DATA PREPROCESSING MANAGEMENT SYSTEM

A Thesis

Presented to

The Graduate Faculty of The University of Akron

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

Kalyani Anumalla

December, 2007
ABSTRACT

A large volume of data is generated in the world everyday continuously and it is always multiplying. This data is a vast collection of facts and trends in any field. The task of extracting useful information from the data is challenging. This data is of little use if the experts cannot analyze the facts and identify the trends and patterns. Extracting useful information from raw data is a challenging task. Dealing with the raw data in itself is a tedious task.

This study presents a system called Data Preprocessing Management System, for preprocessing the data and managing the preprocessed data which is stored in databases. It is implemented using SQL, a language for developing database applications. Preprocessing of raw data includes removal of missing values and discretization. For removal of missing values techniques used are replace missing values, delete missing values and customization. Replace missing values technique replaces all the missing values of the attribute with its mean, delete missing values technique deletes all the missing values, and customization replaces missing values with a user defined value. Discretization techniques implemented are Equal Width Discretization, Equal Frequency Discretization and customized discretization. These techniques are used to discretize the preprocessed data. This research mainly focuses on statistics of the data, preprocessing of raw data, discretization of preprocessed data and management of preprocessed data.
Management is a new concept which is introduced in my research and is used to manage the tables and the data that are created. In this system all the tables that are created at each stage will be stored in different tables. At each stage users can easily compare the processed data with raw data, using the statistics. Users can view and download all the raw data and preprocessed data at a client side.
DEDICATION

Dedicated to my parents

Mr. and Mrs. A. Ramreddy.
ACKNOWLEDGEMENTS

First of all, I would like to thank Dr. Chien-Chung Chan for his guidance and support throughout my thesis. His feedback helped me strengthen my research skills and contributed greatly to this thesis. I want to thank my thesis committee members, Dr. Xuan-Hien Dang and Dr. Zhong-Hui Duan, for their guidance and encouragement.

I thank my parents for their love, support and encouragement in my life. I also want to thank my friend Aditya Anne for his invaluable support. He was always there for me when I needed him.
# TABLE OF CONTENTS

| LIST OF TABLES | ix |
| LIST OF FIGURES | x |

## CHAPTER

### I. INTRODUCTION

1.1 What is data preprocessing .........................................................2

1.2 Why preprocess the data.................................................................2

1.3 What is preprocessing management system. ....................................3

1.4 Statement of the problem.................................................................4

1.5 Objective .........................................................................................4

1.6 Issues considered .............................................................................5

1.7 Motivations and contributions.......................................................5

1.8 Organization of the thesis...............................................................6

### II. CONCEPTS AND METHODOLOGY

2.1 KDD process ....................................................................................7

2.2 Characteristics and nature of KDD application ...............................10

2.3 Preprocessing ..................................................................................11

2.3.1 Preprocessing tasks .....................................................................11
2.3.2 Preprocess missing values.................................................................12
2.3.3 Replace missing values.................................................................13
2.3.4 Delete missing values.................................................................14
2.4 Discretization.........................................................................................15
  2.4.1 Equal Width discretization.........................................................16
  2.4.2 Equal Frequency discretization.................................................19
  2.4.3 Entropy Based discretization.......................................................21

III. DESIGN OF PREPROCESSING MANAGEMENT SYSTEM..................15
  3.1 Conceptual model.................................................................24
  3.2 Overview of the stages..................................................................27
  3.3 Management.................................................................................29

IV. IMPLEMENTATION...........................................................................30
  4.1 Advantages of SQL versus WEKA.................................................30
  4.2 Connection establishment............................................................31
  4.3 Uploading an input file.................................................................31
  4.4 Displaying statistics of input table...............................................33
  4.5 Preprocessing of input table..........................................................36
  4.6 Discretization of preprocessed data.................................................38

V. CONCLUSION AND FUTURE WORK...............................................46
REFERENCES......................................................................................47
APPENDIX..........................................................................................49
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4.3 Input table for Entropy Based discretization</td>
<td>21</td>
</tr>
<tr>
<td>A.1 Input table</td>
<td>50</td>
</tr>
<tr>
<td>A.1.1 Statistics of input file</td>
<td>52</td>
</tr>
<tr>
<td>A.1.2 Statistics of replace missing values table</td>
<td>53</td>
</tr>
<tr>
<td>A.1.3 Statistics of delete missing values table</td>
<td>54</td>
</tr>
<tr>
<td>A.2 Statistics of customized preprocess missing values table</td>
<td>55</td>
</tr>
<tr>
<td>A.3 Distribution of Equal Width discretization</td>
<td>57</td>
</tr>
<tr>
<td>A.3.1 Equal Width discretization table</td>
<td>58</td>
</tr>
<tr>
<td>A.4 Distribution of Equal Frequency discretization</td>
<td>59</td>
</tr>
<tr>
<td>A.4.1 Equal Frequency table</td>
<td>60</td>
</tr>
<tr>
<td>A.5 Distribution of customized discretization</td>
<td>61</td>
</tr>
<tr>
<td>A.5.1 Customized discretization table</td>
<td>62</td>
</tr>
<tr>
<td>A.6 Data management</td>
<td>64</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>KDD process</td>
</tr>
<tr>
<td>3.1</td>
<td>Conceptual model</td>
</tr>
<tr>
<td>3.2</td>
<td>Overview of the stages adopted in this study</td>
</tr>
<tr>
<td>4.3</td>
<td>Snapshot of file upload page</td>
</tr>
<tr>
<td>4.4</td>
<td>Snapshot of statistics page</td>
</tr>
<tr>
<td>4.5</td>
<td>Snapshot of preprocess missing values</td>
</tr>
<tr>
<td>4.6.1</td>
<td>Snapshot of Equal Width and Equal Frequency discretization techniques</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Snapshot of Equal Width discretization technique</td>
</tr>
<tr>
<td>4.6.3</td>
<td>Snapshot of customized discretization</td>
</tr>
<tr>
<td>4.6.4</td>
<td>Customized discretization creation of clusters</td>
</tr>
<tr>
<td>4.6.5</td>
<td>Management of tables</td>
</tr>
<tr>
<td>A.1</td>
<td>Login screen</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

Data Mining refers to extracting or mining knowledge from large amounts of data. Data mining is also referred to as Knowledge Discovery from Databases (KDD), knowledge extraction, data/pattern analysis, data archaeology, and data dredging [2]. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases [3].

Today’s real-world data are highly susceptible to noisy, missing, and inconsistent data. This data can cause confusion for the mining procedure, resulting in unreliable output. The techniques to preprocess data include data cleaning, data integration, data transformations, and data reduction [2]. Main tasks of data cleaning include removing noise and correcting inconsistencies in the data. Data integration merges data from multiple sources into a coherent data store, such as a data warehouse or a data cube [3]. Data transformations such as normalization improve the accuracy and efficiency of mining algorithms involving distance measurements [3]. Data reduction reduces the data size by aggregating, eliminating redundant features, or clustering. Data cleaning routines work to clean the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies [2].
1.1 What is data preprocessing?

Data preprocessing transforms the data into a format that will be more easily and effectively processed for the purpose of the user [2]. The techniques to preprocess data include data cleaning, data integration, data transformation and data reduction [2]. Data cleaning routines work to clean data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies [5]. If we would like to include data from multiple sources in our analysis, then we would integrate multiple databases, data cubes, or files, that is, data integration [5]. Data transformation operations include normalization and data aggregation. Data reduction obtains a reduced representation of the data set that is much smaller in volume, yet produces the same analytical results [5]. Strategies for data reduction include data aggregation, dimension reduction, data compression, and numerosity reduction.

1.2 Why preprocess the data?

Imagine that you are an analyst at a company and have been charged with analyzing the company’s data at your branch. You carefully inspect the company’s database, identifying and selecting the attributes or dimensions to be included in your analysis. Once you start analyzing, you will be able to notice that several of the attributes for various tuples have no recorded value and some of the data may have errors and unusual data.

Real world data are generally

- Incomplete: lacking attribute values, lacking certain attributes of interest;
• Inconsistent: containing discrepancies and inconsistencies due to data integration, where a given attribute can have different names in different databases. Redundancies may also exist;

• Noise is a random error or variance in a measured variable. Noisy data may be due to faulty data collection instruments, data entry problems, data transmission problems, technology limitation [14, 4].

1.3 What is data preprocessing management system?

Data preprocessing management system is the combination of both preprocessing of data and management of the preprocessed data. Preprocessing of data includes removal of missing values and discretization. Depending on the technique used, different files will be created; all these files will be managed by the management system. By removing the missing values three different files will be created, these will be stored by different names. On the client side, the user will be able to view and download these files. The user can select any file as an input file for doing the discretization. In discretization section, three different files will be created; these will be stored using different names and the user will be able to view and download these files. At the end, a list of all the tables created and the techniques implemented will be displayed in a table to the user. Prior to updating the tables into the database, the user will be able to see the distribution of each technique, using which, the user can compare each technique and can select the
appropriate technique to update the table. In this system the user can select any file and can compare it with other files using the statistics.

1.4 Statement of the problem

Real world data consists of noise, inconsistency and missing values. There are many techniques available to preprocess the raw data. But very few editors tried to manage the preprocessed data. Although many features are available through the predefined editors, the whole process still requires the user to fully understand the whole structure and associated objects in the definition process. This research tried to preprocess the data and manage the preprocessed data. Preprocessing of raw data is done in two steps; preprocess missing values and discretization of preprocessed data. Three basic techniques are implemented to preprocess missing values; EQW, EQF and customized discretizers are implemented to discretize the data. All the tables created and used are managed in this research. The user will be able to download any table and can use the table for further processing.

1.5 Objectives

The objective of this work is to preprocess and discretize the raw data using the techniques EQW and EQF discretization. There are various considerations and approaches for data preprocessing and several issues surround this process. Data
preprocessing management system is implemented from scratch from available raw data sources. The main objectives of the research and the contribution of this thesis are analyzing techniques of preprocessing and management of the preprocessed data, including the major issues faced, and demonstrating an efficient solution for them.

1.6 Issues considered:

- Data objects that are grossly different from or inconsistent with the remaining set of data are called outliers. Many data mining algorithms attempt to minimize the influence of outliers or totally eliminate them. However, such action can lead to the loss of important hidden information [20];
- Updating the data into the database once the appropriate technique is selected. In order to do this, distribution of each technique is displayed;
- Allowing the user to perform different discretization techniques on the same dataset;
- Allowing the user to view and download the files on the client side;
- Management of all the input(raw) and preprocessed data;

1.7 Motivations and Contributions

In reality, there is an ever-increasing usage of data across the globe, resulting in ever-expanding databases. Such databases hold raw, unprocessed data, at the same time
an equality large collection of knowledge. Most decision-makers want to be able to use the knowledge gained from their data for more tactical decision-making purposes.

The main contributions of this thesis are as follows:

- Development of a data preprocessing management system, for end users where a user can easily upload source data and can get the useful data.
- Development of a system for end-users, from which a user can easily manage all the source data and preprocessed data.
- Implemented techniques to remove missing values and discretization

1.8 Organization of the Thesis

The Thesis begins with an introduction to the data mining in chapter I which includes introduction, what is preprocessing?, statement of the problem, objective, motivations, and contributions.

Chapter II describes concepts and methodology of this research which includes KDD process, its nature and characteristics, preprocessing, its concepts and different techniques, preprocess missing values its techniques, discretization and techniques Equal Width (EQW), Equal Frequency (EQF) and Entropy Based discretization.

Chapter III describes the design of preprocessing management system which includes conceptual model, design of the stages involved and management. Chapter IV is an explanation of data preprocessing management system with an example. Chapter V is the implementation details of the Thesis. Chapter VI describes conclusions and suggestions for future work.
CHAPTER II
CONCEPTS AND METHODOLOGY

The KDD process extracts knowledge from original data and it begins with the original database from which the knowledge will be extracted [14, 15]. This original data will be used for the whole KDD process. Before the data undergoes data mining, they must be prepared in a preprocessing step that removes or reduces noise and handles missing values. Relevance analyses for omitting unnecessary and redundant data, as well as data transformation, are needed for generalizing the data to higher-level concepts [20]. Preprocessing techniques take the most effort and time, i.e. almost 80% of the whole project time for knowledge discovery in databases (KDD) [20]. The preprocessing step is vital for successful data mining.

2.1 KDD Process [14]

The overall process of KDD consists of the following steps as in Figure 2.1:

- Understanding of the application domain, relevant prior knowledge, and developing the goals of end-users;
- Selection of target data: selecting a dataset based on the requirements and goals;
- **Preprocessing**
  - Removal of noise and outliers;
  - Collecting the necessary information;
  - Strategies for handling missing data;
  - Accounting information of known changes.

- **Transformations: data reduction**
  - Finding useful features to represent the data depending on the objective;
  - Using dimensionality reduction to reduce the effective number of variables;

- **Data mining:**
  - Selecting methods to be used for searching for patterns in the data;
  - Deciding whether to use classification, regression or clustering;
  - Deciding which models and patterns may be useful;
  - Searching for patterns of interest in a particular representation form as classification rules, decision trees, and regression or clustering.

- **Interpreting mined patterns**
  - Removing redundant or irrelevant patterns, and translating the useful patterns into a form understandable by users.

- **Consolidating discovered Knowledge.**
Figure 2.1 KDD Process
2.2 Characteristics and nature of KDD applications [14, 16]

- KDD operates on large data sets
- KDD data sets are large in terms of number of attributes and number of records
- KDD attempts to deal with real world problems and real world data;
- Usually accesses input data several times;
- Builds dynamic and recursive data structures Hash tables, Linked lists, and Trees;
- Size and access of the data structure is data dependent;
- Complex core routines;
- KDD process consists of a number of interacting, iterative stages involving various data manipulation and transformation operations;
- KDD process is explorative;
- Information flows from one stage onto the next, as well as backwards to previous stage;
- At any stage the user can make changes (for instance to choose different tasks or techniques) and repeat the KDD process steps to achieve better results.
2.3 Preprocessing

Preprocessing transforms the data into a format that will be more easily and effectively processed for the purpose of the user [2]. Management manages all the data. In this research preprocessing is done in two steps: preprocess missing values and discretization.

2.3.1 Preprocessing tasks

Data cleaning

- Fill in missing values
  - Ignore the tuple: used when class label is missing;
  - Attribute mean: use the attribute mean to replace or fill in missing values;
  - Predict the missing values by using a learning algorithm.
- Identify outliers and smooth noisy data
  - Clustering: group attribute values in clusters then detect and remove outliers;
  - Binning: sort the attribute values and partition them into bins [9];
  - Regression: smooth data by using regression functions [9].
- Correct inconsistent data: use domain knowledge or expert decision

Data transformation;
• Normalization [3, 4]
  ▪ Scaling attribute values using mean and standard deviation;
  ▪ Scaling attribute values to fall within a specified range.
• Aggregation: moving up in the concept hierarchy on numeric attributes;
• Generalization: moving up in the concept hierarchy on nominal attributes;
• Attribute construction: replacing or adding new attributes inferred by existing attributes.

Data reduction
• Reducing the number of attributes
  ▪ Removing irrelevant attributes: attribute selection;
  ▪ Searching for a lower dimensional space that can best represent the data[17];
• Reducing the number of attribute values
  ▪ Binning (histograms): reducing the number of attributes by grouping them into intervals (bins);
  ▪ Clustering: grouping values in clusters;
  ▪ Aggregation.

2.3.2 Preprocess missing values

In this step all the nulls and missing values will be preprocessed. Three basic techniques are implemented in this thesis to preprocess missing values. According to the user needs, the user can apply any technique. All the three techniques can be applied on the same dataset and can select the desired technique, based on comparison of the
distribution and statistics of each technique. The three techniques are replace missing values, delete missing values, and customization. Missing values considered are null, ‘’,?.

2.3.3 Replace missing values

The idea behind this technique is to replace missing values with the mean of that attribute. If an attribute is numeric, then all the missing values of that attribute will be replaced by Average of that attribute. If an attribute is nominal, then all the missing values will be replaced by Most Recurrent value of that attribute.

Algorithm

X [1...N] input vector of attribute values
A [I] contains all attributes
For the selected attribute of A [1...I]
{
    If (attribute type == “nominal”)
    {
        Missing values of X [1…N] = AVG (X [1…N]);
    }
    Else if (Attribute type == “numeric”)
    {
        Missing values of X [1…N] = Most Recurrent value (X [1…N]);
    }
}
Customization is solely a user technique. According to this technique, the user will have the option to replace missing values. The user can enter any value to replace the missing values of an attribute.

In this research the user can view the distribution of each attribute, so the user will have an idea of the range of the values of the selected attribute. By looking at the distribution the user can enter a value to replace missing values. But if the attribute is numeric, then the user should enter a numeric value to replace missing values, and if the attribute is nominal, then the user should enter a nominal value. By this technique the user is customizing the missing values with a desired value.

2.3.4 Delete missing values

Using this technique, for any attribute all the missing values will be deleted. For example, if an attribute has 12 missing values, then 12 records will be deleted from the data set.

This technique is good if the data set contains few missing values. If the dataset has more missing values then by using this technique we are losing the knowledge of the deleted records.

Once the missing values are processed, the next step is the discretization, in which clusters will be created.
2.4 Discretization

Discretization is the second major part of preprocessing. Discretization is “A process that transforms quantitative data into qualitative data” [17]. With discretization a variable is transformed into a discrete one.

The existing discretization algorithms can be described as supervised versus unsupervised, global versus local and static versus dynamic [17]. The supervised algorithms use class information while unsupervised algorithms do not use it at all. Global discretization is a preprocessing step carried out prior to the process of constructing a classifier, while local methods perform discretization during such processes [17, 19]. The static methods discretize each attribute independently, and do not consider interactions with other attributes, while the dynamic methods are searching for discretization intervals for all the attributes simultaneously, thus capturing attribute interdependencies [17, 18].

- Unsupervised discretization
  - Equal-width binning: split the whole range of numbers in intervals with equal size;
  - Equal-frequency binning: intervals will have an equal number of attribute values.

- Supervised discretization
  - Using class boundaries
    - Sort the values;
    - Place breakpoints between values belonging to different classes;
If too many intervals, then merge intervals with equal or similar class distribution;

- Entropy based discretization
  - The entropy is calculated on the basis of the class label. It finds the best split so that the bins are as pure as possible; the majority of the values in a bin correspond to having the same class label. It is characterized by finding the split with maximal information-gain.

In this section I explain four techniques of discretization.

- Equal Width discretization
- Equal Frequency discretization
- Entropy Based discretization
- Customization

In data preprocessing management system, three techniques are implemented; they are Equal Width, Equal Frequency, and Customization.

2.4.1 Equal Width discretization

Equal width discretization divides the range of attributes into a fixed number of intervals of equal length [2]. The user specifies the number of intervals as a parameter or can be computed by a heuristic formula such as \( k = M/ (3n) \) [3], where \( k \) = number of clusters, \( M \) is the number of examples and \( n \) is the number of classes. The initial discretization scheme is \([i_0, i_n]\). Width \( w \) can be computed as \( \text{max-min}/M \), where \( \text{min} \) is
the minimum value and max is the maximum value of the selected attribute and M is the number of instances. Once the width is computed, the cut-off points can be generated. The cut-off points generated are as follows:

\[ [i_0+w, i_0+2w, \ldots, i_0+nw] \]

It’s a global unsupervised algorithm. In this technique some intervals possibly cover many values occurring in the table; other intervals eventually only contain a small number of values.

Advantages

- Simplicity.

Disadvantages

- Unbalanced intervals;
- Presence of outliers;
- Unsupervised, so it ignores the class information.

Equal frequency discretization overcomes the potential drawback of unbalanced intervals and tries to balance the number of attribute values within each interval.
Algorithm

Input – Data table N

C number of classes in input
k desired number of intervals
M number of instances in input
X [1…M] input vector of attribute values

Output

d [0…k] vector of discretized boundaries with intervals [d0, d1), [d1, d2)…[dk-1, dk]
A[h] contains all attributes
Min = MIN(X (1…H));
Max = MAX(X (1…H));
W = (Max-Min)/k;
For j= 1 to k
Do
  d[j] = Min + W*j;
End For

Output Cut-off points for all attributes, Dataset N with clusters

Listing 2.4.1 Equal Width discretization algorithm
2.4.2 Equal Frequency discretization

Equal frequency discretization divides the data into a specified number of intervals so that approximately the same number of attribute values (instances) falls into each of the intervals [2]. It divides the data into k bins, having each bin with M/k instances where M is the number of instances and k is the number of clusters.

For each and every attribute a sort operation is required. Once the attribute is sorted, depending on the number of clusters each bin will have n/k instances of the selected attribute [1]. So each bin consists of sorted n/k values having equal values in the same bin. It's a global unsupervised algorithm. This technique ignores the class information. But it’s not always possible to get an equal number of attribute values in each cluster; if number of instances is an even number, then the number of instances in each cluster will differ.

Advantages

- It avoids the problem of outliers;
- Each cluster will have an equal number of attribute values;
- It’s easy to interpret.

Disadvantages

- This method ignores the correlation in data[17];
- It loses some useful data information;
- Not always possible to get an equal number of instances in each cluster.
Listing 2.4.2 Equal Frequency discretization algorithm

Algorithm

Input

Data table M

C number of classes in input
k desired number of intervals
M number of instances in input

X [1: M] input vector of attribute values

Output

d [0...k] vector of discretized boundaries with intervals [d0, d1), [d1, d2)... [dk-1, dk]
A[h] contains all attributes

For the selected attribute

For j = 1 to k

Do

Sort(X [1: M]);
Z = M/k;
D0 = Min(X [H]);
Dj = X [j*Z];

End For

Output cut-off points for all attributes
2.4.3 Entropy Based discretization [8]

Entropy based method uses the class-information present in the data. Entropy is calculated on the basis of the class label. Intuitively, it finds the best split so that the bins are as pure as possible; the majority of the values in a bin corresponds to the class label. It is characterized by finding the split with the maximal information gain. Following is an example of this technique [8]:

This example consists of an attribute Age with class value (Yes, No)

<table>
<thead>
<tr>
<th>Marks</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Yes</td>
</tr>
<tr>
<td>16</td>
<td>No</td>
</tr>
<tr>
<td>24</td>
<td>No</td>
</tr>
<tr>
<td>26</td>
<td>No</td>
</tr>
<tr>
<td>28</td>
<td>No</td>
</tr>
<tr>
<td>18</td>
<td>Yes</td>
</tr>
<tr>
<td>16</td>
<td>No</td>
</tr>
</tbody>
</table>

Fraction of Marks with class value Yes $p_1 = 4/9$

Fraction of Marks with class value No $p_2 = 5/9$

The Entropy $S$ is defined as
Entropy(S) = \( -p_1 \log_2(p_1) - p_2 \log_2(p_2) \)

So the entropy according to this example = .9910

If the entropy is small, then the set is relatively pure. The smallest possible value is 0. If the entropy is large, then the set is mixed. The largest possible value is 1, which is obtained when \( p_1 = p_2 = 0.5 \)

Given a set of samples S, if S is partitioned into two intervals S1, and S2 using boundary T, the entropy after portioning is [8]

\[
E(S, T) = \frac{|S_1|}{|S|} \text{Entropy}(S_1) + \frac{|S_2|}{|S|} \text{Entropy}(S_2)
\]

If we choose the midpoint of the attribute Marks \( T = 14 \)

\( S_1 = [(0, \text{Yes}), (4, \text{Yes}), (12, \text{Yes}) \) and \( S_2 = (16, \text{No}), (16, \text{No}), (18, \text{Yes}), (24, \text{No}), (26, \text{No}), (28, \text{No})] \)

\[
E(S, T) = (\frac{3}{9})E(S_1) + (\frac{6}{9})E(S_2)
\]

= 0.4333

Information gain of the split, \( \text{Gain}(S, T) = \text{Entropy}(S) - E(S, T) \)

\( \text{Gain} = 0.5577 \)

• The objective of this algorithm is to find the split with the maximum information gain and is obtained when \( E(S, T) \) is minimal [8].

• The best splits are found by examining all possible splits and then selecting the optimal split [8].
The process is recursively applied to partitions obtained until some stopping criterion is met

\[ \text{Entropy}(S) - \text{Entropy}(T, S) > \text{Threshold value.} \]

Where \( \text{Threshold value} > (\log (N-1) / N) + (\Delta (T, S) / N) \) [8]

\[ \Delta(S, T) = \log_2(3^c - 2) - [c \, \text{Entropy}(S) - c_1 \, \text{Entropy}(S_1) - c_2 \, \text{Entropy}(S_2)] \]

c is the number of classes in S, c1 is the number of classes in S1 and c2 is the number of classes in S2 [8].

Customization

This technique depends on user decision. Depending on the distribution of the attribute a user can enter values as cut-off points. The number of clusters is user defined; once the number of clusters is entered, the user will have to enter the cut-off points according to the range of values of the attribute.

For each cluster user should enter a range of values (minimum and maximum). Once the clusters are defined, the attribute distribution will be displayed according to the cluster range. The user can change the range and can view the distribution for the new cluster. Using this method the user will have more flexibility in getting accurate clusters according to the user’s need. Once the clusters are accurate, then the user can update the dataset to reflect the changes in attribute values.
CHAPTER III
DESIGN OF PREPROCESSING MANAGEMENT SYSTEM

The design of data preprocessing management system is explained in two sections; conceptual model and design of the stages. Conceptual model explains the basic details from the business perspective. This is explained using the entity-relationship diagram assuming all the major parts as entities and the methods involved as relationships. Second section which is the design of the stages involved explains all the stages in detail.

3.1 Conceptual Model

A conceptual model defines the types of entities or objects which are of immediate interest and the relationships between them without being specific to any particular database semantics. Below figure explains the conceptual model, with all the entities in rectangular boxes, relations in diamond shape and attributes in oval shape connected with each other using connectors.
As in Figure 3.1 Input file, input table statistics, preprocessed table, and discretized table are the main entities of the system. Input file has attributes “missing values” and “attributes”. Input table statistics has an attribute “statistics”. Preprocessed table has
attributes “data” and “attributes of the preprocessed table”. Discretized table has attributes “clusters” and “attributes.” Relationships preprocess missing values had attributes “replace missing values” and “delete missing values”; relation discretization has attributes “EQF” and “EQW”.

The two entities; input file and input table statistics are related using statistics; preprocessed table and discretized table entities are related with the relationship discretization.

The whole procedure lying in this system was to develop a system where a user uploads the input file; statistics of the input file will be displayed; from this input file missing values will be removed; using this preprocessed missing values table discretized tables will be created by applying the discretization techniques. As in figure 3.1 input file consists of missing values and data, preprocessed table consists of attributes and data with no missing values, and discretized table consists of clusters and attributes. Finally all these tables will be managed. In future user can add any new technique to this system and can use any table as an input table; extendibility is achieved in this way.
3.2 Overview of the Stages

Start

Connect to SQL Server

Upload flat file
Text, Excel, CSV

Statistics of Raw data

Preprocessing

Preprocess of missing values

Discretization

Management

End

Figure 3.2 Overview of the Stages
As explained in figure 3.2 the main stages of this thesis are uploading the raw data, displaying the statistics of the raw data, preprocessing of missing values, discretization and management of the preprocessed data. In the first stage, the user will be connected to the SQL server using the server name, database of the server mentioned, login id, and password.

Once the connection is established, a user can upload a text or Excel or CSV file with delimiters specified. The uploaded file will be stored in a table in the database entered. The next stage in this study is displaying the statistics of the raw data. Statistics includes input file information and details of the data.

The main step of this research is the preprocessing of raw data, which includes two parts; preprocessing of missing values and discretization of the preprocessed data. For removing the missing values three techniques are implemented. They are replace missing values, delete missing values and customization. In this stage user will be able to observe the distribution of each technique before applying the technique. Discretization uses the preprocessed data as input data. For implementing discretization three techniques are implemented: equal width discretization, equal frequency discretization and customized discretization. Once the data is discretized, clusters will be created. But all the raw and preprocessed data is yet to be managed.

The last stage of this research is the management. In this stage all the raw data and preprocessed data will be managed by storing all the data in different tables and providing an option for the user to view and download all the data at the client side. At this stage all the table names which are created and the techniques applied on those tables will be displayed so that the user can use these tables for further processing.
3.3 Management

Managing all the input data and output data at each stage is done in this section for each technique.

In this thesis I have implemented data management.

- In the uploading stage, displaying all the tables which already exists in the database;
- Storing all the data at each stage in a different table;
- Allowing the user to enter a table name;
- Displaying the data statistics after implementing each technique;
- Once all the techniques are implemented, then a table is created with all the information of all the tables, and displaying the techniques through which the tables were created;
- In that table, for every table created, its base table and type of table is displayed so that user will have an idea of which table to use for further processing;
- All the tables that are created are available to both the client and the server side to view and download.
CHAPTER IV
IMPLEMENTATION

Data preprocessing management system uses SQL server 2005 as backend and C#.Net, ASP.Net as front-end.

4.1 Advantages of SQL versus WEKA

- There is no limit on the file size in SQL where as WEKA doesn’t allow big files;
- SQL supports all file types but WEKA has limitations on the file types;
- The dataset uploaded to WEKA cannot be reused or is not permanent, but SQL server allows the reuse of the table and the dataset will be in the database until the user chooses to delete it;
- SQL supports working with multiple datasets, but WEKA doesn’t allow the user to do this;
- The preprocessing techniques which are not implemented in WEKA can be implemented using SQL queries;
- Using SQL we can get the list of all the tables and databases of the server but in WEKA we won’t be able to do that;
To store the data of different techniques, seven tables will be created in this application. The following sub sections give a brief overview of the components of the application. This application consists of 8 ASP.Net pages.

4.2 Connection establishment

SqlServer.aspx

This is the start page of the application, where all the login details should be entered. In this page user should enter the location of the server. If it’s local, then local should be entered, or if it’s ipaddress, then ipaddress should be entered. The user should also enter name of the server, name of the database which is accessible, userid which is the login name for the SQL server, and password which is the password of the login entered. Once all the details are entered, control will be directed to the next page, which is the input file upload page. Using the ConnectionString, connection will be established and the ConnectionString will look like this:

```
ConnectionString = "connectionString" value="Data Source=(local);Initial catalog = Thesis; user id=kanni; pwd=anumalla";
SqlConnection connection = new SqlConnection(connectionString);
```

Using the above statements, connection will be established.

4.3 Uploading an input file

Home.aspx

Once the connection to the SQL server is established, the next step, which is uploading the input file, is done in this page. In this section there are two options to upload input
file; use an existing table as an input table or upload a new input file to SQL server. In the first option all the existing tables of the selected database will be shown in a drop down box, from which a table can be selected.

For uploading a new input file, there will be a Browse button, using which a file can be browsed and uploaded. The column delimiters allowed are comma, colon, CRLF, period, semicolon and tab, and the row delimiters allowed are CRLF, comma, period, semicolon and tab. A text box is provided on the page through which the user can enter a name for the table to be created.

```csharp
{  
    string sql = "Create Table " + tableName + "(";
    string[] tokens = data.Split(new string[] { colDelimiter }, StringSplitOptions.None);
    
    for (int i = 0; i < tokens.Length; i++)
    {  
        if (i > 0)
            sql += ",";
        sql += tokens[i] + " varchar(20)";
    }
    sql += ");"
    
    command.CommandText = sql;
}
```

Using the above code, a table will be created, colDelimiter is the column delimiters; the input file will be split using the colDelimiters, and the data will be stored with the datatype varchar(20) initially.
Below is the snapshot of File uploading page (Home.aspx)

![Snapshot of File Upload Page](image)

**Figure 4.3 Snapshot of File Upload Page**

4.4 Displaying Statistics of input table

Statspage.aspx

In this page all the statistics of the input file will be displayed. All the attributes of the table are displayed in a RadioButtonList from which, if an attribute is selected, then all the statistics and distribution of the selected attribute will be displayed.

If an attribute is selected then its table name, attribute name, type of the attribute, number of instances, distinct count, unique count, min, max, average, standard deviation
and missing value count for a numeric attribute, and for a nominal attribute table name, attribute name, type of the attribute, number of instances, distinct count, unique count, and missing count will be displayed. For each selected attribute, its distribution will be displayed. Distribution includes attribute values with their distinct counts.

Column\texttt{Type} is computed using the following function:

\begin{verbatim}
if (double.TryParse(value, out value2))
    columnType = "Numeric";
\end{verbatim}

Here \texttt{value} is the data of the table, excluding nulls and missing values, and \texttt{value2} is initialized to zero.

Figure 4.4 Snapshot of Statistics Page
string avg = dt2.Compute("Avg(" + lstColumns.SelectedValue + ")", ",").ToString();

Using the above statements, all the statistics operations are done; for example, for computing the StdDeviation, use StDev in place of Avg; and dt2 is the datatable, which consists of all the attribute values of the selected column (attribute).

As shown in Figure 4.4 all the attributes will be displayed in a RadioButtonList from which if an attribute is selected then its statistics will be displayed as ColumnGeneralInfo as in the above figure, and Attribute distribution will be displayed in a Grid with distinct values of the attribute with their count. Initially the first 15 values will be displayed; if the user wants to see more, just click on “Click for more,” which will redirect the user to a form Distinct.aspx, which is explained in the next section.

Distinct.aspx

In this page the distribution will be displayed. In the previous page, for any selected attribute the first 15 rows distribution will be displayed. If the user wants to see the total distribution of the selected attribute, then this page will be used. In this “from and to” row, numbers of the distribution should be entered.

The distribution is displayed in all the pages using a Grid, is explained in the following SQL query.

SqlDataAdapter daa;
da =("SELECT Value, Count FROM ( SELECT DISTINCT Count(" + 
 lstColumns.SelectedValue + ") AS Count, (" + lstColumns.SelectedValue + ")Value, ROW_NUMBER() OVER (ORDER BY " + lstColumns.SelectedValue + ")Value, ROW_NUMBER() OVER (ORDER BY " + lstColumns.SelectedValue + ")Value, ROW_NUMBER() OVER (ORDER BY " + lstColumns.SelectedValue + "

35
DESC) AS Row from " + qsl + " GROUP BY " + lstColumns.SelectedValue + ")
AS emp1 WHERE Row BETWEEN 1 AND 15", connectionString);
da.Fill(dtt);
grdData.DataSource = dtt;
grdData.DataBind();

In the above query lstColumns.SelectedValue is the selected attribute of which, the user
wants to view the distribution. In SQL server 2005 a new function was introduced, for
finding the RowNumber which is not available in the previous versions, and the function
is ROW_NUMBER() Over (order by columnname). The grdData is the data grid name.

4.5 Preprocesssing of input table

Preprocess.aspx

In this section the preprocessing is implemented. In this page, three basic
techniques of preprocess missing values were implemented. All the attributes will be
displayed in a RadioButtonList as shown in Figure 4.5. If an attribute is selected, its base
table statistics will be displayed.

The three implemented techniques are: delete missing values, replace missing
values, and customization. Customization technique replaces all the missing values with a
user defined value. Delete missing values technique deletes all the missing values.
Replace missing values technique replaces all the missing values with the mean of the
attribute if it is numeric, and if it’s nominal then, all the missing values will be replaced
by most occurring value. The user can compare each technique using the statistics and
distribution.
As shown in the figure 4.5 the three techniques will be displayed in a RadioButtonList. Initially the base table distribution and statistics will be displayed. Next to the statistics there will be a DropDownBox which will be empty initially, once a technique is implemented, the user can select the technique to view its statistics and distribution. In this page, the user should enter a name for the table to store the data of each technique. Once all the techniques are implemented, the user can select any discretization technique, and the control will be redirected to the selected discretization technique page.

The query used for delete missing values technique:

```csharp
SqlDataAdapter dal = new SqlDataAdapter("DELETE FROM " + ppp + " WHERE " + lstColumns.SelectedValue + " = 'Null' OR " + lstColumns.SelectedValue + ")
```
The query used for customization:

```
Dal = ("UPDATE " + ppp + " SET " + lstColumns.SelectedValue + " = " + replacevalue + " WHERE " + lstColumns.SelectedValue + " ='null' or " + lstColumns.SelectedValue + " ='? ' OR " + lstColumns.SelectedValue + " ='NULL'", connectionString);
```

Here replacevalue is the value entered by user for replacing the missing values and ppp is the table name which is being updated.

4.6 Discretization of preprocessed data

WidthandFrequency.aspx

In this section discretization techniques are implemented. In this page two discretization techniques are implemented, equal width and equal frequency discretization. The input table for discretization, is the output table of the preprocess missing values; in the previous section all the missing values were deleted, and output tables were created. In this page all the preprocess missing values tables will be displayed in a RadioButtonList. From the list user should select a table as an input table. Once the table is selected, the user can select an attribute to do the discretization. In this page also all the tables, statistics and distribution will be displayed. The user should enter the number of clusters. As in Figure 4.6 Base table statistics and discretized table statistics
can be used for comparison purpose, where a user can easily compare each technique statistics with the base table statistics, and can select the best technique.

Figure 4.6.1 Snapshot of Equal Width and Equal Frequency discretization Techniques

In equal width discretization, if the user enters three clusters, then three clusters will be created; and width between each cluster will be equal. As in Figure 4.6e there will be two buttons, one is show equal width disc and the other one is the commit equal width disc. When the first button is clicked, then for the entered number of clusters the distribution, and cluster cut-off points will be displayed. When the second button is clicked, the actual updations to the table will be done i.e., a new table will be created with the clusters instead of attribute values (clusters will be created). Once all the experiments
are done, if the exact desired clusters are created then the user should click the commit equal width disc button; by using this actual updations will be done in the SQL server.

Figure 4.6.2 Snapshot of Equal Width discretization technique implementation

As in Figure 4.6.2 distribution consists of three clusters with the distinct count in each cluster, and the clusters with the cut-off points will be displayed. The width between each cluster is same as cluster 1 from 0 to 34, cluster 2 from 34 to 68, and cluster 3 from 68 to 102. The main disadvantage of this technique is that the number values in each cluster are not equal; in this case cluster 1 consists of 1 value, cluster2 also 1, and cluster 3 consists of 47.

Query for implementing equal width discretization:

```c
for (j = 1; j <= cluster; j++)
{
    B1[0] = min;
    B1[j] = min + (j * width);
}
```
In this loop all the clusters cut-off points will be computed. According to the algorithm width = Max-Min/cluster, where cluster is the number of clusters and min is the minimum value of that attribute.

```c
for (j = 1; j <= cluster; j++)
{
    SqlDataAdapter aa = new SqlDataAdapter("select (" + j + ")Cluster, count(" + lstColumns.SelectedValue + ")Count from " + qsl1 + " WHERE " + lstColumns.SelectedValue + " >= " + min + " AND " + lstColumns.SelectedValue + " <= " + B1[j], connectionString);
    aa.Fill(dttt);
    grdData4.DataSource = dttt;
    grdData4.DataBind();
    min = B1[j];
}
```

In equal frequency technique, number of clusters is a user defined number. According to this technique each cluster contains equal number of values. But if the number of instances in the dataset is an odd number then all the clusters does not contain equal number of values, if it’s an even number then all the clusters will contain equal number of values. The number of values in each cluster will be equal to N/k, where N is the number of instances and k is the number of clusters. If the button show equal frequency desc is clicked then the distribution for these clusters will be displayed. If the button commit equal frequency desc is clicked, then the cluster values will be updated into the table. The user can select any table, (delete missing values, replace missing values, customized table) as an input table, and all the techniques can be applied on the
same dataset. The last technique is the customized discretization, which is implemented in another page.

The query used for implementing equal frequency technique:

```sql
sql2 = "SELECT * INTO #test FROM " + qsl1 + " ORDER BY " + qsl1 + "." + lstColumns.SelectedValue + " ASC ALTER TABLE #test ADD id int IDENTITY (1, 1) NOT NULL UPDATE #test SET " + lstColumns.SelectedValue + " = " + j + " WHERE id >= " + mi + " AND id <= " + ttcount + " ALTER TABLE #test DROP COLUMN id TRUNCATE TABLE " + qsl1 + " INSERT INTO " + qsl1 + " SELECT * FROM #test DROP TABLE #test"
```

This is a little complicated query; in this another ideal table (temporary) will be created for updating the base table, all the changes will be done in this table, and once it is done, all the data from this table will be copied back to the base table. In the above query #test is an ideal table, where a column id will be created temporarily for rownumbers.

Custom.aspx

In this page customized discretization is implemented. This is a user defined discretization, where the user will enter the number of clusters, and the cut-off points for each cluster, so the user can change these values, and can select the best cut-off points. Below is the snapshot of this implementation.
Figure 4.6.3 Customized discretization implementation

As shown in Figure 4.6.4 once user enters the number of clusters, the textboxes will be created dynamically to enter the cut-off points for the clusters.

Figure 4.6.4 Customized discretization creation of clusters
For each cluster the min value and the max value should be entered, and these values should be in the range of the attribute. Once the cut-off points are entered, the distribution will be displayed. The number of clusters can be changed at any time. If the button commit customization is clicked, then the cluster values will be updated to the table. In all the three techniques, the created tables can be downloaded from the links specified in each page onto the client side.

The query used for customized discretization:

```sql
sqlc = "UPDATE " + custtab + " SET " + lstColumns.SelectedValue + " = " + i + " WHERE " + lstColumns.SelectedValue + " >= " + lower + " AND " + lstColumns.SelectedValue + " <= " + upper;
```

Here upper and lower are the cut-off points of each clusters entered by user.

Management.aspx

In this section a table will be displayed consists of all the tables created, and all the techniques implemented along with the input tables. This is helpful for further processing, so that the user can understand easily, the tables downloaded and the techniques applied on those tables. In this page all the table names and the techniques applied are maintained in sessions; using those sessions all the tables and techniques used on those tables is displayed in the below figure.
<table>
<thead>
<tr>
<th>Type of Data Table</th>
<th>Table Name</th>
<th>Base Table</th>
<th>Delete Table</th>
<th>Average Table</th>
<th>Custom Replace Table</th>
<th>Equal Width Table</th>
<th>Equal Freq Table</th>
<th>Custom Dist Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Table</td>
<td>grades</td>
<td>RAW Table</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Delete mean value table</td>
<td>delete</td>
<td>grades</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Replace mean value table</td>
<td>replace</td>
<td>grades</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Custom mean value table</td>
<td>custom</td>
<td>grades</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Equal width table</td>
<td>width</td>
<td>delete</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Equal frequency table</td>
<td>freq</td>
<td>delete</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Customized table</td>
<td>custom</td>
<td>delete</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Figure 4.6.5 Management of tables
CHAPTER V

CONCLUSION AND FUTURE WORK

Data preprocessing management system preprocesses the raw data into a format, which will be more easily and effectively processed for the purpose of the user. Using this system, the user can easily compare each technique, by using the distribution and statistics of all the techniques. Data and statistics are managed at every stage for better understanding of the techniques. In all the three techniques, Equal Width is simple to implement but has the disadvantage of outliers. Equal Frequency is also simple to implement and it doesn’t have the problem of outliers but it’s not true, that always all the clusters will have equal number of values. Coming to the customization user can select the bin ranges but, the algorithm has to be improved considering all the possibilities.

More work has to be done in data management, considering the client side security of the files downloaded. The user should have the option of selecting the tables from data management and should be able to implement their own techniques
REFERENCES


This section presents an example implementing the procedures presented in the previous chapters.

Table A.1 Input table
Above file is the raw input file, which consists nulls, ‘ ‘ and ?.

Let us consider the above table Grades, which holds data about Marks. The columns in that table are HWTotal, Midterm, Midterm2, Midterm3 and LetterGrade, with LetterGrade of type nominal and all the others are of type numeric. This example table is used in the following sections as base table.

Below is the login screen for the data preprocessing management system.

![Login Screen](image)

**Figure A.1 Login screen**

In the login screen enter location of the server ex: local or 130.101.10.2
Database is the name of the database of SQL server.

User id is the login of the SQL server and password is the password of the SQL server, for the login entered.

For the uploaded input file statistics are displayed as in Figure A.3.

Table A.1.1 Statistics of the input table

<table>
<thead>
<tr>
<th>Statistics</th>
</tr>
</thead>
</table>
| TableName: grades  
Name: HWTotal  
Type: Numeric  
Instances: 73  
Distinct: 42  
Unique count: 27 (36%)  
Min: 0  
Max: 200  
Avg: 156  
StDev: 45.83  
Missing Values: 12(16%) |

For the attribute HWTotal, above are the statistics of the input file. TableName is the name of the input table, Name is the attribute name, Type is the type of the selected attribute, Instances are the number of records of HWTotal, distinct is the distinct count, Unique count is the unique count of HWTotal, min and max are the range of values, AVG is the average of HWTotal, StDev is the standard deviation of HWTotal, and missing values is the missing value (null, ‘’,?) count of the attribute.
A.1 Preprocess missing values

Three techniques are implemented in this section; replace missing values, delete missing values and customization. From the statistics table, average of HWTotal is 156. Using replace missing values technique all the missing values are replaced by 156 for the attribute HWTotal.

Below are the statistics of the table after applying replace missing values.

Table A.1.2 Statistics of the replace missing values table

<table>
<thead>
<tr>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>TableName: ReplaceMissing</td>
</tr>
<tr>
<td>Name: HWTotal</td>
</tr>
<tr>
<td>Type: Numeric</td>
</tr>
<tr>
<td>Instances: 73</td>
</tr>
<tr>
<td>Distinct: 42</td>
</tr>
<tr>
<td>Unique count: 27 (36%)</td>
</tr>
<tr>
<td>Min: 0</td>
</tr>
<tr>
<td>Max: 200</td>
</tr>
<tr>
<td>Avg: 156</td>
</tr>
<tr>
<td>StDev: 41.83</td>
</tr>
<tr>
<td>Missing Values: 0(0%)</td>
</tr>
</tbody>
</table>

From the table A.4, the missing values are nil as the missing values are processed. The modified data is stored in a different table. Grades is the input file, ReplaceMissing is the table name for replace missing values technique.

Delete missing values technique deletes 12 records for the selected attribute.
Below are the statistics of delete missing values technique.

Table A.1.3 Statistics of delete missing values table

<table>
<thead>
<tr>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>TableName: DeleteMissing</td>
</tr>
<tr>
<td>Name: HWTotal</td>
</tr>
<tr>
<td>Type: Numeric</td>
</tr>
<tr>
<td>Instances: 61</td>
</tr>
<tr>
<td>Distinct: 41</td>
</tr>
<tr>
<td>Unique count: 27 (44%)</td>
</tr>
<tr>
<td>Min: 0</td>
</tr>
<tr>
<td>Max: 200</td>
</tr>
<tr>
<td>Avg: 156.23</td>
</tr>
<tr>
<td>StDev: 45.83</td>
</tr>
<tr>
<td>Missing Values: 0(0%)</td>
</tr>
</tbody>
</table>

From the table A.5, it is clear that there are no missing values in the table. The number of instances in input table is 73, but the “DeleteMissing” consists of 61 instances; from this it is clear that 12 records are deleted from the table.

Using the delete missing values technique some useful records is deleted. This technique is useful when the dataset consists of few missing values. In each technique user is allowed to enter a name for the table, so that all the processed data is stored into that table.
A.2 Customization

In this technique all the missing values are replaced by a user entered value. The user should enter a value to replace the missing values.

Below are the statistics of customized table

Table A.2 Statistics of customized preprocess missing values table

<table>
<thead>
<tr>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>TableName: Customized</td>
</tr>
<tr>
<td>Name: HWTotal</td>
</tr>
<tr>
<td>Type: Numeric</td>
</tr>
<tr>
<td>Instances: 73</td>
</tr>
<tr>
<td>Distinct: 42</td>
</tr>
<tr>
<td>Unique count: 27 (36%)</td>
</tr>
<tr>
<td>Min: 0</td>
</tr>
<tr>
<td>Max: 200</td>
</tr>
<tr>
<td>Avg: 146.82</td>
</tr>
<tr>
<td>StDev: 46.97</td>
</tr>
<tr>
<td>Missing Values: 0(0%)</td>
</tr>
</tbody>
</table>

Table name is customized. Missing value count for the attribute HWTotal is 0; all the nulls are replaced by value ‘99’.

For each technique, a new table is created with a name specified by the user and the updated table is stored into the new table.

In this thesis the user can view and download the tables which are created, on the client side.
A.3 Discretization

This section uses the tables which are created in the previous section as input to this section. If the user wants to use replace missing values as input table to the discretization then that table should be selected.

What we did till now is uploading the input raw table, observing the statistics of the raw table and implemented the techniques of preprocess missing values and observed its statistics and compared them with the raw table.

In this section three techniques are implemented; equal width discretization, equal frequency discretization, and customized discretization. Users have the flexibility of applying techniques as they can view the distribution of each technique before applying it, so the user can compare and then, user can apply the technique.

In all the three techniques number of clusters $k$ is user defined. Replace missing values table is the input table for all the discretization techniques. Below is the example of equal width discretization
Table A.3 Distribution of Equal Width discretization

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>63</td>
</tr>
</tbody>
</table>

From the above figure Cluster column has 1, 2, 3 values which are cluster1, cluster2, cluster3, with another column Count, which is the count of the attribute values in each cluster. Using this equal width discretization the width between each cluster is equal, but the count of values in each cluster is varying.

Below are the cut-off points for this technique.

Cluster 1: (0, 69)
Cluster 2: (69, 138)
Cluster 3: (138, 207)

Cluster 1 has the attribute values between the ranges 0 to 69, cluster 2 has the attribute values between the ranges 69 to 138 and cluster 3 has the attribute values between the ranges 138 to 207. Width between each cluster is equal.
Table A.3.1 Equal Width Discretization file

From the above figure it is clear that, all the values are transformed into discrete values. We can observe from the distribution, that each cluster has different count of values. To avoid this, equal frequency discretization is implemented.
A.4 Equal Frequency discretization

In this technique each cluster has equal number of attribute values. Number of clusters is a user defined number.

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
</tr>
<tr>
<td>3</td>
<td>23</td>
</tr>
</tbody>
</table>

Table A.4 Distribution of Equal Frequency discretization

In this algorithm the selected attribute is sorted first; and each cluster will have $M/k$ values where $M$ is the number of instances in this table and $k$ is the number of clusters. So the cut-off points are generated from row number 1 to $M/k$ and $M/k+1$ to $2*M/k$…$k*M/k$.

Below are the cut-off points for Equal Frequency discretization algorithm

Cluster 1: From 1 To 25
Cluster 2: From 26 To 50
Cluster 3: From 51 To 75
From the distribution and from the figure A.11 it is clear that there is equal number of values in each cluster. The user can view and download this file in client side.
A.5 Customized Discretization

In this technique the user should enter the cut-off points for each cluster. Depending on the distribution user can commit the changes or modify the cut-off points.

In this number of clusters is a user defined value, once the user enters this, depending on the number of clusters textboxes are created for entering the cut-off points.

For example if the user enters 3 as the number of clusters then the following boxes are created for entering cut-off points or bin ranges.

1. Lowerlimit 0 Upperlimit 100
2. Lowerlimit 101 Upperlimit 150
3. Lowerlimit 151 Upper limit 200

For the attribute HWTotal range is 0 to 200, so the user should enter the cut-off points between that ranges.

Table A.5 Distribution of the customized discretization

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>55</td>
</tr>
</tbody>
</table>
Before updating the table the bin ranges can be modified, and can compare which cut-off points will work better for getting exact clusters. Once the cut-off points are selected then the user should enter a name for the table to store the updates table.

Table A.5.1 Customized discretization file

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWTotal</td>
<td>Midterm</td>
<td>Midterm2</td>
<td>Midterm3</td>
<td>Final</td>
<td>Letter/Grade</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2 D</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3 D</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2 C</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2 A</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3 A</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3 B</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3 B</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>13</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2 A</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>17</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3 B</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3 A</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2 B</td>
</tr>
<tr>
<td>21</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3 B</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2 B</td>
</tr>
<tr>
<td>23</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3 B</td>
</tr>
</tbody>
</table>
From this figure it is clear that the attribute LetterGrade is not discretized. So in a table the user can discretize only the necessary attributes. All the above techniques do not use any class information to do the discretization.

A.6 Data Management

In this study I tried to implement the data management at every stage by storing the data in different tables. Providing the statistics of the data at every stage, allowing the user to view and download the files in client side.

On the server side files cannot be modified, security is there at server side but client side any user can view and download the files, any user can modify the files, so the security feature is not supported on the client side. But if the user wants to verify the file which is on the client side to see if it’s modified or not, user can download the file from server. In this thesis I have displayed all the tables, and their type, and input table of that table, the techniques which are applied on that table are displayed.

So using that information user will be able to see which table to use for further processing and what table to use as input table.
<table>
<thead>
<tr>
<th>Type of the Table</th>
<th>Table Name</th>
<th>Base Table</th>
<th>Delete Table</th>
<th>Average Table</th>
<th>Custom Table</th>
<th>Equal Width Disc</th>
<th>Equal Frequency Disc</th>
<th>Custom Disc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Table</td>
<td>Grades</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Delete Missing Values</td>
<td>DeleteTable</td>
<td>Grades</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Replace Missing Values</td>
<td>ReplaceTable</td>
<td>Grades</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Customized Table</td>
<td>CustomTable</td>
<td>Grades</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Equal Width Discretization</td>
<td>EqualWidthTable</td>
<td>ReplaceTable</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Equal Frequency Discretization</td>
<td>EqualFrequencyTable</td>
<td>ReplaceTable</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Customized Discretization</td>
<td>CustomizedTable</td>
<td>DeleteTable</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

From the above figure, first column denotes the types of the tables, second column denotes name of the table, third is the base table (input table) for the table which is created, fourth column denotes if the table created was using the technique delete missing values or not, fifth column denotes if the table created was using the technique replace missing values or not, same with all the remaining columns.

According to the example, base table name is Grades, replace missing values table name is ReplaceTable, delete missing values table created was DeleteTable, customized
preprocess missing values table was CustomTable; for all these tables input table is Grades.

For equal width discretization technique and equal frequency discretization techniques, ReplaceTable is the input table and for customized discretization, DeleteTable is the input table.

From the above figure the user will be able to know what are the input tables and output tables, so user can easily select which table to use for further processing.