(x_i|C)}{P(x_1, \dots, x_n)} \quad (4)$$

From this assumption, we can assume $P(x_1, \dots, x_n)$ is constant and can be written as:

$$P(C|x_1, \dots, x_n) = P(C)\prod_{i=1}^{n} P(x_i|C) \quad (5)$$

The naïve Bayes classifier combines this model with a decision rule (which is known as *maximum a posteriori* or *MAP* decision rule. Hence, the corresponding classifier in the function *classify* can be defined as follows:

$$Classify\ (x_1, \dots, x_n) = argmax_C P(C)\prod_{i=1}^{n} P(x_i|C)$$

Although it's a simple classification, it can outperform other more sophisticated algorithms and is extremely useful in many applications such as spam detection, document classification, etc. The main advantages of this algorithm are that it is fast, simple and easily trained with a small dataset. The main disadvantage is that it assumes that every feature is independent, which is not always possible, particularly in practical applications. Despite its challenges, NBC is commonly used in the industry and as such we have considered it in our experiment.

2.1.2 C5.0

The C5.0 classifier is an algorithm used in machine learning and is based on decision trees [RQR2011]. The C5.0 algorithm is an extension of C4.5 algorithms, which iteratively visits each decision node to select the optimal split. This process is continued until no further split is possible. The algorithm mainly uses the concept of information gain or entropy reduction to

select the optimal split. Information gain is the increase in information produced by partitioning the training data according to the candidate split. C5.0 algorithm chooses the split with highest information gain as the optimal split. The information gain measure is used to select the best attribute at each node in the decision tree. The C5.0 method uses post-pruning method and thus enhances the accuracy of the classification. C5.0 has many features such as large decision trees, which can be viewed as a set of rules; acknowledgment of missing and noisy data, and error pruning. The C5.0 classifier contains a simple command line interface that makes it easy to generate the decision trees. In classification technique, the C5.0 classifier can anticipate which attributes that are relevant or from those that are not. Compared to C4.5 and other state of the art discretization algorithms, C5.0 provides good accuracy and automatically eliminates unhealthy attributes. C5.0 also classifies the data in less time compared to other classifiers. Given that, C5.0 is a commercial version of C4.5 which uses less memory, is fast, and builds smaller rule sets than C4.5, while being more accurate, we selected C5.0 classifier to compare classification accuracy and number of intervals in our experiment.

Description of the C5.0 algorithm is provided in Table 1.

INPUT: Example, Target Attribute, Attribute
OUTPUT: Classified instances
Algorithm
STEP 1: To make the tree, create a root node.
STEP 2: Check for the base cases.
STEP 3: Construct a Decision Tree using training data.
STEP 4: Find the attribute with the highest Info gain (A_Best).
STEP 5: A_Best is assigned with Entropy Minimization.
STEP 6: Create a decision node that splits on A_Best.
STEP 7: According to A_Best, Split into different partitions.
STEP 8: Repeat on the sublists obtained by splitting on A_Best.
By applying reduced Error-pruning technique, classification can be defined as follows:
STEP 9: For each training data (t_i), $t_i \in D$, apply the Decision Tree to determine its class.

Table 1: C5.0 Algorithm

2.2 Discretization

Discretization is a data pre-processing technique, which is used to convert numerical attributes into categorical attributes, using data mining techniques that can be used in a classifier. Discretization can be performed repetitively on an attribute data set to select the best cut points that splits continuous value ranges into discrete number of bins usually referred to as states.

2.2.1 Supervised Discretization Process

A Supervised discretization process is used to find a set of cut points to partition a

range of continuous values into a small number of sub-intervals. Two key aspects of discretization are: finding the number of intervals with the help of the user and determining the width and boundaries of a given range of continuous values. Typically, the process involves four steps: 1) Sorting the continuous values to be discretized; 2) Evaluating cut-point for splitting or adjacent intervals for merging; 3) Splitting or merging intervals of continuous values according to some criterion; and 4) Stopping at a point based on some criterion. Figure 1 presents a schematic.

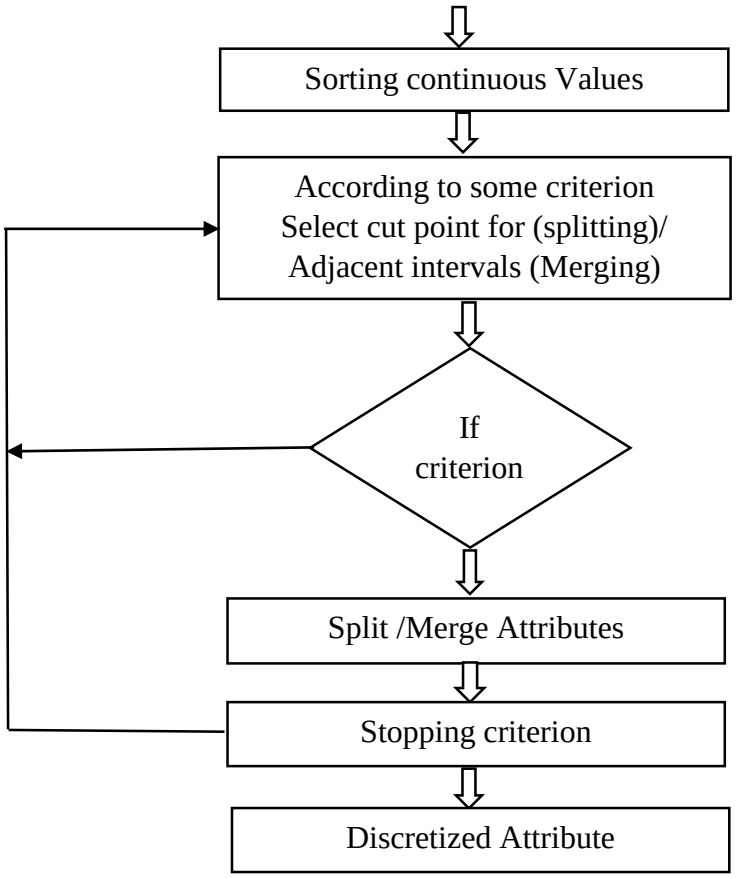

Figure 1: Supervised Discretization Process

2.2.2 Categories of Discretization Methods

There are several ways by which discretization methods can be categorized. The following categories are mentioned in some research papers [DKS1995], [DNA2007]. Some of these methods are *Supervised vs Unsupervised, Splitting vs Merging, Local vs Global, Static vs dynamic and Non-incremental vs incremental.* A brief explanation of these methods is provided in this sub-section.

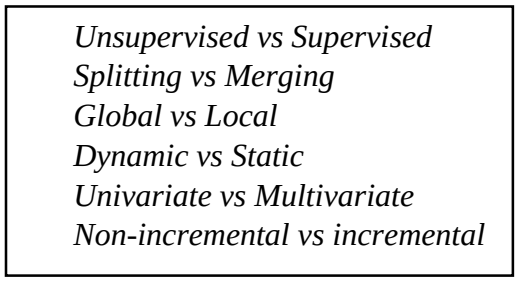

Figure 2: Categories of Discretization Methods

Unsupervised methods can be discretized without the knowledge of class label whereas *Supervised* methods use class information to carry out discretization. The second category is *Splitting vs Merging* in which *Splitting* is a top-down method starts with an empty set of cut points and gradually divides these intervals and subintervals to obtain discretization. In contrast, *Merging* is a bottom-up approach that considers all the possible cut points and then eliminates these cut points by merging intervals. The third is *Local vs Global* where *Local* methods produce partitions that are applied to localize regions of instance space and *Global* methods use the entire space and form a mesh over entire n-dimensional continuous instance spaces, where each feature is partitioned into regions independent of other attributes. The fourth one is *Static vs Dynamic* in which *Static* methods require some parameters such that it indicates the largest number of intervals to discretize a feature and it is done prior to the classification task and on the opposite side, dynamic methods conduct a search through the space of possible *k* values for all features

simultaneously. The fifth one is *Univariate vs Multivariate*, where *Univariate* takes one feature at a time and *Multivariate* considers multiple features simultaneously. The last is *non-incremental* methods that consider only the available historical data values. By taking time into consideration, data values can be completely new and are not considered while *incremental methods* try to cover the new values.

CHAPTER III

DISCRETIZATION METHODS

Discretization methods can be analyzed and are separated into two important groups being top-down versus bottom-up in which the top-down methods use Unsupervised (or class bind) algorithms discretize without using having the knowledge of class information. Some popular unsupervised discretization algorithms are equal width and equal frequency [CWG1991]. Supervised discretization algorithms discretize continuous attributes and have additional knowledge by considering class information. Some of these state of art supervised top-down and bottom-up algorithms include Paterson-Niblett [APN1987], chi Merge [RKB1992], chi2 [HLS1997], Maximum Entropy [AWC1987], CADD (Class-Dependent Discretization for Inductive Learning from continuous and Mixed-Mode data) [JKC1995], IEM (Information entropy Maximization) [IKB1993], FCAIM (Fast Class Attribute Interdependence Maximization) [KLJ2003], CAIM (Class-Attribute Interdependence Maximization) [KLJ2004], CACC (Class-Attribute Contingency Coefficient) [TCL2008], DCR (Class Attribute Information to reduce Number of intervals) [PTK2009], UCAIM (Uncertain CAIM) [JYT2010], CACM (Class Attribute Coherence Maximization) [LMN2011], MCAIM (Modified CAIM) [VSM2012], ECAIM (Enhanced Class-Attribute Interdependence Maximization) [SKW2012], Ur-CAIM (Improved CAIM Discretization algorithm) [CAO2016], CAIA (Class-Attribute Interval Average) [BAS2014].

In this chapter, we discuss two unsupervised discretization algorithms and three supervised discretization algorithms, methods, criterion and their importance when compared to other state of the art supervised discretization algorithms to analyze the performance, accuracy and how they provide an optimal solution to the problem. The following methods are discussed based on several research papers [DKS1995], [IKB1993], [KLJ2004], [TCL2008]. The two common unsupervised discretization methods considered are Equal Width, which requires the user specified number of intervals and adopts the heuristic formula also used in *caim* to estimate the number of discrete intervals and Equal Frequency, which is same as equal width in specifying the intervals with user supervision and it divides the range so that every interval contains the equal number of distributed sorted values. The main drawback of both these methods is the user input of intervals and the uneven distribution of values in case of Equal Width discretization. Three supervised discretization methods were considered: IEM (based on entropy), CAIM and CACC which uses the top-down (splitting) approach and provide the best results when compared with other state of the art algorithms. The main idea behind the IEM approach is to find a potential cut-point to split a range of continuous values into two intervals and also use class information entropy to select boundaries for discretization. It considers one big interval and then recursively partition this big interval into smaller sub intervals such that the stopping criterion (such as MDLP) satisfies. To overcome this, we use a better approach by using a greedy method in IEM as a stop criterion, more advanced approach than MDLP. The other two important supervised discretization algorithms are CAIM and CACC, such that CAIM is one of the most progressing algorithm which does not have user supervision and the main use of *caim* is it will test all possible cut-points and generate one in each loop for a continuous attribute and stops until it satisfies the specific condition and it also finds local maximum CAIM value.

Although it outperforms the other state of the art methods, it still has some limitations such as not considering the data distribution and also number of intervals is close to the target classes. In the case of CACC, it introduces a new term named the contingency coefficient to measure the strength of the dependence between variables. It introduces log(*n*) to reduce the huge influence of variable *n* which increases the discretization process. Both methods use a greedy approach to generate the sub-optimal discretization scheme but have a searching and stopping criterion problem. In this thesis, we focus on improving the stopping criterion problem in the greedy algorithm, check to find out whether cut points are more or less than *S,* and evaluate their potential to improve the threshold as a stopping criterion and show better performance by testing different intervals for the three supervised discretization algorithms.

3.1 Unsupervised Discretization Methods

Among the unsupervised discretization methods, the simple ones are Equal Width and Equal Frequency and the more sophisticated ones are based on the clustering analysis (such as k-means discretization). They divide continuous ranges into sub-ranges based on user specified width or frequency. Each of these two discretization methods are discussed in detail in this section.

3.1.1 Equal Width Discretization (EWD)

Equal Width discretization is the simplest method, which is used to divide the range of observed values for a feature into *N* equal sized bins, where *N* is the parameter, which is provided by the user defining the specified number of intervals. This process involves solving the observed values of a continuous feature by finding the minimum and maximum values and

the interval can be computed by dividing the range of observed values into N equally sized bins from the equation below.

$$Interval = \frac{X_{max} - X_{min}}{N} \qquad (7)$$

$$Boundaries = X_{min} + (\, i \times interval\,) \qquad (8)$$

The boundaries can be built by $i = 1 \ldots N$ using equation (8). However, this type of discretization is sensitive to outliers and may drastically skew the range. The drawbacks of this method are uneven distribution of data points in which some intervals may have more data points than other which is not well distributed. An example is provided in Table 2

Data values	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
Original values	10	20	25	15	30	40	35	45	50	60
Sorted values	10	15	20	25	30	35	40	45	50	60

$$Interval = \frac{X_{max} - X_{min}}{N} = \frac{60 - 10}{2} = \frac{50}{2} = 25 \quad N = 2$$

$$Boundaries = X_{min} + i \times Interval = 10 + 1 \times 25$$

Table 2: Equal Width Example

3.1.2 Equal Frequency Discretization (EFD)

Equal Frequency discretization is another simple form of discretization, which is similar to equal width but with some differences. It is used to determine the minimum and maximum value of the discretized attributes and sort all values in ascending order and divide the sorted continuous values into N equally sized bins such that each interval contains approximately n/N data instances with adjacent values. This algorithm tries to overcome the equal width discretization by dividing the domain in intervals with the same distribution of data points, which

is a little different compared to equal width discretization. The data instances with identical value must be placed in the same interval and it is not possible to generate exactly *N* equal frequency intervals. Since, these unsupervised discretization methods do not utilize instance labels in setting partition boundaries, the main problem is it is not always possible to generate exactly *N* equal frequency intervals because it tries to replace the data instance with an equivalent value in the same interval *N = 2.* Each interval will contain 10/2 = 5 instances. An example is provided in Table 3.

Data values	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
Original values	10	20	25	15	30	40	35	45	50	60
Sorted values	10	15	20	25	30	35	40	45	50	60
N = 2, So, each interval contains 10/2 = 5 instances										
instances	5	5	5	5	5	5	5	5	5	5

Table 3: Equal Frequency Example

3.2. Supervised Discretization Methods

As discussed above, various Supervised Discretization Methods are proposed in the literature but among these methods, entropy-based discretization, interval splitting and merging methods are the easiest to use. Supervised Discretization methods make use of class labels when partitioning the continuous features. We considered three methods in our experiment, each of these methods are discussed in detail in this section.

3.2.1 Information Entropy Minimization (IEM)

Fayyad and Irani proposed a method of discretization based on the entropy measure, also called as entropy minimization heuristic and this approach of discretization begins by sorting all the instance values of an attribute in ascending order. It then identifies the probable cut-points by

examining the class of each instance; if the class value changes between two instance values, then the mid-point of these values can be considered as a potential cut-point. After finishing a list of complete potential cut-points, it evaluates each potential cut point.

$$entropy(S) = -\sum_{i=1}^{n} p_i \, log_2 p_i \qquad (9)$$

where, p_i is the probability of class *i* and is estimated as c_i/S , c_i being the total number of data instances of class i. A log function of base *2* is used because the information is encoded in bits and *n* is the total number of instances. Based on this entropy measure, J. Ross and Quinlan developed an algorithm called ID3 to induce best split in decision trees. ID3 employs a greedy search to find potential split points within existing range of continuous values using the formula:

$$entropy\ (S,T) = -p_{left} \sum_{j=1}^{m} p_j\ , left\ log p_j, left - p_{right} \sum_{j=1}^{m} p_j\ , right\ log p_j, right \qquad (10)$$

From the above equation, $p_j left$ and $p_j right$ are the probabilities that an instances belong to class *j*, which is on the left or right side of the potential split-point *T* and *m* is the number of classes. The cut-point minimized in equation (10) is the best split point and it is used to separate initial values into two subsets. This process is repeated recursively on each of the halves and continues until algorithm reaches a stopping criterion. The stopping criterion of the entropy method is MDLP (Minimum Description length principle). It will stop discretization only if the best split at the given level results in a lower information gain than the amount of information it would cost to encode the theory being created by the new split.

$$gain < \frac{log(N-1) + log_2(3^k - 2) - Eentropy(S) + E_1 entropy(S_1) + E_2 entropy(S_2)}{N} \qquad (11)$$

This can be calculated using equation (9), where N is the number of instances, c is the number of classes, E is the measure of entropy for the existing split, E_1 and E_2 measure the entropy in the proposed splits, with S_1 and S_2 as the number of classes in the proposed splits.

Entropy minimization evaluates as a candidate cut-point, which is the mid-point between each successive pair of the sorted values. For this process to be in execution, to evaluate each candidate cut-point, the data are discretized into two intervals so that the resultant class information entropy is calculated. A binary discretization is determined by selecting by choosing the cut-point so that the entropy is smallest amongst all other candidate cut-points. So, the binary discretization is applied recursively so that it always selects the best cut-point. Hence, as mentioned earlier, to stop this discretization, a minimum description length is applied. The main use of this discretization algorithm is it makes use of class labels when partitioning the continuous features. A detailed description of the IEM algorithm is provided in Table 4.

Step 1

1.1 Select data values from minimum to maximum.

1.2 Select original continuous values based upon step 1.1.

1.3 Use class labels when partitioning the continuous features before sorting values.

1.4 Sort continuous values in ascending order.

1.5 Find the cut-points in the continuous attributes based on the minimum and maximum values of each class.

Step 2

2.1 Compute the class information entropy by considering each data value as a split point.

2.2 Takes the one that generates the best gain out of all possibly splitting values.

2.3 A binary discretization is determined by selecting the bin boundary for which the entropy is minimal amongst all candidates.

2.4 The above method in step 2.3 can be recursively applied until stopping criterion is achieved.

2.5 Minimum Description Length Principle (MDLP) is proposed to decide when to terminate discretization process.

Table 4: IEM Algorithm

3.2.2 Class Attribute Interdependency Maximization (CAIM) Discretization

Discretization transforms a continuous attribute's values into a finite number of intervals and associates with each interval a numerical, discrete value. For mixed mode

(continuous and discrete) data, discretization is usually performed prior to the learning process. Discretization is a two-step process. The first process is to find the number of discrete intervals. Only a few discretization algorithms execute this automatically, but the user must designate the number of intervals or provide a heuristic rule. The second process is to find the width or the boundaries of the intervals given the range of values of a continuous attribute. Our proposed CAIM algorithm performs both tasks by automatically selecting a number of discrete intervals and finding the width of every interval based on the interdependency between classes and attribute values at the same time.

The CAIM algorithm not only discretizes an attribute into a small number of intervals, but also makes it much easier for the subsequent machine learning task by maximizing the class-attribute interdependency. The algorithm does not require user supervision since it automatically selects the proper number of discrete intervals. The CAIM algorithm uses class-attribute interdependency as defined in the literature.

The goal of our proposed CAIM algorithm is to find the minimum number of discrete intervals and minimum loss of the class-attribute interdependency. The algorithm uses the class attribute interdependency information as the criterion for the optimal discretization. We introduce several basic definitions for the criterion. For a certain classification task, assume that we have a training data set consisting of M examples and that each example belongs to only one of the S classes. F indicates any of the continuous attributes from the mixed-mode data.

Then, there exists a discretization scheme D on F, which discretizes the continuous domain of attribute F into n discrete intervals bounded by the pairs of number.

$$D: \{[d_0, d_1], [d_1, d_2],..., [d_{n-1}, d_n]\} \quad (12)$$

Where d_0 is the minimal values and d_n is the maximal value of attribute *F*, and the values in (12) are organized in ascending order. These values constitute the boundary set $\{d_0, d_1, d_2, \ldots, d_n\}$ for the discretization *D*. Each value of attribute *F* can be classified into only one of the *n* intervals defined in (12). With the change of discretization *D*, the membership value of each value within a certain interval for attribute *F* may also change. The class variable and the discretization variable of attribute *F* can be treated as two random variables, thus, a two-dimensional frequency matrix (called quanta matrix) can be set up as shown in Table 5.

Class	Interval					Class Total
	$[d_0,\ d_1]$	…	$[d_{r-1},\ d_r]$	…	$[d_{n-1},\ d_n]$	
C_1	q_{11}	…	q_{1r}	…	q_{1n}	M_{1+}
.	.		.		.	.
.	.		.		.	.
C_i	q_{i1}	…	q_{ir}	…	q_{in}	M_{1+}
.	.		.		.	.
.	.		.		.	.
C_s	q_{s1}	…	q_{sr}	…	q_{sn}	M_{1+}
Interval Total	M_{+1}	…	M_{+r}	…	M_{+n}	M

TABLE 5 Quanta Matrix for Attribute *F* and Discretization Scheme *D*

In tables, q_{ir} is the total number of continuous values belonging to the i^{th} class that are within interval $(d_{r-1}, d_r]$. M_{i+} is the total number of object belonging to the i^{th} class and M_{+r} is the total number of continuous values of attribute *F* that are within interval $(d_{r-1}, d_r]$, *for* $i=1, 2, \ldots, S$ and $r = 1, 2, \ldots, n$.

Given the quanta matrix as shown in Table 5, the Class-Attribute Interdependency Maximization (CAIM) criterion that measures the dependency between the class variable *C* and the discretization variable *D* for attribute is defined as:

$$CAIM(C, D|F) = \frac{\sum_{r=1}^{n} \frac{max_r^2}{M_{+r}}}{n} \qquad (13)$$

where, n is the number of intervals, r iterates through all intervals, i.e., $r = 1, 2, ..., n$, max_r is the maximum value among all q_{ir} values (maximum value within the r^{th} column of the quanta matrix), $i = 1, 2, ..., S$, M_{+r} is the total number of continuous values of attribute F that are within the interval $(d_{r-1}, d_r]$. The CAIM criterion is a heuristic measure that quantifies interdependence between classes and the discretized attribute. The criterion is independent of the number of classes and the number of the continuous attributes and has the following properties:

1. The larger the value of CAIM, the higher the correlation between the class labels and the discrete intervals. The bigger the number of values belonging to class C_i within a particular interval, the higher the interdependence between C_i and the interval. If the number of values belonging to C_i within the interval is the largest, and then C_i is called the leading class within the interval. The CAIM criterion accounts for the trend of maximizing the number of values belonging to a leading class within each interval by using max_r. The value of CAIM grows when the values of max_r grow, which relates to the increase of the interdependence between the class labels and the discrete intervals. The highest interdependence between the class labels and the discrete intervals (and, at the same time, the highest value of CAIM) is achieved when all values within a particular interval belong to the same class for all intervals.

   In this case, $max_r = M_{+r}$ and $CAIM = M/n$

2. It takes on real values from the interval $[0, M]$, where M is the number of values of the continuous attribute F.
3. The squared max_r value is divided by the M_{+r} for a reason: To eliminate the negative impact that the values belonging to classes other than the class with the maximum

number of values within an interval have on the discretization scheme. The more such values the bigger the value of M_{+r} will decrease the value of CAIM.

4. Because the criterion favors discretization schemes with a smaller number of intervals, the summed value is divided by the number of intervals *n*.
5. The M_{i+} values from the quanta matrix are not used because they are defined as the total number of objects belonging to the i^{th} class, which does not change with different discretization schemes. The value of the CAIM criterion is calculated with a single pass over the quanta and it maximizes the class-attribute interdependency.

Given: Data consisting of M examples, S classes and continuous attributes F_i. For every F_i

Step 1

Find minimum (d_0) and maximum (d_n) values of F_i.

Form a set of all distinct values of F_i in ascending order, and initialize all possible interval boundaries B with minimum, maximum and all the midpoints values of all the adjacent pairs in the set.

Set the initial discretization scheme as $D: \{[d_0, d_n]\}$, set $GlobalCAIM = 0$.

Step 2

2.1 Initialize $k=1$.

2.2 Tentatively add an inner boundary from B which is not already in D, and add it in D and calculate corresponding CAIM value.

2.3 After all the tentative additions have been tried, accept the one with highest CAIM value.

2.4 If $(CAIM > GlobalCAIM)$ or $k < S$ then update D with the boundary accepted in step 2.3 and set $GlobalCAIM = CAIM$.

2.5 Set $k = k+1$ and go to step 2.2

TABLE 6 CAIM Discretization Algorithm

3.2.3 Class Attribute Contingency Coefficient (CACC) Discretization

This is another type of discretization, which is similar to CAIM and implemented in order to raise the quality of the generated discretization scheme by extending the idea of contingency coefficient which combines with the greedy method. It also generates a better discretization method to obtain an enhancement of accuracy for C5.0. With regards to the execution time of the

discretization, the number of generated rules and the execution time of C5.0, CACC also achieves promising results. By using the same Quanta matrix described in CAIM, researchers use the contingency coefficient as shown in the below equation to measure the strength of dependence between the variables.

$$C = \sqrt{\frac{y}{y + M'}} \tag{14}$$

$$where\ y = M\left[\left(\sum_{i=1}^{s}\sum_{r=1}^{n}\frac{q_{ir}^2}{M_{i+}M_{+r}}\right) - 1\right] \tag{15}$$

M is the total number of instances, *n* is the number of intervals and q_{ir} is the number of samples with class *i (i=1, 2, …, S)* and *r (r= 1, 2, …, n)* in the interval (d_{r-1}, d_r). M_{i+} is the total number of samples with class *i* and M_{+r} is the total number of samples in the interval (d_{r-1}, d_r). From Formula (15), the contingency coefficient takes a distribution of all samples into account by using $[(q_{ir})^2/M_{i+}M_{+r}]$. The contingency coefficient is a very good criterion to measure the interdependence between target class and discretized attributes, which can be considered as two variables. However, we divide *y* by *log (n)* and define the cacc value as:

$$cacc = \sqrt{\frac{y'}{y' + M'}} \tag{16}$$

$$y' = \frac{\left[\left(\sum_{i=1}^{s}\sum_{r=1}^{n}\frac{q_{ir}^2}{M_{i+}M_{+r}}\right) - 1\right]}{log(n)} \tag{17}$$

We divide *y* by *log (n)* for two reasons such as (a) to speed up the discretization process, and (b) a discretization scheme may contain too many intervals and should not suffer from an

overfitting problem. CAIM also took these reasons into account so that in the CAIM criterion, the summed value was divided by the number of intervals *n* and it makes its discretization schemes unreasonable due to the huge influence of the variable *n.* CAIM almost always generates a number of discretization schemes in which the number of intervals is very close to the number of target classes. Hence, *cacc* uses *log (n)* to reduce its influence of *n* as in CAIM. In order to reduce the computation cost, *cacc* also uses the greedy method like *caim* to generate a sub-optimal discretization scheme. In other words, *cacc* not only finds the best division point but records a *Globalcacc* value. If the generated *cacc* value in loop *k+1* is less than the *Globalcacc* obtained in loop *k, cacc* would terminate and output the discretization scheme. To generate a rational discrete result, such a greedy mechanism is ignored if the number of generated intervals is less than the number of target classes. Since the main framework of *cacc* is similar to that of CAIM. The CAIM and CACC algorithm work in a greedy top-down manner. They start with a single interval that covers all possible values of a continuous attribute and divide it iteratively. From all possible division points that are tried (with replacement) in Step 2.2, it chooses the division boundary that gives the highest value of the CAIM and CACC criterion. The algorithm assumes that every discretized attribute needs at least a number of intervals equal to the number of classes because this guarantees the discretized attribute that can improve subsequent classification. The CAIM and CACC algorithms use a trade-off between finding a discretization with the highest possible class-attribute interdependency and a reasonable computational cost. The main advantage of this algorithm is that it finds a small number of discretization intervals, which gives a low computational cost, and at the same time high class-attribute interdependency. The literature survey proposed various discretization methods with the help of greedy algorithm. However, in this thesis, we focus mainly on finding the best decreasing rate using searching

algorithm which is used to find the new stopping criterion in which we want to see the criterion increases or decreases in case of searching algorithms by using a greedy algorithm and also test whether the criterion increases or decreases. In IEM, it is used repetitively to find stop points using a decreasing rate and also for finding an optimal threshold which needs to do cross validation in the training data set.

Given: Data consisting of M examples, S classes and continuous attributes F_i. For every F_i

Step 1

1.1 Find minimum (d_0) and maximum (d_n) values of F_i.

1.2 Form a set of all distinct values of F_i in ascending order, and initialize all possible interval boundaries B with minimum, maximum and all the midpoints values of all the adjacent pairs in the set.

1.3 Set the initial discretization scheme as $D: \{[d_0, d_n]\}$, set $GlobalCACC = 0$ in consideration with *cacc* algorithm.

Step 2

2.1 Initialize $k=1$.

2.2 Tentatively add an inner boundary from B which is not already in D, and add it in D and calculate corresponding *CACC* value.

2.3 After all the tentative additions have been tried, accept the one with highest CACC value.

2.4 If $(CACC > GlobalCACC)$ or $k < S$ then update D with the boundary accepted in step 2.3. Set $GlobalCACC = CACC$, else terminate.

2.5 Set $k = k+1$ and go to step 2.2

Output: Discretization Scheme D

TABLE 7 CACC Discretization Algoritm

CHAPTER IV

GLOBAL ENTROPY BASED GREEDY ALGORITHM

In classification, decision tree and Bayesian network classifiers require discretization during preprocessing; IEM has been consistently used even though CAIM and CACC were proposed with better experimental results. The reason to conventionally use IEM is probably its own theoretical background and competitive performance. Our preliminary experiment also shows that CAIM and CACC were not clearly superior to IEM as shown in the Table 8.

In this chapter, we propose a global Entropy-based greedy algorithm. Global Entropy (GE) was originally defined in [WMP2005], to search for cut-points with GE as a fitness function. In the proposed method, instead of recursively splitting intervals in IEM, searching cut-points are iteratively decided by measuring GEs for all possible cut-points and selecting a cut-point that has minimum entropy. While the entropy in IEM is for only two intervals, Global Entropy (GE), is measured with multiple intervals. This is similar with CAIM and CACC where *caim* and *cacc* values are iteratively measured in their algorithms. GE is defined as follows.

$$\sum_{i=1}^{n} [\ (\ |p_i|(log_2|p_i|) - \sum_{j=1}^{m} |p_{im}|(log_2|p_{im}|\ )] \left(\frac{|P_i|}{q}\right) \qquad (18)$$

From this equation (18), n is the number of partitions, p_i the set of instances in the i^{th} partition, m is the number of unique class labels in $p_{i,}$ p_{im} the set of instances of partition p_i that have the m^{th} class label, and q the number of instances in the entire input file.

In the CAIM and CACC algorithms, intervals are continually split as long as *caim* and *cacc* are greater than the values in previous iterations or the number of intervals is less than the number of classes, *S*. It means that the number of intervals can be greater than *S* only when *caim* and *cacc* is increasing after the number of intervals is *S*. However, the preliminary experiment (Tables 10 and 11), CAIM and CACC algorithms return *S* as the final number of intervals in most data sets, and also GE always decrease when the number of intervals increase. Thus, only *S* is considered as a stopping criterion in the proposed algorithm but not the GE value. More precisely, intervals are iteratively split as long as the number of intervals is less than *S*, which is the number of classes. The GE-based greedy algorithm is defined in Table 8. Although, it seems that *S* is a good number as a threshold to stop splitting, it is worth exploring the performance could be influenced by the threshold. To this end, first we have observed how GE changes with a different number of intervals. In the experimental results, the change rate of GE had a similar pattern in most datasets as mentioned above. More precisely, GE was exponentially decreasing and converged when the number of interval was increasing as shown in figures 5 and 6. Therefore, it is better when the threshold is determined by the decreasing rate of GE rather than a constant number such as *S*.

Given: Data consisting of M examples, S classes and continuous attributes F_i. For every F_i

Step 1

1.1 Find minimum (d_0) and maximum (d_n) values of F_i.

1.2 Form a set of all distinct values of F_i in ascending order, and initialize all possible interval boundaries B with minimum, maximum and all the midpoints values of all the adjacent pairs in the set.

1.3 Set the initial discretization scheme as $D: \{[d_0, d_n]\}$.

Step 2

2.1 Initialize $k=1$.

2.2 Tentatively add an inner boundary from B which is not already in D, and add it in D and calculate corresponding GE value.

2.3 After all the tentative additions have been tried, accept the one with *lowest GE* value (GE_i).

2.4 If $k<S$ then update D with the boundary accepted in step 2.3.

2.5 Set $k = k+1$ and go to step 2.2

Output: Discretization Scheme D

Table 8: Global Entropy Based Algorithm

The decreasing rate of GE is simply defined as $1\text{-}GE_{i-1}/GE_i$. If the decreasing rate of GE is used as a threshold, Step 2.4 in the algorithm (Table 8) should be changed as follows.

2.4 If $1\text{-}GE_{i-1}/GE_i > threshold$ then update D with the boundary accepted in step 2.3.

CHAPTER V

AN EXPERIMENTAL ANALYSIS

In this Experimental Analysis, the results of the searching algorithms such as IEM, CAIM, and CACC with five other leading Discretization algorithms are analyzed on thirteen well-known continuous and mixed-mode datasets.

5.1 The Experimental Setup

The thirteen datasets used to test the searching algorithms with better performance and accuracies are:

1. Breast Cancer Wisconsin (Original) Dataset (bre).
2. Bupa Liver Disorders Dataset (bup).
3. Glass Identification Dataset (gla).
4. Statlog Project Heart Disease Dataset (hea).
5. John Hopkins University Ionosphere Dataset (ion).
6. Iris Plants Dataset (iri).
7. Optical Recognition of Hand-Written Digits (opt).
8. Page Blocks Classification Dataset (pag).
9. Pen-Based Recognition of Hand-Written Digits (pen).
10. Pima Indians Diabetes Dataset (pid).
11. Statlog Project Satellite Dataset (sat).
12. Thyroid Disease Dataset (thy).

13. Waveform Dataset (wav).

All of these thirteen datasets have been obtained from Machine Learning Repository [LCM2013]. Tests were performed for the five discretization algorithms with two classifiers. The five Discretization algorithms used in our experiment are:

Unsupervised: Equal Width and Equal Frequency.

Supervised: IEM, CAIM and CACC

Classifiers: NBC and C5.0.

Datasets	bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav
# instances	699	345	214	270	351	150	5620	5473	10992	768	6435	7200	5000
# attributes	9	6	9	13	34	4	62	10	16	8	36	21	21
# continuous attributes	9	6	9	5	32	4	61	10	16	8	36	6	21
# classes	2	2	6	2	2	3	10	5	10	2	6	3	3

TABLE 9: Datasets

5.2 Analysis of Classifications using Discretized Datasets

In this analysis, the accuracy and number of intervals were compared for the five discretization algorithms. Since C5.0 and NBC can generate data models from continuous attributes by comparing its performance while it generates rules from raw data against the results achieved using discretized data using five algorithms. We investigate the cut-points and also discretize intervals by performing five-fold cross-validation on training data for the three searching algorithms to find an optimal threshold.

Datasets		bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav	avg
EW	acc	96.8	63.3	62.4	81.4	87.3	87.3	89.4	93.0	88.1	74.9	80.1	93.7	80.6	82.9
	sd	2.1	7.8	10.4	7.4	5.9	8.1	1.2	1.0	1.0	4.7	1.5	0.9	1.7	4.1
	rank	2	1	4	4	5	5	5	5	1	1	5	5	2	3.5
EF	acc	97.6	62.6	66.5	82.2	92.8	88.5	91.6	93.0	87.6	74.9	80.4	98.5	81.0	84.4
	sd	1.8	8.3	10.0	7.3	4.3	8.4	1.1	1.0	1.0	4.8	1.5	0.5	1.7	4.0
	rank	1	3	1	3	1	4	1	4	2	2	4	3	1	2.3
IEM	acc	96.7	62.8	65.1	82.4	90.0	92.2	90.0	93.7	86.0	74.2	80.7	98.6	72.9	83.5
	sd	2.1	8.4	10.5	7.1	5.0	7.2	1.3	1.0	1.0	4.9	1.5	0.5	2.1	4.0
	rank	5	2	3	2	3	3	4	1	5	3	3	1	5	3.1
CAIM	acc	96.7	62.4	65.7	82.4	87.3	93.4	90.5	93.6	86.9	72.7	80.8	98.2	79.9	83.9
	sd	2.2	8.5	10.3	7.2	6.5	6.4	1.2	1.1	1.1	5.0	1.6	0.8	1.8	4.1
	rank	4	4	2	1	4	1	2	3	4	4	2	4	3	2.9
CACC	acc	96.8	62.3	42.6	81.1	90.2	93.4	90.5	93.7	86.9	72.5	81.2	98.5	78.9	82.2
	sd	2.2	8.3	10.4	7.8	4.9	6.5	1.2	1.2	1.0	5.1	1.5	0.6	1.9	4.0
	rank	3	5	5	5	2	2	3	2	3	5	1	2	4	3.2

TABLE 10: NBC ACCURACIES

Table 10 and 11 describes the best accuracies among five discretization algorithms with two classifiers separately. NBC shows the average best accuracies when compared with C5.0 in bre, bup, thy and also C5.0 has better accuracies with datasets hea, iri, and page. IEM performs accuracies well with NBC when analyzed with C5.0, even though it is more competitive studied with the other supervised discretization algorithms CAIM and CACC, which were developed after IEM. Overall, C5.0 shows the highest average accuracies when compared with NBC.

Datasets		bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav	avg
EW	acc	94.5	60.9	63.4	79.8	87.3	91.8	88.1	93.9	95.7	73.8	86.3	94.6	75.3	83.5
	sd	2.6	7.7	10.2	7.5	5.6	6.9	1.4	1.0	0.6	4.8	1.3	0.8	1.9	4.0
	rank	5	3	4	2	5	5	4	5	1	2	1	5	3	3.5
EF	acc	94.6	60.1	67.8	80.1	88.1	93.0	88.9	96.0	95.7	73.7	85.5	99.0	75.1	84.4
	sd	2.6	8.9	10.1	7.5	5.4	6.8	1.3	0.8	0.6	5.0	1.4	0.4	1.9	4.1
	rank	4	5	2	1	4	4	1	4	2	3	4	2	4	3.1
IEM	acc	95.4	60.1	66.8	78.9	91.6	94.4	87.7	96.4	90.7	74.2	85.0	99.2	69.2	83.8
	sd	2.5	8.9	10.1	7.6	4.9	6.1	1.4	0.8	1.0	4.8	1.4	0.3	2.2	4.0
	rank	3	4	3	3	1	1	5	1	5	1	5	1	5	2.9
CAIM	acc	95.4	62.0	68.7	78.5	89.3	93.9	88.7	96.2	95.3	73.1	85.8	98.7	77.2	84.8
	sd	2.6	8.7	9.8	7.4	5.2	5.9	1.4	0.8	0.7	4.8	1.4	0.7	1.9	4.0
	rank	1	1	1	4	2	3	2	2	3	5	2	4	1	2.4
CACC	acc	95.4	61.1	51.3	78.2	88.9	94.0	88.6	96.2	94.7	73.5	85.7	98.9	76.0	83.3
	sd	2.6	8.8	11.5	7.6	5.0	5.8	1.4	0.8	0.8	5.0	1.4	0.5	1.9	4.1
	rank	2	2	5	5	3	2	3	3	4	4	3	3	2	3.2

TABLE 11: C5.0 ACCURACIES

Experimental results in Tables 12 and 13 show the number of intervals and classes (S) of three supervised discretization algorithms. However, from these two tables, CAIM and CACC return S as the final number of intervals in most datasets, and GE always decrease when the number of intervals increases and hence, S can be considered as a stopping criterion and the intervals are recursively split as long as the number of intervals is less than the number of classes S. Although S can be used as a good number for the threshold but there is a need to explore the performance and how it can be influenced by the threshold. Therefore, we have observed how GE changes with a different number of intervals when compared with both classifiers. Hence, C5.0 generated the average highest number of intervals when compared with NBC.

Datasets		bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav
# of classes		2	2	6	2	2	3	10	5	10	2	6	3	3
IEM	# int	3.0	7.1	4.1	4.8	4.4	4.7	11.8	26.4	75.9	8.5	32.1	143.8	16.1
CAIM	# int	2.0	2.0	6.0	2.0	2.0	3.0	9.5	5.0	10.0	2.0	6.0	3.0	3.0
CACC	# int	2.0	4.3	22.6	4.4	2.9	3.1	10.0	5.4	22.6	2.8	8.3	28.1	3.4

Table 12: NBC Intervals

Datasets		bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav
# of classes		2	2	6	2	2	3	10	5	10	2	6	3	3
IEM	# int	2.4	6.1	3.3	1.7	4.0	3.7	11.0	24.0	71.5	7.6	31.3	137.3	4.4
CAIM	# int	2.0	2.0	6.0	2.0	2.0	3.0	9.5	5.0	10.0	2.0	6.0	3.0	3.0
CACC	# int	2.0	4.3	22.6	4.4	2.9	3.1	10.0	5.4	22.6	2.8	8.3	28.1	3.4

Table 13: C5.0 Intervals

In tables 14 and 15, we compare the IEM discretization algorithm with the proposed Global Entropy based approach such that the stopping criterion of MDLP is less than the proposed approach when we considered number of intervals is equal to the number of classes by using Global Entropy Based Greedy approach. From this observation, we measured the decreasing rate of all the datasets to show that the GE always decreases and converges.

Hence, the results proved that the GE approach is more effective and shows the highest average accuracies when used with both classifiers and also both Classifiers show that a greater number of datasets have an improvement in accuracy when tested with GE approach. Hence, the proposed GE approach is more robust and shows the best performance.

Datasets			bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav	avg
IEM	MDLP	acc	96.7	62.8	65.1	82.4	90.0	92.2	90.0	93.7	86.0	74.2	80.7	98.6	72.9	83.0
		sd	2.1	8.4	10.5	7.1	5.0	7.2	1.3	1.0	1.0	4.9	1.5	0.5	2.1	4.5
		# int	3.0	7.1	4.1	4.8	4.4	4.7	11.8	26.4	75.9	8.5	32.1	16.1	143.8	
GE greedy	# intervals == # classes	acc	97.2	65.7	65.1	82.6	88.4	94.0	90.4	92.5	87.4	73.7	81.3	98.3	80.4	84.4
		sd	2.0	8.1	10.4	7.1	5.5	6.1	1.2	1.3	1.0	4.9	1.4	0.5	1.7	3.9
		# int	3.0	3.0	7.0	3.0	3.0	4.0	10.3	6.0	11.0	3.0	7.0	4.0	4.0	

Table 14: NBC Results for GE Approach

Datasets			bre	bup	gla	hea	ion	iri	opt	pag	pen	pid	sat	thy	wav	avg
IEM	MDLP	acc	95.4	60.1	66.8	78.9	91.6	94.4	87.7	96.4	90.7	74.2	85.0	99.2	69.2	83.8
		sd	2.5	8.9	10.1	7.6	4.9	6.1	1.4	0.8	1.0	4.8	1.4	0.3	2.2	4.0
		# int	2.4	6.1	3.3	1.7	4.0	3.7	11.0	24.0	71.5	7.6	31.3	4.4	137.3	
GE greedy	# intervals == # classes	acc	95.3	64.4	68.4	78.6	88.3	93.7	88.8	96.5	95.7	72.0	85.8	98.5	77.4	84.9
		sd	2.6	7.7	9.8	7.6	5.5	6.3	1.3	0.8	0.7	4.9	1.4	0.5	1.9	3.9
		# int	2.0	2.0	6.0	2.0	2.0	3.0	9.5	5.0	10.0	2.0	6.0	3.0	3.0	

Table 15: C5.0 Results for GE Approach

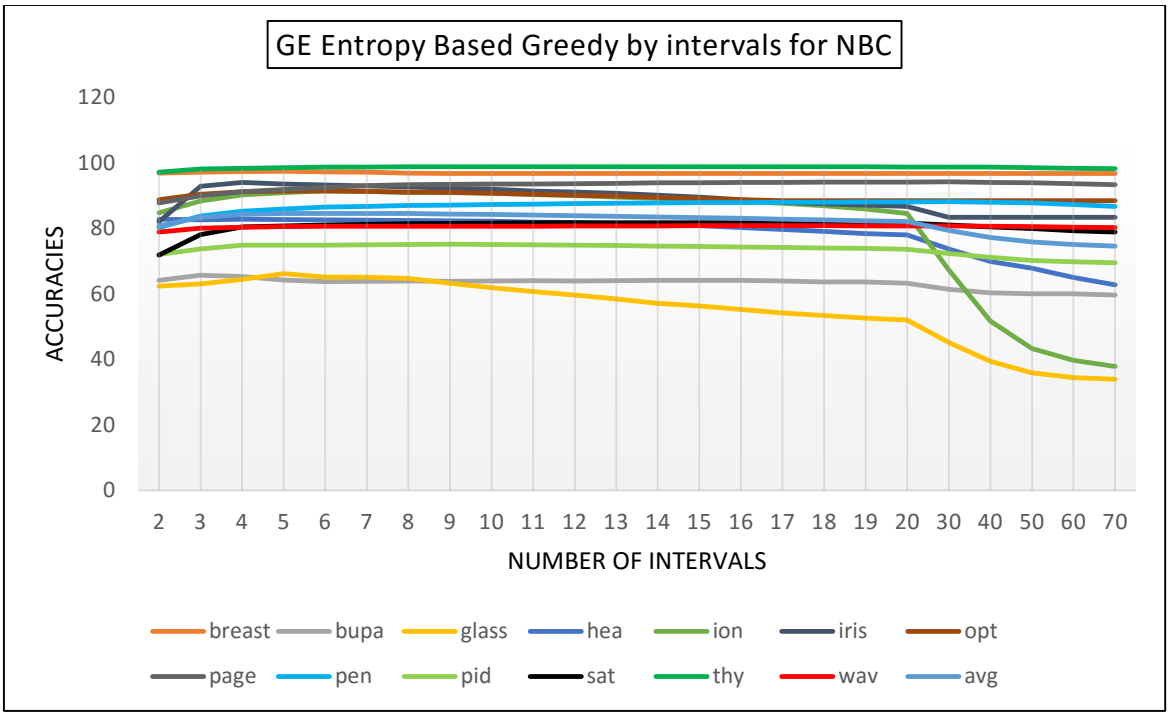

Figure 3: GE Entropy Based Greedy by intervals for NBC

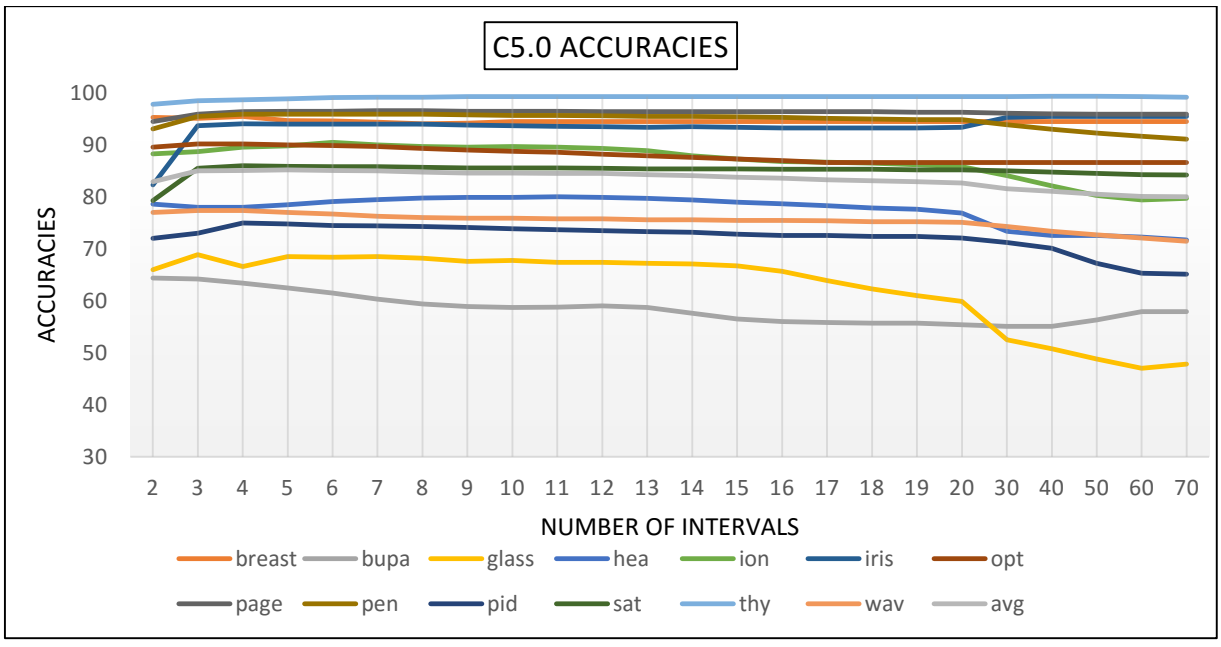

Figure 4: GE Entropy Based Greedy by intervals for C5.0.

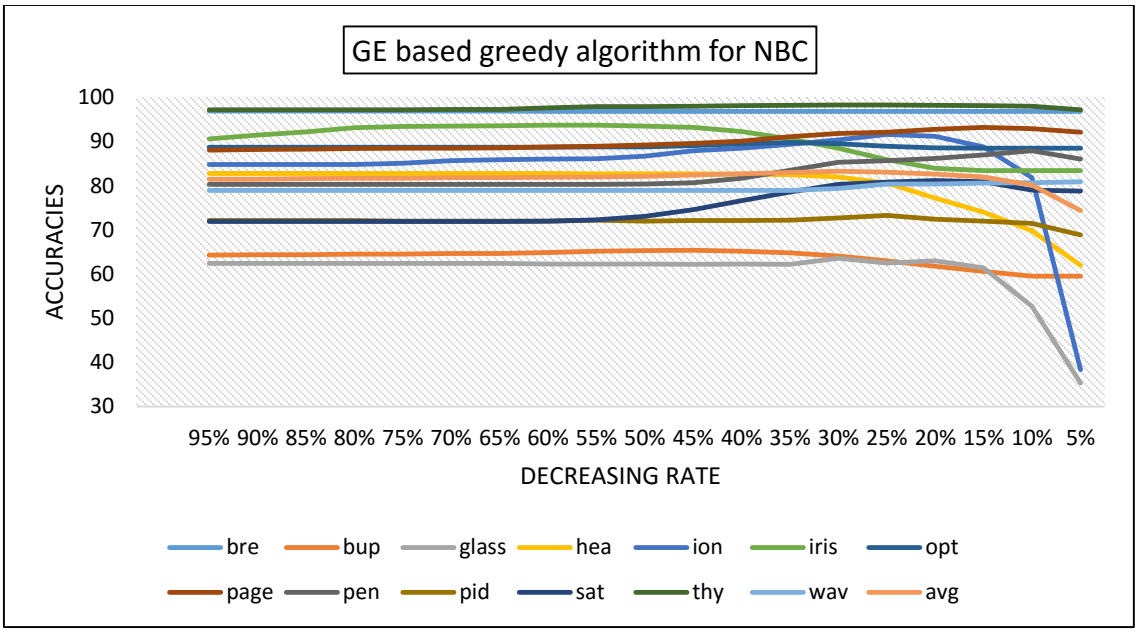

Figure 5: GE based greedy algorithm for NBC

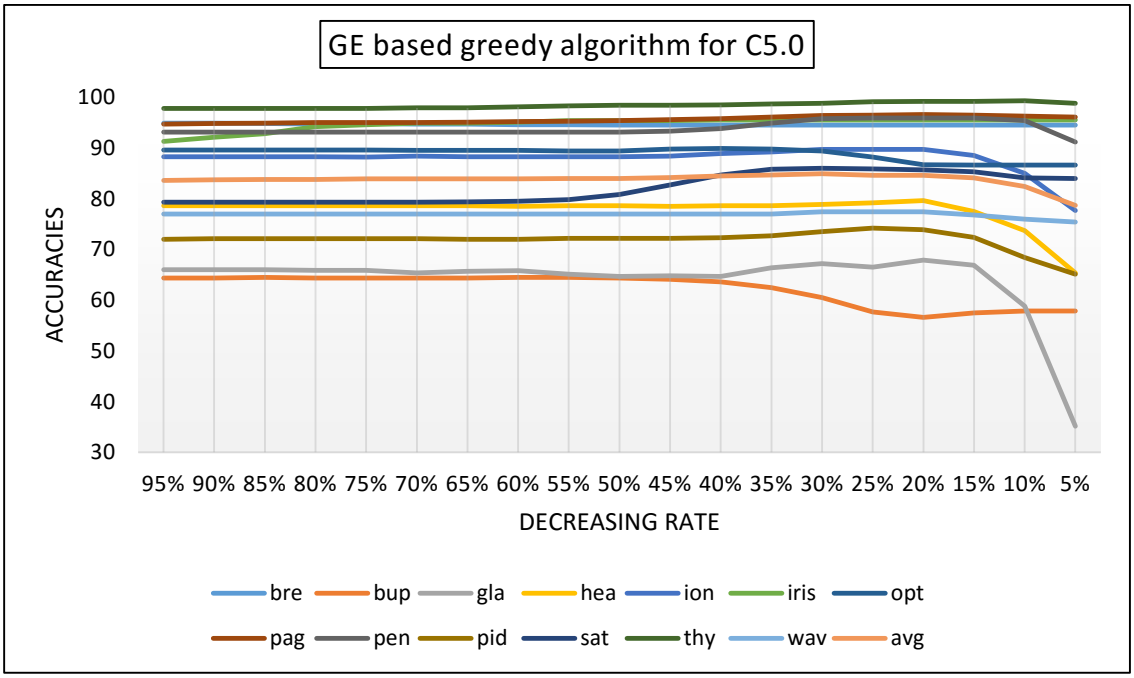

Figure 6: GE Based greedy algorithm for C5.0

Figure 3 and Figure 4 investigates the potential to find the thresholds as a stopping criterion but not considered *S* as a good stopping criterion to stop splitting when tested with different number of intervals. Hence, we can observe from this graph that the GE constantly

changes the accuracy rate with different datasets. In case of NBC, glass dataset shows the best accuracy by but bupa and glass have better accuracy in C5.0 by using number of intervals as a threshold. More number of datasets such as glass, bupa, hea and opt have better accuracy with NBC. Therefore, by comparing with both datasets NBC shows the better results rather than C5.0 but both performs well with different number of intervals.

Figure 5 and 6 shows that GE have the best change rate compare to other datasets and it was exponentially decreasing and converged when the number of interval is increasing. Therefore, it can be understood that the threshold can be determined by the decreasing rate of GE rather than used with *S.* NBC and C5.0 shows the best cut-point with 25 percent when considered decreasing rate as a threshold when compared with all the other datasets. The tests result shows that GE based approach is mostly effective.

CHAPTER VI

CONCLUSION

In conclusion, many discretization algorithms have been proposed based on IEM, which are more competitive due to their efficiency and good performance in classification stage while CAIM and CACC came after IEM. The main contribution of this paper is to propose a new threshold as a stopping criterion. In this thesis, we proposed a global entropy-based greedy algorithm, with GE as a fitness function. In the proposed method, searching cut-points are recursively decided by measuring GEs for all possible cut-points that has minimum entropy. In short, GE is measured with multiple intervals when compared with IEM, which has only two intervals. In contrast, intervals are continually split as long as *caim* and *cacc* are greater than the values and the number of intervals is greater than the number of classes, *S.* Although it seems that *S* is a good number as a threshold to stop splitting, the performance could be explored and influenced by the threshold. However, from the experimental analysis, GE was exponentially decreasing and converged when the number of intervals was increasing. Therefore, it is better when the threshold is determined by the decreasing rate of GE rather than a constant number, such as *S*.

REFERENCES

[DKS1995] Dougherty, J., Kohavi, R. and Sahami, M., 1995, July. *Supervised and unsupervised discretization of continuous features. In Machine learning:* proceedings of the twelfth international conference (Vol.12,pp.194-202).

[CMB1996] Chmielewski, M.R. and Grzymala-Busse, J.W., 1996. *Global discretization of continuous attributes as preprocessing for machine learning. International* journal of approximate reasoning, 15(4), pp.319-331.

[DNA2007] D.J. Newman A. Asuncion. UCI machine learning repository, 2007. *H. Liu, F. Hussain, C. L. Tan, and M. Dash. Discretization: An Enabling Technique. Data Mining and Knowledge Discovery, 6(4):393–423, October 2002.*

[IKB1993] Irani, Keki B. *"Multi-interval discretization of continuous-valued attributes for Classification learning." (1993).*

[FUI1992] Fayyad, Usama M., and Keki B. Irani. *"On the handling of continuous-valued attributes in decision tree generation." Machine learning 8.1 (1992): 87-102.*

[KLJ2004] Kurgan, Lukasz A., and Krzysztof J. Cios. *"CAIM discretization algorithm". Knowledge and Data Engineering, IEEE Transactions on 16.2 (2004) 145-153.*

[KSD2006] Kotsiantis, Sotiris, and Dimitris Kanellopoulos. *"Discretization techniques: A recent survey". GESTS International Transactions on Computer Science and Engineering 32.1 (2006): 47-58.*

[TCL2008] Tsai, Cheng-Jung, Chien-I. Lee, and Wei-Pang Yang. *"A discretization algorithm based on class-attribute contingency coefficient." Information Sciences 178.3 (2008): 714-731.*

[VSM2012] Vora, Shivani V., and R. Mehta. *"MCAIM: Modified CAIM Discretization Algorithm for Classification " International Journal of Applied Information Systems 3: 3*

[JQK1993] J. R. Quinlan. C4.5: *Programs for Machine Learning. Morgan Kaufmann, 1993.*

[YGW2002] Y. Yang and G.I.Webb. *A comparative study of discretization methods for Naïve Bayes Classifiers. In Proceedings of PKAW2002, The 2002 Pacific Rim Knowledge Acquisition Workshop pages 159-173, Tokyo, 2002.*

[YWB2003] Y. Yang and G.I. Webb. *On Why Discretization Works for Naive-Bayes Classifiers in AI 2003: Advances in Artificial Intelligence, pages 440–452. Springer Berlin, 2003.174.*

[IHE2005] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufman, 2 editions, 2005.*

[SKW2012] Sriwanna, Kittakorn, Kamthorn Puntumapon, and Kitsana Waiyamai.*"An enhanced class-attribute interdependence maximization discretization algorithm" Advanced Data Mining and Applications. Springer Berlin Heidelberg, 2012. 465-476.*

[WXN2015] Wu, Xun. *A Global Discretization Approach to Handle Numerical Attributes as Preprocessing. Diss. University of Kansas, 2015.*

[RLM2005] Rokach, L., and Maimon, O. *(2005). Decision Trees. In The Data Mining and Knowledge Discovery Handbook, Springer, pp. 165-192.*

[QJR1996] Quinlan, J. R. (1996). *"Improved use of continuous attributes in C4.5". In Journal of Artificial Intelligence Research, 4, 77-90.*

[CTJ1991] Catlett, J. (1991*). "On changing continuous attributes into ordered discrete Attributes". In Proc. Fifth European Working Session on Learning, pp.164-17 177.*

[CWL2011] Chao, Wei-Lun. *"Machine Learning Tutorial."*

[GSO2013] Garcia, Sergio, et al. *"A survey of discretization techniques: Taxonomy and Empirical analysis in supervised learning ". Knowledge and Data Engineering, IEEE Transactions on 25.4(2013): 734-750.*

[CJA1995] Ching, John Y., Andrew KC Wong, and Keith CC Chan. *"Class-dependent Discretization for inductive learning from continuous and mixed-mode data." Pattern Analysis and Machine Intelligence, IEEE Transactions on 17.7 (1995): 641-651.*

[LCI2007] Lee, Chien-I., et al. *"A top-down and greedy method for discretization of Continuous attributes." Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007 Fourth International Conference on. Vol. 1. IEEE, 2007.*

[DRP2011] Dash, Rajashree, Rajib Lochan Paramguru, and Rasmita Dash. *"Comparative Analysis of Supervised and Unsupervised Discretization techniques". International Journal of Advances in Science and Technology2.3 (2011): 29-37.*

[AGS2002] Agre, Gennady and Stanmir Peev. *"On Supervised and Unsupervised Discretization." Cybernetics and information technologies 2.2 (2002): 43-57.*

[MFR2005] Muhlenbach, Fabrice, and Ricco Rakotomalala. *"Discretization of continuous Attributes ".Encyclopedia of Data Warehousing and Mining 1 (2005): 397-402.*

[LWY2011] Loh, Wei-Yin. *"Classification and regression trees". Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1.1 (2011): 14-23.*

[PTK2009] Pongaksorn, Prachya, Thanawin Rakthanmanon, and Kitsana Waiyamai. *"DCR: Discretization using class information to reduce number of intervals". Data Analysis and Knowledge Discovery Laboratory (DAKDL), P. Lenca and S. Lallich (Eds): QrMIE/PAKDD (2009).*

[BVL2013] B. Hemada, K.S. Vijaya Lakshmi.*" Discretization Technique using Maximum Frequent values and Entropy criterion". International Journal of Advanced Research in Computer Science and Software Engineering 3(11), November-(2013), pp. 231 -237.*

[ZWI2010] Zhu, Wenzhi, et al. *"A discretization algorithm based on information distance criterion and ant colony optimization algorithm for knowledge extracting on industrial database "Mechatronics and Automation (ICMA), 2010 International Conference on. IEEE, 2010.*

[CJS2012] Carneros, Juan Jesús, et al. *"Data discretization using the extreme learning Machine neural network". Neural Information Processing. Springer Berlin Heidelberg, 2012.*

[KRM1996] Kohavi, Ron, and Mehran Sahami. *" Error-Based and Entropy-Based Discretization of Continuous Features". KDD. 1996.*

[SVL2011] Susila, S., and S. Vadivel. *"Web Service Selection based on QoS Attributes using Entropy Discretization Method". International Journal of Computer Applications 30.2 (2011).*

[FDM2011] Farid, Dewan Md. *"Improve the quality of supervised discretization of continuous valued attributes in data mining." Computer and Information Technology (ICCIT), 2011 14th International Conference on. IEEE, 2011.*

[JRC2009] Jin, Ruoming, Yuri Breitbart, and Chibuike Muoh. *"Data discretization unification". Knowledge and Information Systems 19.1(2009): 1-29.*

[BRD2004] Butterworth, Richard, et al. *"A greedy algorithm for supervised discretization" . Journal of biomedical informatics 37.4 (2004): 285-292.*

[JZO2011] Juan, Zhao, et al. *"Discretization Based on Positive Domain and Information Entropy." Computational Intelligence and Security (CIS), 2011 Seventh International Conference on. IEEE, 2011.*

[KYF2008] Kayah, F. *"Discretizing Continuous Features for Naive Bayes and C4. 5 Classifiers". University of Maryland publications: College Park, MD, USA (2008)*

[LJL2008] Lustgarten, Jonathan L., et al. *"Improving classification performance with discretization on biomedical datasets." AMIA. 2008.*

[IMV2003] Ismail, Michael, and Victor Ciesielski. *"An Empirical Investigation of the Impact of Discretization on Common Data Distributions." HIS. 2003.*

[KAR1999] Kaufman, Kenneth A., and Ryszard S. Michalski. *"Learning from inconsistent and noisy data: the AQ18 approach." Foundations of Intelligent Systems. Springer Berlin Heidelberg, 1999. 411-419.*

[HTA2004] Hidekazu, Tokunaga, et al. *"Estimating sentence types in computer related new product bulletins using a decision tree." Information Sciences 168.1 (2004):* 185-200.

[ZSJ2006] Zadrożny, Sławomir, and Janusz Kacprzyk. *"Computing with words for text processing: An approach to the text categorization." Information Sciences176.4 (2006): 415-437.*

[GDM2006] Gómez, Daniel, Javier Montero, and Javier Yáñez. *"A coloring fuzzy graph approach for image classification." Information Sciences 176.24 (2006): 3645*

[KIR1995] Kononenko, Igor. *"On biases in estimating multi-valued attributes." Ijcai. Vol. 95. 1995.*

[WAD1987] Wong, Andrew KC, and David KY Chiu. *"Synthesizing statistical knowledge from incomplete mixed-mode data." Pattern Analysis and Machine Intelligence, IEEE Transactions on 6 (1987): 796-805.*

[CWG1991] D. Chiu, A. Wong, B. Cheung, *Information discovery through hierarchical maximum entropy discretization and synthesis, in G.Piatetsky-Shapiro, W.J. Frawley (Eds.), Knowledge Discovery in Databases, AAAI Press, 1991, pp. 125–140.*

[KSV2007] K.G. Srinivasa, K.R. Venugopal, L.M. Patnaik, *A self-adaptive migration model genetic algorithm for data mining applications, Information Sciences 177 (20) (2007) 4295–4313.*

[RRY2006] R.R. Yager, *An extension of the naive Bayesian classifier, Information Sciences 176 (5) (2006) 577–588.*

[CFG2007] G. Cardoso, F. Gomide, *Newspaper demand prediction and replacement model based on fuzzy clustering and rules, Information Sciences 177 (21) (2007) 4799–4809.*

[CRM2007] S. Cohen, L. Rokach, O. Maimon, Decision-tree instance-space decomposition with grouped gain-ratio, Information Sciences 177 (17) (2007) 3592–3612.

[RQN2005] R. Quinlan, Data Mining Tools, 2005. *<http://www.rulequest.com/see5-info.html>*.

[RJS2006] Ranjit Abraham, Jay B.Simha, Iyengar S.S., "*A comparative analysis of discretization methods for Medical datamining with Naïve Bayesian classifiers",9th International Conference for Information Technology (ICIT2006), pp. 235-236, 2006.*

[LMN2011] Li, Min, et al. *"An effective discretization based on class-attribute coherence maximization." Pattern Recognition Letters 32.15 (2011): 1962-1973.*

[BAS2014] Baka, Abdulloh, Wiphada Wettayaprasit, and Sirirut Vanichayobon. *"A novel discretization technique using Class Attribute Interval Average." Digital Information and Communication Technology and it's Applications (DICTAP), 2014 Fourth International Conference on. IEEE, 2014.*

[JYT2010] Ge, Jiaqi, Yuni Xia, and Yicheng Tu. *"A discretization algorithm for uncertain data." Database and Expert Systems Applications. Springer Berlin Heidelberg, 2010.*

[CAO2016] Cano, Alberto, et al. *"ur-CAIM: improved CAIM discretization for unbalanced and balanced data." Soft Computing 20.1 (2016): 173-188.*

[KLJ2003] Kurgan, Lukasz A., and Krzysztof J. Cios. *"Fast Class-Attribute Interdependence Maximization (CAIM) Discretization Algorithm." ICMLA. 2003.*

[RQR2011] Information on See5/C5.0 - RuleQuest Research Data Mining Tools,2011. [Online]. Available: http://www.rulequest.com/see5-info.html.

[WMP2005] Waldron, Mike, and P. Manuel. "Genetic algorithms as a data discretization method." *Proceeding of Midwest Instruction and Computing Symposium*"2005.

BIOGRAPHICAL SKETCH

Sai Jyothsna Jonnalagadda was born in a city called Sriharikota (ISRO) in southern part of India on July 9, 1990, to her parents Vema Narayanamma Jonnalagadda and Ravichandraiah Jonnalagadda. She obtained her high school degree from Space Central School, Sriharikota in 2007. She graduated from Jawaharlal Nehru Technological University, Anantapur in 2011 with Bachelor's Degree in Information Technology. She finally decided to pursue her Master Degree from University of Texas Rio Grande Valley where she received her Master of Science in Computer Science in May 2016.

Her permanent mailing address is:

Sai Jyothsna Jonnalagadda

D/O Ravichandraiah Jonnalagadda

Andhra Pradesh

India