CSE 572 DATA MINING PROJECT REPORT

Ganesh Zilpe (1207578251)
Madhu Koteshwara Manjunath (1206283503)

1. Introduction

In today's world the amount of data that is being collected and stored in different forms across various databases has been increasing tremendously. Further, the stored data needs to be fetched at some point in time and needs to be processed accordingly. There are several techniques available in Data Mining that efficiently extract the data present in these databases and transform further transform them into a meaningful structure which could be useful in future for several purposes.

The Data Mining process has been used across several businesses across the world today. The advertisements that appear on the web browser are usually of the user's interest. This is known by collecting the data that is being browsed by the user in the past and accordingly the advertisements are set and displayed to the user. This is just one among several other applications of Data Mining.

Data Mining also has several applications across retail stores. In this process, data mining helps to know and learn the things that are being brought by a customer, the interest levels of customers as well as the items that are being sold frequently and so on. By learning these data items, a pattern could be derived and based on this pattern, the retail stores could concentrate more towards the items that are of higher interest to the customers.

2. Literature review

Data mining has been an ongoing process and a lot of research has been going on this field for the past few years. Almost every web page today has huge amounts of data in it and there are several transactions going on in many of these websites. All the information that is being recorded and collected in the form of data are stored in a database. It is a very tedious and difficult task to fetch the data from a huge database and this is where the data mining processes of classification, identification and recognition of patterns, associations come into picture. A Data Mining process basically involves information and Knowledge about the information. Information is nothing but the data that is being collected [1]. Further, the knowledge about the information is gained by learning the data using the different data mining processes.

Data preprocessing is the first process whenever any data is being collected from any source. In this preprocessing step of Data Mining, all the unwanted and useless data is removed that does not help in the knowledge process of the data. There are several forms of invalid data that may be useless in the form of duplicate data, missing data and unwanted data [2].

Classification of Data is the step that needs to be performed after the data is being preprocessed. There are several classification methods and algorithms being proposed and used over the years.

Classification is a technique in Data Mining used to predict the outcome of data based on previously available data. Data is being fed onto the classifier based on the training data for which the class labels are already known by the classifier. Once, this classifier is ready, the testing data is fed onto the classifier for the prediction of class labels. The accuracy is predicted based on how accurately the test labels are being predicted from the classifier.

There are certain challenges involved in collecting such huge amounts of data. Some of the key challenges to be highlighted are security challenges while handling confidential data. Another challenge would be the distribution of data. Since, the data grows very quickly in a very short span and since it is an ongoing process, it is a very hard task to bring all the data together and analyze them together.

3. Implementation methods

The learning algorithms that are used for classification are basically classified as **Supervised and unsupervised.** In this project we have chosen the supervised learning. This is because in the case of the unsupervised learning option, the samples that are being fed are not labeled and the hidden structure will have to be found in the data. [1]

The features were extracted from the given dataset which were provided from the training (there were 3400 features in the given training set), the testing dataset along with training label. Further, with respect to the model selection, we used several algorithms J48 Decision Trees, Naïve-Bayes and Support Vector machine (SMO).

A. J48 Decision Trees

A decision tree can be used to predict a value based on several attribute values of the training data. The attribute that is being predicted is known as the dependent variable and all the other attributes that contribute towards this process form the set of independent variables. The J48 tree is based on an algorithm that creates a decision tree based on the attribute values of the available training data. So when the training set is being passed, the attribute that discriminates the various instances most clearly is identified. The feature that is able to give most of the necessary information about the data is said to have the highest Information Gain. [1] The values for the feature are identified in such a way that there is no ambiguity, ie the data instances falling within its category have the same value for the target variable, the branch is terminated and the new value is assigned.

The top 48 features were chosen and the attribute selection was done based on Information gain and the accuracy during the Initial submission for the training set was 85.5% (correctly classified instances). The figure 1 below verifies this.

Further the percentage for the correctly classified instances for the test set was 85.89%.

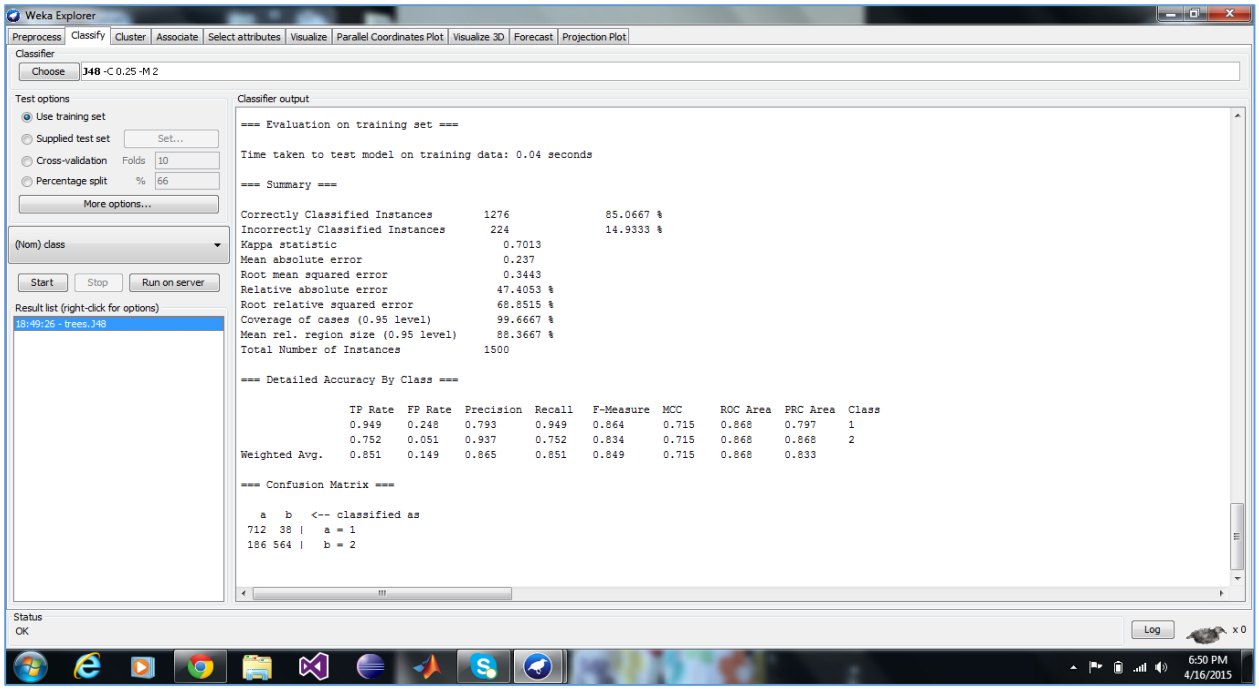

Figure 1: J48 Training set Correctly Classified Instances

B. Support Vector Machine (SVM) *(Sequential Minimal Optimization (SMO) algorithm)*

An SVM model is a special representation of data points such that the data points in different classes are divided by a clear decision boundary [3]. Testing labels are classified depending on which side of the decision boundary they fall into. They are a non-probabilistic binary linear classifier which constructs a hyper plane in a high dimensional space which can be further used for classification.

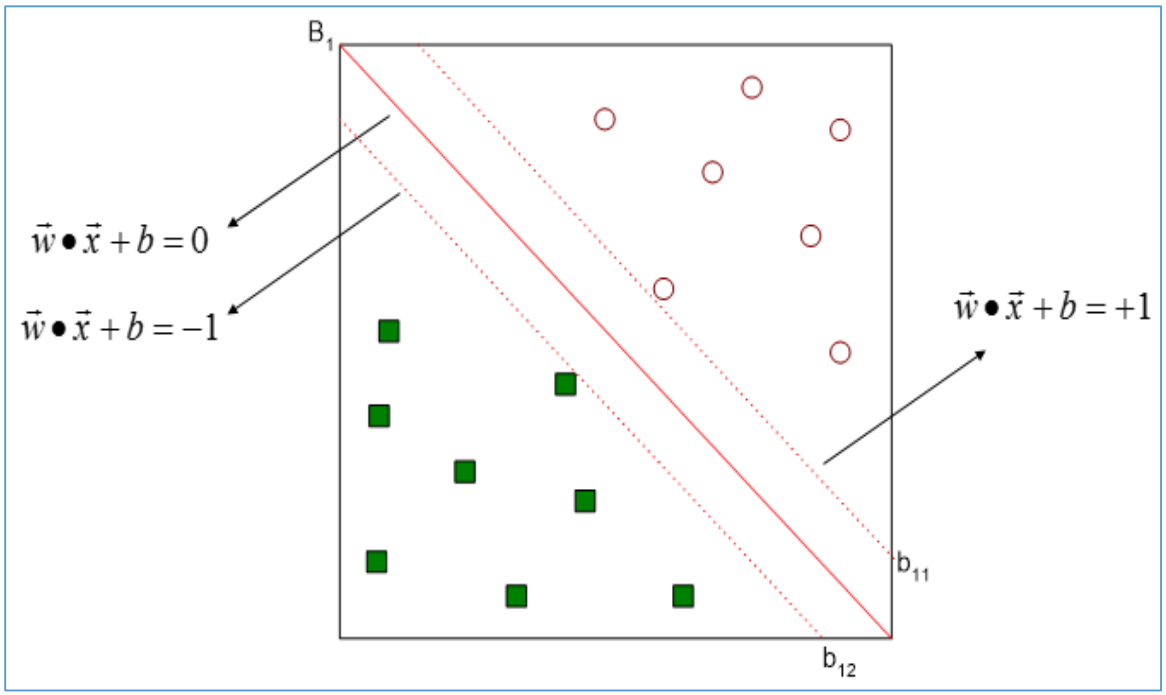

Figure 2: SVM Classification Model

The goal is to find the hyper-plane that maximizes the margin and separates both the classes. Maximizing the margin refers to maximizing,

$$\text{Margin} = \frac{2}{\| \vec{w} \|}$$

The sequential Minimal Optimization (SMO) algorithm for SVM was being used. The SMO algorithm basically breaks the initial problem into sets of smaller problems and then solves the smaller problems analytically. [4]

The top 48 features were chosen and the attribute selection was done based on Information gain and the accuracy during the second submission for the training set was 91.5% (correctly classified instances). The figure 3 below verifies this.

Further the percentage for the correctly classified instances for the test set was 86.6%.

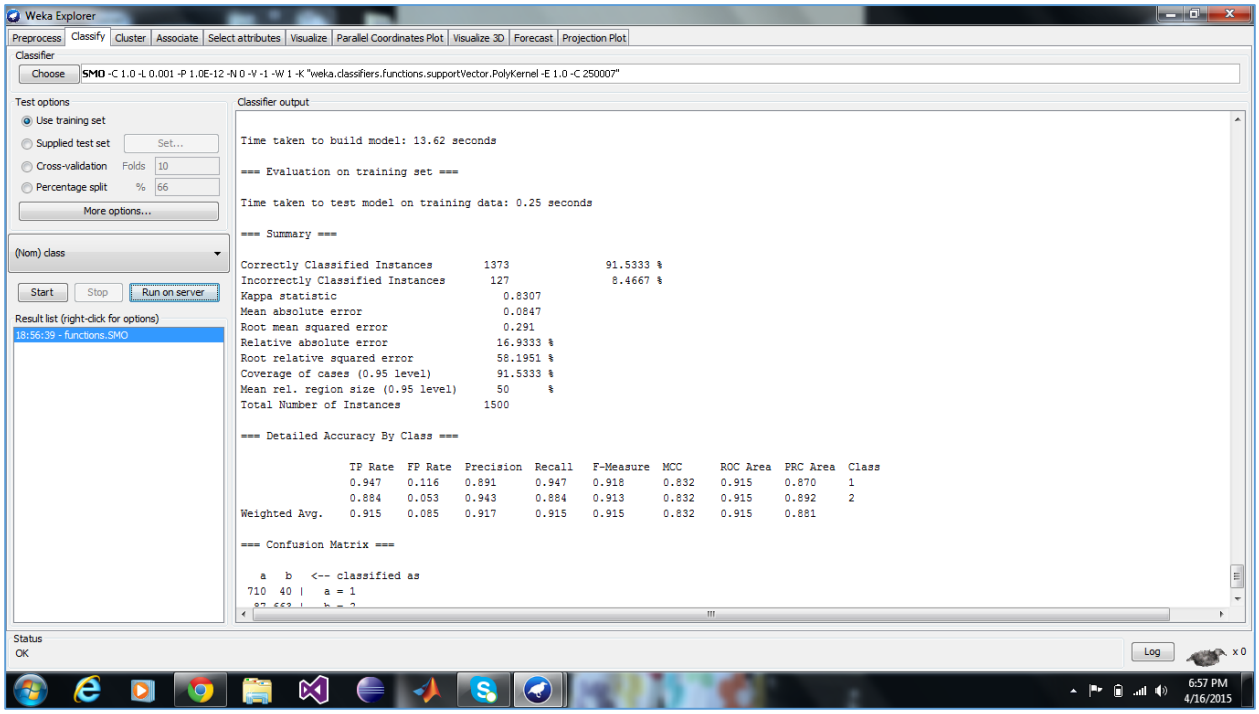

Figure 3: SVM Training set Corrctly Classified Instances

C. **Naïve Bayes**

Naïve Bayes is based on a set of supervised learning algorithms assuming that each and every pair of features is independent. Because of the assumption that all the attributes are independent, the parameters of each attribute can be learned separately. The advantage of using Naïve Bayes is that it requires very less amount of training data to identify the parameters required for classification [5]. Given a class variable C and vectors A1,….,An, Bayes theorem states the following model.

$$P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}$$

Figure 4: Bayes Classifier

The top 48 features were chosen and the attribute selection was done based on Information gain and the accuracy for the training set was 68.2% (correctly classified instances). Hence, we did not go with this approach and did not submit this result.

4. Format Unification of Training and Testing data set

For enabling more algorithms in WEKA software, we need to convert the numeric attribute into nominal. Otherwise, J48, SMO and other classifier are not available for numeric class variable. For this, we have used NumericToNominal preprocessing technique for converting all attributes from numeric to nominal of training set and testing set.

We have saved these file in ARFF format. After this, we have used these files in WEKA software. But, both files are not compatible with each other i.e. training and test data set were not compatible with each other. The reason was as follows:

Consider Feature 1 is having range of values 1-6 in the training data set. But, it is having different range say 4-8 in testing data set. Then the @attribute parameter in the ARFF file has having different values for training and testing data sets e.g. {1,3,4,5,6} and {4,5,6,7,8}. Manually unifying for all the attributes is very tedious and time consuming task. So, we have developed the Java program which unify the difference and give the correct result e.g. {1,3,4,5,6,7,8}.

Following is the Java Code:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Vector;

public class DataMining
{
	List<String> file1;
	List<String> file2;
	List<String> file3;
	private List<String> readFile(String filename)
	{
		List<String> records = new ArrayList<String>();
		try
		{
			BufferedReader reader = new BufferedReader(new
			FileReader(filename));
```

```
            String line;
            while ((line = reader.readLine()) != null)
            {
                  records.add(line);
            }
            reader.close();
            return records;
      }
      catch (Exception e)
      {
            System.err.format("Exception occurred trying to read
            '%s'.", filename);
            e.printStackTrace();
            return null;
      }
}
public void operation()
{
      String sample = "@attribute 'Feature ";
      for(int i=0; i<file1.size(); i++)
      {
            String[] fileinput1 = file1.get(i).split("\\{");
            String[] fileinput2 = file2.get(i).split("\\{");

            fileinput1 = fileinput1[1].split("}");
            fileinput2 = fileinput2[1].split("}");

            fileinput1 = fileinput1[0].split(",");
            fileinput2 = fileinput2[0].split(",");

            String result ="";
            Vector vect = new Vector();
            result = result + fileinput1[0];
            vect.add(fileinput1[0]);
            for(int l=1; l< fileinput1.length; l++)
            {
                  vect.add(fileinput1[l]);
                  result=result+","+fileinput1[l];
            }


            for(int k=0; k<fileinput2.length; k++)
            {
                  if(!(vect.contains((String)fileinput2[k])))
                  {
                        vect.add((String)fileinput2[k]);
                        result = result+","+fileinput2[k];
                  }
            }

            String []resultArray = null;
            if(result.contains(","))
            {
                  resultArray = result.split(",");
                  Arrays.sort(resultArray);
                  result ="";
                  if(resultArray[0]!="")
```

```
                        result = resultArray[0];
                    else
                        result = resultArray[1];
                    for(int m=1; m<resultArray.length; m++)
                        result = result + "," + resultArray[m];
                    System.out.println(sample+(file3.size()+1)+"'"+"
                    {"+result+"}");
                    file3.add(sample+(i+1)+"'"+" {"+result+"}");
                }
                else
                {
                    file3.add(sample+(i+1)+"'"+" {"+result+"}");
                }
            }
            FileWriter fw;
            try {
                fw = new FileWriter("H:\\Study\\MS_II\\Data
Mining\\Project\\3rd Submission\\Testing\\file3.txt");
                for (int i = 0; i < file3.size(); i++) {
                    fw.write(file3.get(i)+"\n");
                }
                fw.close();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        public static void main(String[] args)
        {
            DataMining dm = new DataMining();
            dm.file1 = dm.readFile("H:\\Study\\MS_II\\Data
Mining\\Project\\3rd Submission\\Testing\\file1.txt");
            dm.file2 = dm.readFile("H:\\Study\\MS_II\\Data
Mining\\Project\\3rd Submission\\Testing\\file2.txt");
            dm.file3 = new ArrayList<String>();
            dm.operation();
        }
}
```

5. Result Extraction from the classification models

The test data does not have class variable. We need to find class variable for test data set. But, there is no provision to output the class variable in the WEKA software. We employed two methods for getting the class variable from the classification model.

A. From the Java Program (Failed):

At this step, we already know that which classification we are going to use. We have written a Java program for creating specific classification model and then extract the class variable by evaluating the test data set against it. Though we were able to create classification model from training data, the code failed to extract the class variable for the test data. There is no problem with the code, but we could not able to identify the exact problem. Then, we used WEKA tool for the same. Following is the Java code:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;

import weka.classifiers.functions.SMO;
import weka.core.Instances;
public class Project
{
	public static void main(String[] args) {
		try
		{
		BufferedReader reader = new BufferedReader(new
FileReader("H:/Study/MS_II/Data Mining/Project/NewTesting2/train.arff"));
			Instances train;
			train = new Instances(reader);
			train.setClassIndex(0);//first attribute is class attribute

			reader = null;
			reader = new BufferedReader(new
			FileReader("H:/Study/MS_II/Data
			Mining/Project/NewTesting2/testWithEmptyClass.arff"));
			Instances test = new Instances(reader);
			test.setClassIndex(0);

			reader.close();

			SMO smoTree = new SMO(); //classifer

			smoTree.buildClassifier(train);
			Instances labeled = new Instances(test);

			//label instances
			for(int i=0; i<test.numInstances(); i++)
			{
				double classLabel =
				smoTree.classifyInstance(test.instance(i));
				labeled.instance(i).setClassValue(classLabel);
			}

			BufferedWriter writer = new BufferedWriter(new
			FileWriter("H:/Study/MS_II/Data
			Mining/Project/NewTesting2/testWithEmptyClass.arff"));
			writer.write(labeled.toString());
			System.out.println("Done");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}
}
```

B. From the WEKA tool (Successful):

Though there is no direct way (as per our study) to get the class variable for the test data set, there are some settings in the tool where we can see the result.

In the classify tab, there is button "More options..". On click, we get new window named "Classifier evaluations options". In this window, choose "Plain Text" as the "Output predictions". After evaluating test data set, you can see the class predictions for each test sample in the classifier output space.

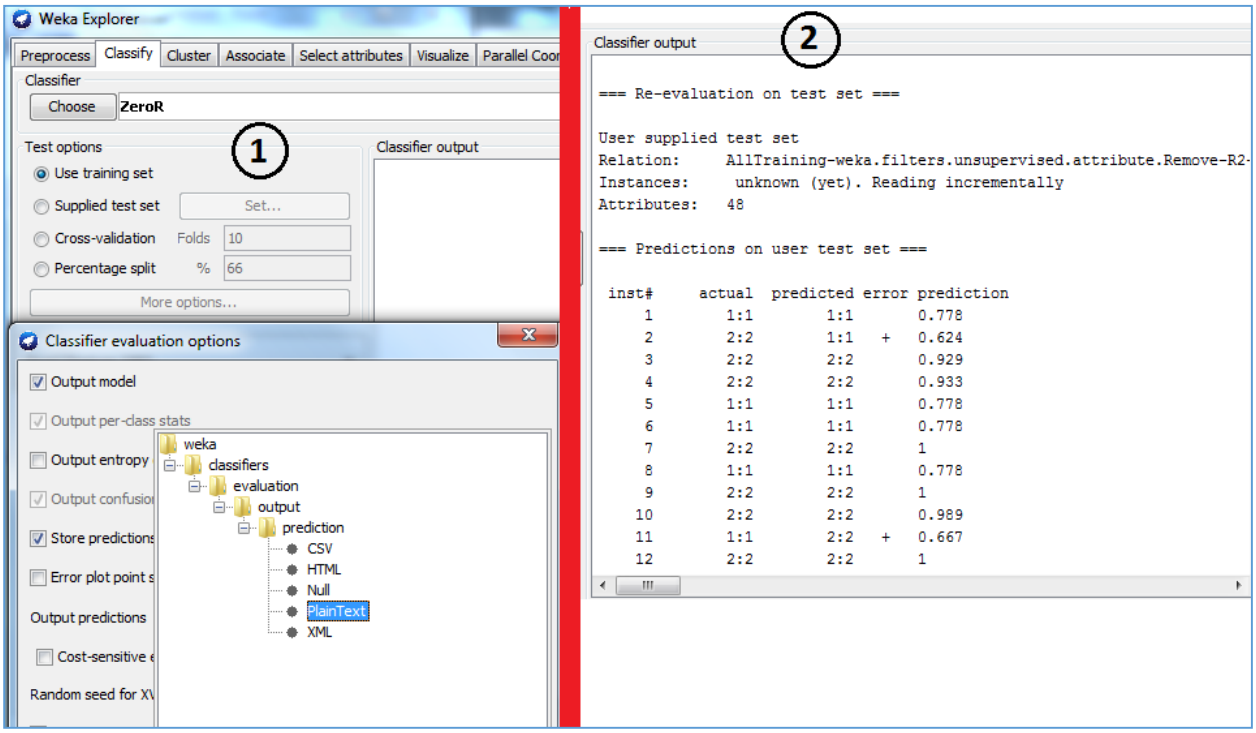

Figure 5: Extraction of class variable for test data set

6. Encountered problems and solutions

Problem: When the training set was loaded initially inside WEKA and we ran the classifiers, the accuracy was found to be really low a few times.
Solution: We found that the problem was with the features and we reduced the features and the size of the training data.

7. Conclusion

Several Data preprocessing techniques which included data reduction, data normalization and dimensionality reduction were identified. Further, hands on experience was obtained in the WEKA tool while implementing this project. We found that Support Vector Machine using the SMO algorithm was the best fit for our problem.

8. References

[1] http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html
[2] http://www.ercim.eu/publication/ws-proceedings/12th-EDRG/EDRG12_JeDiRe.pdf
[3] http://en.wikipedia.org/wiki/Support_vector_machine
[4] http://en.wikipedia.org/wiki/Sequential_minimal_optimization
[5]http://www.d.umn.edu/~padhy005/Chapter5.html