CHAPTER 1

INTRODUCTION

1.1. Background of the Study

In our daily life there is lots of records, phone call records, salary records, homework records, assignment record, personal details record, sales record, song, videos and so on. These all records kept in a table are called data; we have lots of data in different field.

Big data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process the data within a tolerable elapsed time. Big data sizes are a constantly moving target, as of 2012 ranging from a few dozen terabytes to many petabytes of data in a single data set. In a 2001 research report and related lectures, META Group (now Gartner) analyst Doug Laney defined data growth challenges and opportunities as being three-dimensional, i.e. increasing volume (amount of data), velocity (speed of data in and out), and variety (range of data types and sources). Gartner, and now much of the industry, continue to use this "3Vs" model for describing big data. In 2012, Gartner updated its definition as follows: "Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization."

Whenever there is data we can have lots of information, patterns, meaning etc. and the process of Extracting or "mining" knowledge from large amount of data is called Data mining [1]. Data mining also can be defined as Exploration and analysis of large quantities of data to discover meaningful pattern from data and is also known as "Knowledge discovery from data (KDD)" [1].

In data mining [1] there are lots of techniques to mine the knowledge from data which are recently used widely in different fields such as Business, Scientific Research, Computer Science, Machine Learning, Information Science, Statistics, and Database Technology etc. Most commonly used data mining techniques are **Classification**, **Regression, Clustering and Dependencies and Associations.**

Decision Tree is the most widely applied supervised classification technique. The learning and classification steps of decision tree induction are simple and fast and it can be applied to any domain [2].

1.2. Statement of Problem

Data mining applications has got rich focus due to its significance of classification algorithms. The comparison of classification algorithm is a complex and it is an open problem. First, the notion of the performance can be defined in many ways: accuracy, speed, cost, reliability, etc. Second, an appropriate tool is necessary to quantify this performance. Third, a consistent method must be selected to compare with the measured values. The selection of the best classification algorithm for a given dataset is a very widespread problem. In this sense it requires to make several methodological choices.

1.3. Objectives of the Study

The objectives of this research was:

- To compare the different decision tree algorithms (BFTree, J48, RandomTree, REPTree and SimpleCart).

1.4. Limitations of the Study

Limitations of my research were:

- This research had focused on comparison of Accuracy, Precision, Recall, F-measure and Tree Size of the implemented algorithms.
- All the algorithms had implemented in commonly used open source data mining tool Weka version 3.7.10.

1.5. Structure of the Report

This report is organized in three chapters including the following chapters.

- Chapter 1 ***"Introduction"*** explains the Background of the research, Statement of problems, Objectives of the study as well as Limitations of the study.
- Chapter 2 ***"Literature Review"*** describes the various concepts of Data mining, decision tree and related works in our domain.
- Chapter 3 ***"Research Methodology"*** explains the framework and algorithms.

- Chapter 4 ***"Experiment and Result Analysis"*** explains about experiments, results and context analysis.
- Chapter 5 ***"Conclusion and Future Works"*** explains the conclusion and future direction of research.
- ***References***
- ***Appendix***

CHAPTER 2

LITERATURE REVIEW

2.1. Data Mining

Various data when processed can provide lots of information, patterns, meaning etc. and the process of Extracting or "mining" knowledge from large amount of data is called Data mining [1]. Data mining also can be defined as Exploration and analysis of large quantities of data to discover meaningful pattern from data and is also known as "Knowledge discovery from data (KDD)" [1].

In data mining [1] there are lots of techniques to mine the knowledge from data which are recently used widely in different fields such as Business, Scientific Research, Computer Science, Machine Learning, Information Science, Statistics, Database Technology etc. Most commonly used data mining techniques are **Classification**, **Regression, Clustering and Dependencies and Associations.** These above mentioned techniques are effective in different field separately.

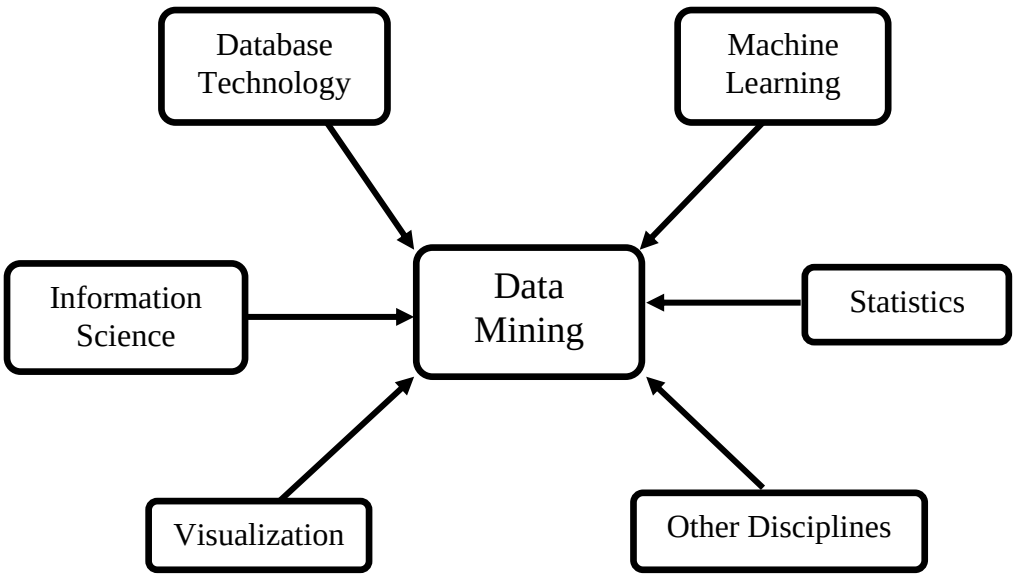


Figure 2.1: Data mining as confluence of multiple disciplines

2.1.1. Classification

Classification is a data mining technique used to predict the category of categorical data by building a model based on some predictor variables (to classify data). Predictor variable/attribute is called class label attribute (predefined class). Since the training data are accompanied by labels indicating the class of the observations it is also called supervised learning and new data is classified based on the training set.

It is a two-step process

1. Model Construction (Learning step or Training Phase)

   - Build a model to explain the target concept

   - Model is represented as classification rules, decision trees, or mathematical formulae

2. Model Usage (Testing Phase)

   - is used for classifying future or unknown cases

   - estimate the accuracy of the model

A. Decision Tree

A decision tree is a flowchart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node. A typical decision tree is shown in *figure 2.2* [1].

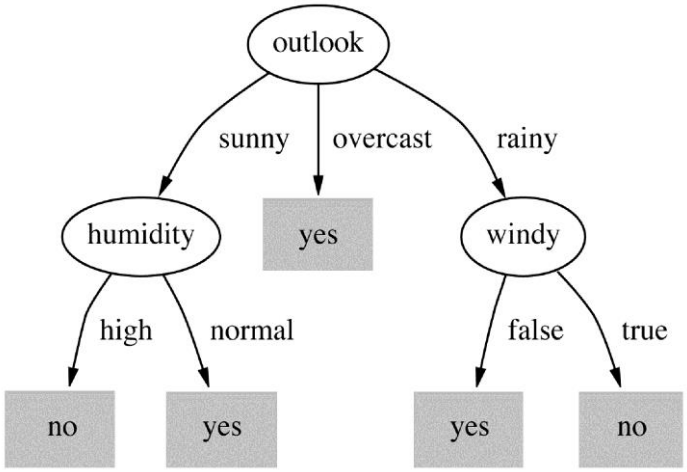


Figure 2.2: Decision tree example [1]

During the late 1970s and early 1980 J. Ross Quinlan, a researcher in machine learning, developed a decision tree algorithm known as **ID3** (Iterative Dichotomiser). This work expanded on earlier work on concept learning system, described by E. B Hunt, J. Marin, and P. T. Stone. Quinlan later presented C4.5 [3] [4] (a successor of ID3), which become a benchmark to which newer supervised learning algorithms are often compared. In 1984, a group of statisticians published the book classification and regression trees (CART) [4], which described the generation of binary decision trees. ID3 and CART were invented independently of one another at around the same time, yet follow a similar approach for learning decision trees from training tuples. These two cornerstone algorithms spawned a flurry of work on decision tree induction. The basic decision tree algorithm is summarized as below:

- **Decision Tree Construction Algorithm**

Input: A data set, D

Output: A decision tree

- If all the instances have the same value for the target attribute then return a decision tree that is simply this value (not really a tree - more of a stump).
- Else

1. Compute Gain values for all attributes and select an attribute with the highest value and create a node for that attribute.
2. Make a branch from this node for every value of the attribute
3. Assign all possible values of the attribute to branches.
4. Follow each branch by partitioning the dataset to be only instances whereby the value of the branch is present and then go back to 1.

- **Attribute Selection Measures**

In a data set there are lots of attributes and we do have problem on selection of attribute as node and as leaf. There arise questions **which attribute first?**

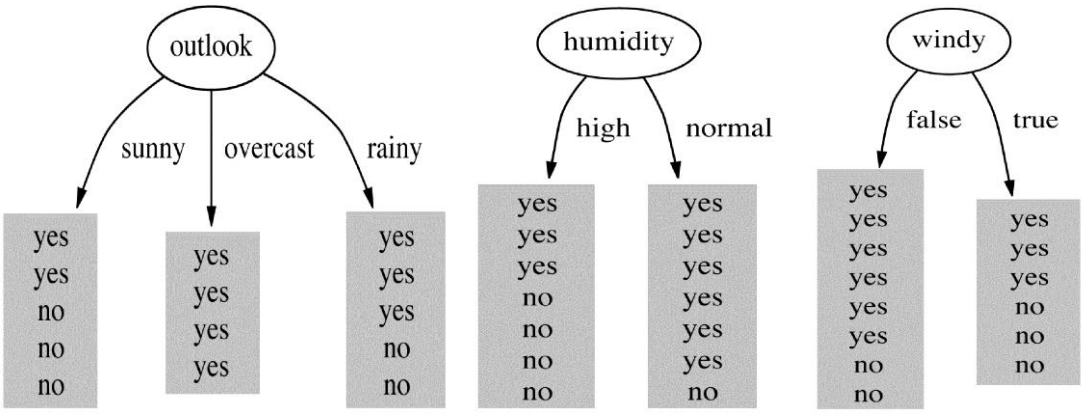


Figure 2.3: Possibility of attribute as node [1]

Attribute selection measure [1] is a heuristic for selecting the splitting criterion that "best" separates given data partition, D, of class-labeled training tuples into individual classes. Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split. The attribute selection measure provides a ranking for each attribute describing the given training tuples. The attribute having the best score for the measure is chosen as the splitting attribute for the given tuples.

- **Information Gain**

ID3 uses information gain as its attribute selection measure. This measure is based on pioneering work by Claude Shannon on information theory, which studied the value or "information content" of messages [1].

Information gain = (information before split) – (information after split) bits

$$Gain(A) = Info(D) - Info_A(D)\ bits$$ ----------Equation 2.1

Where,

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)\, bits$$ ----------Equation 2.2

- $P_i = |Ci,D| / |D|$
- A having v distinct values, $\{a_1, a_2, \ldots, a_v\}$
- $D_1, D_2, \ldots, D_v$ then,

$$Info_A(D) = \sum_{j=1}^{v} \frac{|Dj|}{|D|} \times Info(Dj)\, bits$$ ----------Equation 2.3

- **Gain Ratio:**

The information gain [1] measure is biased toward tests with many outcomes. That is, it prefers to select attributes having a large number of values. C4.5[3] [4], a successor of ID3, uses an extension to information gain known as **gain ratio**, which attempts a overcome this bias. It applies a kind of normalization to information gain using a "Split information" value define analogously with $Info(D)$ as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|Dj|}{|D|} \times \log_2\left(\frac{|Dj|}{|D|}\right) bits$$ ----------Equation 2.4

The gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$ ----------Equation 2.5

The attribute with the maximum gain ratio is selected as the splitting attribute. Note, however, that are the split information approaches 0, the ratio becomes unstable. A constraint is added to avoid this, whereby the information gain of the test selected must be large-at last as great as the average gain over all tests examined.

- **Gini Index**

The Gini index [1] is used in CART [4]. Using the notation previously described, the Gini index measures the impurity of D, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2\, bits$$ ----------Equation 2.6

Where, P_i is the probability that a tuple in D belong to class C_i and is estimated by $|C_{i,D}| / |D|$. The sum is computed over m classes. The Gini index considers a binary split for each attribute. Let's first consider the case where A is a discrete-valued attribute having v distinct values, {a1, a2, ..., av}, occurring in D. If A has v possible values, then there are 2^v possible subsets but we exclude the power set, and the empty set from consideration since, conceptually, they do not represent a split. Therefore there are 2^v-2 possible ways to form two partitions of the data, D, based on a binary split on A.

When considering split, we compute a weighted sum of the impurity of each resulting partition. For example, if a binary split on A partitions D into D_1 and D_2, the Gini index of D given that partitioning is

$$Gini_A(D) = \left\{ \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \right\} bits$$ ----------Equation 2.7

For each attribute, each of these possible binary splits is considered. For discrete-valued attribute, the subset that gives the minimum Gini index for that attribute is selected as its splitting subset.

The reduction in impurity that would be incurred by a binary split on a discrete-or continuous-valued attribute A is

$$\Delta Gini(A) = \{Gini(D) - Gini_A(D)\} bits$$ ----------Equation 2.8

The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute.

2.2. Application Programming Interface (API) for Data Mining

Application Programming Interface (API) [5] is a set of routines used by an application program to direct the performance of procedures by the computer's

operating system. To achieve our goal we will be using most commonly used API for data mining which is Weka.

2.2.1. WEKA

Weka [6] is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes [6].

Weka was developed at the University of Waikato in New Zealand; the name stands for *Waikato Environment for Knowledge Analysis*. (Outside the university, the Weka, pronounced to rhyme with *Mecca*, is a flightless bird with an inquisitive nature found only on the islands of New Zeland) The system is written in Java and distributed under the terms of the General Public License (GNU). It runs in almost any platform and has been tested under Linux, Windows and Macintosh operating systems- and even on a personal digital assistant [7]. Weka's native data storage method is *Attribute-Relation File Format* (ARFF) [8].

2.3. Related Works

The research [2] compares the performance of ID3, C4.5 and CART algorithms. Study was conducted with student's qualitative data to know the influence of qualitative data in student's performance using decision tree algorithms. The results showed that the CART had the best classification accuracy when compared to ID3 and C4.5.

In [9], four decision tree algorithms (J48, RandomForest, RandomTree and REPTree) were applied on the mobile CDR dataset for analyzing the usage of mobile services. The result indicates that J48 decision classifier outperforms other classifiers, whereas RT (random tree) classifier consumes less time.

According to the results of Experiment: [10] First, with the same size cache, S2.1 algorithm had higher hit rate than S2 algorithm, and even with the same hit rate, S2.1 algorithm is in responses to the client URL requested faster than S2 algorithm, because of the decision tree of GATree algorithm was far smaller than C4.5 algorithm.

In [11], the study of authors proved that the classification of Web object through log mining by using CART, Multivariate Adaptive Regression Splines (MARS), Random Forest (RF) and TreeNet (TN) models can be applied in cache server.

In [12] [13] experiment had explored how GAs can be used to directly evolve decision trees and the experiments were compared the results of C4.5, OneR and GATree.

CHAPTER 3

RESEARCH METHODOLOGY

3.1. Background

This chapter deals with the framework of research and used algorithms.

3.2. Source of Data

In this research Source of data were secondary source and all the data sets were downloaded from data mining data repository; University of California machine learning repository (UCI Repository[1]).

3.3. Algorithms

In this research, five decision tree algorithms were compared and they are

a) BFTree b) J48 c) RandomTree

d) REPTree e) SimpleCart

3.3.1. BFTree

BFTree [7] constructs a decision tree using a best-first expansion of nodes rather than the depth-first expansion used by standard decision tree learners (such as C4.5 [3]). Pre- and post-pruning options are available with the based on finding the best number of expansions to use via cross-validation on the training data. While fully grown trees are the same for best-first and depth-first algorithms, the pruning mechanism used by BFTree will yield a different pruned tree structure than that produced by depth-first methods.

1. http://archive.ics.uci.edu/ml/datasets.html

Best-first algorithm

OPEN = [initial state]

While OPEN is not empty or until a goal is found

Do

1. Remove the best node from OPEN, call it n.
2. If n is the goal state, backtrace path to n (through recorded parents) and return path.
3. Create n's successors.
4. Evaluate each successor, add it to OPEN, and record its parent.

Done

3.3.2. J48

J48 is a Weka implementation of C4.5 [3] decision tree classifier algorithm, which uses Gain ratio *(see equation 2.5 in page number 7)* for the attribute selection.

- **Gain Ratio:**

The information gain [1] measure is biased toward tests with many outcomes. That is, it prefers to select attributes having a large number of values. C4.5 uses an extension to information gain known as **gain ratio**, which attempts a overcome this bias. It applies a kind of normalization to information gain using a "Split information" value define analogously with $Info(D)$ as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|Dj|}{|D|} \times \log_2\left(\frac{|Dj|}{|D|}\right) bits$$ ----------Equation 3.1

The gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$ ----------Equation 3.2

The attribute with the maximum gain ratio is selected as the splitting attribute. Note, however, that are the split information approaches 0, the ratio becomes unstable. A

constraint is added to avoid this, whereby the information gain of the test selected must be large-at last as great as the average gain over all tests examined.

Pseudocode

1. Check for base cases
2. For each attribute *a*
    - Find the normalized information gain ratio from splitting on *a*
3. Let *a_best* be the attribute with the highest normalized information gain
4. Create a decision *node* that splits on *a_best*
5. Recurse on the sublists obtained by splitting on *a_best*, and add those nodes as children of *node*

Algorithm

Input: A data set, D

Output: A decision tree by C4.5 (J48)

- If all the instances have the same value for the target attribute then return a decision tree that is simply this value (not really a tree - more of a stump).
- Else
    1. Compute Gain ratios for all attributes and select an attribute with the highest value and create a node for that attribute.
    2. Make a branch from this node for every value of the attribute
    3. Assign all possible values of the attribute to branches.
    4. Follow each branch by partitioning the dataset to be only instances whereby the value of the branch is present and then go back to 1.

3.3.3. RandomTree

Tree built by RandomTree [7] test a given number of random features at each node, performing no pruning. Also has an option to allow estimation of class probabilities based on a hold-out set (back fitting).

Back fitting

Additive models are a class of non-parametric regression models of the form:

$$Y_i = \alpha + \sum_{j=1}^{p} f_j(X_{ij}) + \epsilon_i$$

where each $X_1, X_2, \ldots, X_p$ is a variable in our p-dimensional predictor X, and Y is our outcome variable. ϵ represents our inherent error, which is assumed to have mean zero. The $f_j$represent unspecified smooth functions of a singleX_j. Given the flexibility in the f_j, we typically do not have a unique solution: α is left unidentifiable as one can add any constants to any of thef_j and subtract this value from α. It is common to rectify this by constraining

$$\sum_{i=1}^{N} f_j(X_{ij}) = 0 \quad \text{for all } j$$

Leaving

$$\alpha = 1/N \sum_{i=1}^{N} y_i$$

3.3.4. REPTree

REPTree [7] builds a decision or regression tree using information gain/ variance reduction and prunes it using reduced-error pruning. Optimized for speed, it only sorts valued for numeric attributes once. It deals with missing values by splitting instances into pieces, as C4.5 does. We can set the minimum number of instances per leaf, maximum tree depth (useful when boosting trees), minimum proportion of training set variance for a split (numeric classes only), and number of folds for pruning.

Information gain = (information before split) – (information after split) bits

$$Gain(A) = Info(D) - Info_A(D)\ bits$$ ----------Equation 3.3

Where,

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) bits$$ ----------Equation 3.4

- P_i = $|Ci,D| / |D|$
- A having v distinct values, $\{a_1, a_2, \ldots, a_v\}$
- $D_1, D_2, \ldots, D_v$ then,

$$Info_A(D) = \sum_{j=1}^{v} \frac{|Dj|}{|D|} \times Info(Dj)\ bits$$ ----------Equation 2.3

3.3.5. SimpleCart

SimpleCart is a Weka implementation of CART [4] (Classification and Regression Tree) algorithm, which uses Gini index as attribute selection metric and employs the minimal cost-complexity pruning strategy [7].

- **Gini Index**

The Gini index [1] measures the impurity of D, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2\ bits$$ ----------Equation 3.5

Where, P_i is the probability that a tuple in D belong to class C_i and is estimated by $|C_{i,D}| / |D|$. The sum is computed over m classes. The Gini index considers a binary split for each attribute. Let's first consider the case where A is a discrete-valued attribute having v distinct values, {a1, a2, ..., av}, occurring in D. If A has v possible values, then there are 2^v possible subsets but we exclude the power set, and the empty set from consideration since, conceptually, they do not represent a split. Therefore there are 2^v-2 possible ways to form two partitions of the data, D, based on a binary split on A.

When considering split, we compute a weighted sum of the impurity of each resulting partition. For example, if a binary split on A partitions D into D_1 and D_2, the Gini index of D given that partitioning is

$$Gini_A(D) = \left\{ \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2) \right\} bits$$ ----------Equation 3.6

For each attribute, each of these possible binary splits is considered. For discrete-valued attribute, the subset that gives the minimum Gini index for that attribute is selected as its splitting subset.

The reduction in impurity that would be incurred by a binary split on a discrete-or continuous-valued attribute A is

$$\Delta Gini(A) = \{Gini(D) - Gini_A(D)\} bits$$ ----------Equation 3.7

The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute.

Algorithm

1. Establish Classification Attribute (in Table D)
2. Compute Gini index for all the attributes of the table.
3. Select Attribute with the minimum Gini Index to be the next Node in the tree (starting from the Root node).
4. Remove Node Attribute, creating reduced table (RD).
5. Repeat steps 3-5 until all attributes have been used, or the same classification value remains for all rows in the reduced table.

3.4. Experiments Protocol

Experiments were done by using famous open source data mining tool Weka v3.7.10.

3.4.1. Validation

For the validation of the analysis of implemented algorithms following evaluation metrics were compared.

- **5-fold cross-validation**
- **Accuracy**
- **Precision**
- **Recall**
- **F-measure**
- **Tree Size**

3.4.2. Experimental Setup

Scheme1: weka.classifiers.trees.BFTree -S 1 -M 2 -N 5 -C 1.0 -P POSTPRUNED *Scheme2: weka.classifiers.trees.J48 -C 0.25 -M 2* *Scheme3: weka.classifiers.trees.RandomTree -K 0 -M 1.0 -S 1* *Scheme4: weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1* *Scheme5: weka.classifiers.trees.SimpleCart -S 1 -M 2.0 -N 5 -C 1.0*
Relation: Different Datasets (.arff format)
Test mode: 5-fold cross-validation

Table 3.1: Experimental Parameters

CHAPTER 4

EXPERIMENT AND RESULT ANALYSIS

4.1. Background

This section deals with the successful implementation and *Comparative analysis of decision tree Classification algorithms*. The experiments were performed in famous data mining tool, Intel® core™ i3 CPU 3110 M @ 2.40GHz 2.40GHz with 4.00 GB RAM in 32 bit Windows 7 Ultimate Operating System.

4.2. Tool

The system can be implemented using data mining tools. In this research, Weka v3.7.10 was used.

Weka [6] is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes [6].

Weka was developed at the University of Waikato in New Zealand; the name stands for *Waikato Environment for Knowledge Analysis*. (Outside the university, the Weka, pronounced to rhyme with *Mecca*, is a flightless bird with an inquisitive nature found only on the islands of New Zeland.) The system is written in Java and distributed under the terms of the General Public License (GNU). It runs in almost any platform and has been tested under Linux, Windows and Macintosh operating systems- and even on a personal digital assistant [7]. Weka's native data storage method is *Attribute-Relation File Format* (ARFF) [8].

4.3. List of data

The main data used in this study are enlisted below:

SNO	Relation	Instances
1	Anneal	898
2	Audiology	226
3	Balance-scale	625
4	Breast-cancer	286
5	Credit-g	1000
6	Diabetes	768
7	Glass	214
8	Heart-statlog	270
9	Irish	150
10	Labor	57
11	Lymph	148
12	Multiplexor	100
13	Mushroom	8124
14	Soybean	39
15	Students	498
16	Vote	435
17	Zoo	101

Table 4.1: List of Data

4.4. Experiments and Results

In this section, each steps of the methodology was implemented for simulation and results were described. For the evaluation followings are evaluation metrics.

4.4.1. 5-fold cross-Validation

In **5-fold cross-validation**, the initial data are randomly partitioned into 5 mutually exclusive subsets or "folds" i.e. D_1, D_2, D_3, D_4 and D_5 each of approximately equal size. Training and testing is performed 5 times in the ratio of 4:1 means to say 4 fold as Training and 1 fold as Testing

4.4.2. Confusion Matrix

A confusion matrix is a table for analyzing the result of the classifiers. It deals with how classifier can recognize tuples of different classes. In order to develop the confusion matrix, the following terms should be considered:

- **True Positive (TP):** Positive tuples that are correctively labelled by the classifier.
- **True Negative (TN):** Negative tuples that are correctly labelled by the classifier.
- **False Positive (FP):** Negative tuples that are incorrectly labelled as positive.
- **False Negative (FN):** Positive tuples that are mislabelled as negative.

		Predicted Class		
		Yes	No	Total
Actual Class	Yes	**TP**	**FN**	**P**
	No	**FP**	**TN**	**N**
	Total	**P′**	**N′**	**P+N**

Table 4.2: Confusion Matrix

4.4.3. Accuracy

Accuracy of a classifiers on a given test set is the percentage of test set tuples that are correctly classified by the classifiers. It also refers to the recognition rate of the classifier that means how the classifier recognizes tuples of the various classes.

$$\text{Accuracy} = \frac{TP + TN}{P + N}$$ ---------- **Equation 4.1**

		Accuracy (%)				
SNO	**Relation**	**BFTree**	**J48**	**RandomTree**	**REPTree**	**SimpleCart**
1	Anneal	95.77	95.88	94.54	93.21	95.66
2	Audiology	73.45	76.55	67.26	69.03	75.22
3	Balance-scale	77.76	77.60	76.64	77.44	78.08
4	Breast-cancer	6s7.83	74.13	65.38	67.48	68.53
5	Credit-g	72.20	73.30	65.70	71.70	73.70
6	Diabetes	73.18	71.22	72.53	72.92	73.57
7	Glass	71.50	65.42	66.82	66.36	72.43
8	Heart-statlog	76.67	77.78	73.33	80.00	77.41
9	Irish	96.00	96.00	95.33	94.67	95.33
10	Labor	75.44	77.19	85.96	64.91	75.44
11	Lymph	79.73	80.41	77.03	75.68	79.73
12	Multiplexor	57.00	63.00	67.00	57.00	57.00
13	Mushroom	99.94	100.00	100.00	99.95	99.94
14	Soybean	94.87	97.44	94.87	89.74	94.87
15	Students	83.53	82.13	80.12	81.33	82.73
16	Vote	95.17	96.55	94.48	95.63	95.63
17	Zoo	40.59	92.08	69.31	40.59	40.59
	Average	78.27	82.16	79.19	76.33	78.58

Table 4.3: Result taking Accuracy

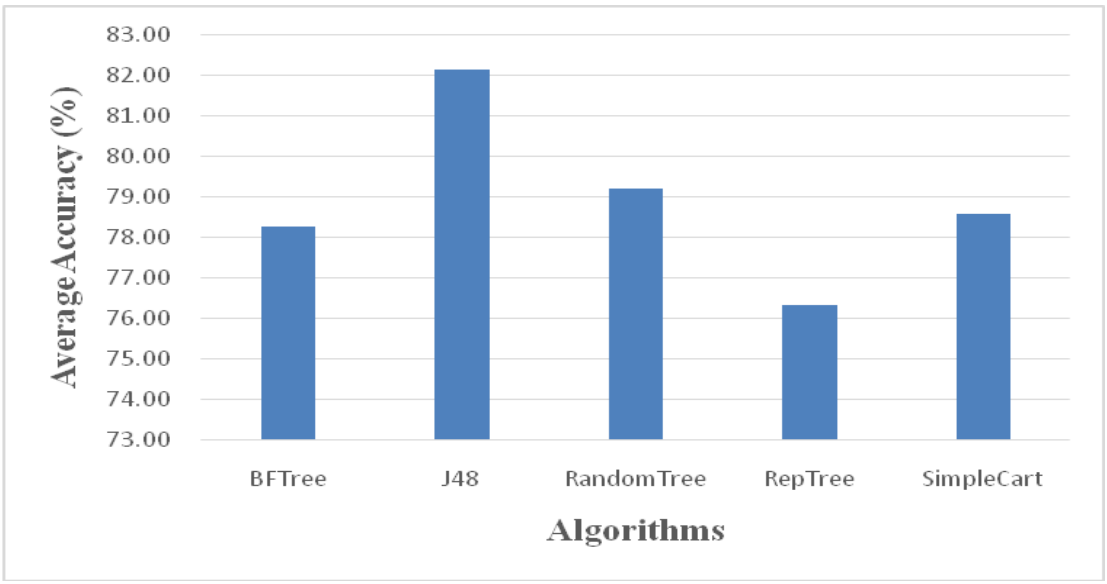


Figure 4.1: Graph of table 4.3

4.4.4. Precision

Precision refers to the measure of exactness that means what percentage of tuples labeled as positive are actually such.

$$\text{Precision} = \frac{TP}{TP + FP} \text{----------Equation 4.2}$$

		Precision (%)				
SNO	**Relation**	**BFTree**	**J48**	**RandomTree**	**REPTree**	**SimpleCart**
1	Anneal	95.40	95.60	94.50	92.50	95.70
2	Audiology	70.10	73.20	65.10	65.20	69.90
3	Balance-scale	78.10	74.60	80.10	73.40	74.80
4	Breast-cancer	62.80	73.30	62.60	62.20	63.70
5	Credit-g	70.20	72.00	66.20	69.30	72.20
6	Diabetes	72.60	71.50	72.90	72.10	72.10
7	Glass	70.70	66.40	67.70	65.20	65.20
8	Heart-statlog	76.60	77.70	73.20	80.10	77.40
9	Irish	96.00	96.00	95.30	94.70	95.30
10	Labor	75.00	77.00	85.80	66.70	75.00
11	Lymph	80.00	80.50	76.10	74.30	78.20
12	Multiplexor	85.20	82.20	80.10	82.70	84.20
13	Mushroom	55.50	62.60	66.60	54.80	55.50
14	Soybean	99.90	100.00	100.00	100.00	99.90
15	Students	95.10	97.70	95.10	91.10	95.10
16	Vote	95.20	96.60	94.50	95.70	95.70
17	Zoo	16.50	92.20	70.10	16.50	16.50
	Average	76.17	81.71	79.17	73.91	75.67

Table 4.4: Result taking Precision

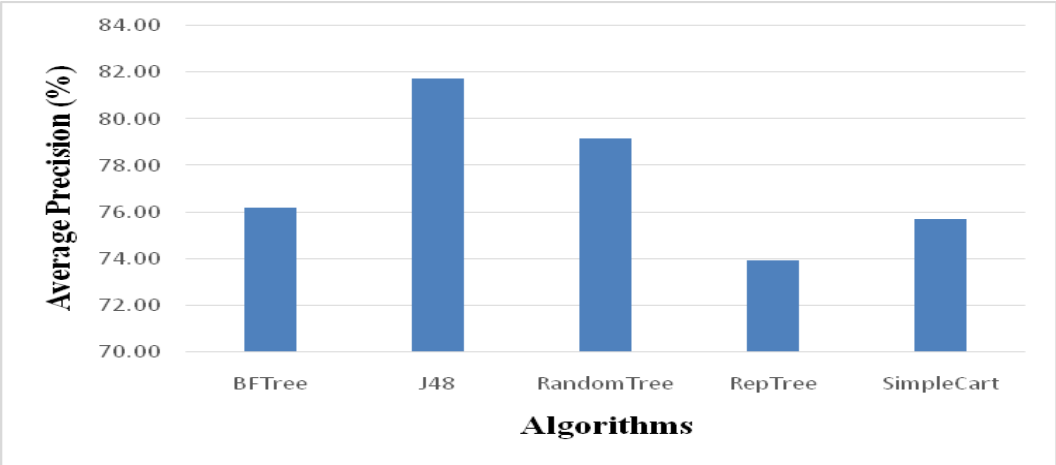


Figure 4.2: Graph of table 4.4

4.4.5. Recall

Recall refers to the true positive rate that means the proportion of positive tuples that are correctly identified. It is also known as sensitivity of the classifier.

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{TP}{P} \text{ ----------Equation 4.3}$$

		Recall (%)				
SNO	**Relation**	**BFTree**	**J48**	**RandomTree**	**REPTree**	**SimpleCart**
1	Anneal	95.80	95.90	94.50	93.20	95.70
2	Audiology	73.50	76.50	67.30	69.00	75.20
3	Balance-scale	77.80	77.60	76.60	77.40	78.10
4	Breast-cancer	67.80	74.10	65.40	67.50	68.50
5	Credit-g	72.20	73.30	65.70	71.70	73.70
6	Diabetes	73.20	71.20	72.50	72.90	72.90
7	Glass	71.50	65.40	66.80	66.40	66.40
8	Heart-statlog	76.70	77.80	73.30	80.00	77.40
9	Irish	96.00	96.00	95.30	94.70	95.30
10	Labor	75.40	77.20	86.00	64.90	75.40
11	Lymph	79.70	80.40	77.00	75.70	79.70
12	Multiplexor	83.50	82.10	80.10	81.30	82.70
13	Mushroom	57.00	63.00	67.00	57.00	57.00
14	Soybean	99.90	100.00	100.00	100.00	99.90
15	Students	94.90	97.40	94.90	89.70	94.90
16	Vote	95.20	96.60	94.50	95.60	95.60
17	Zoo	40.60	92.10	69.30	40.60	40.60
	Average	78.28	82.15	79.19	76.33	78.18

Table 4.5: Result taking Recall

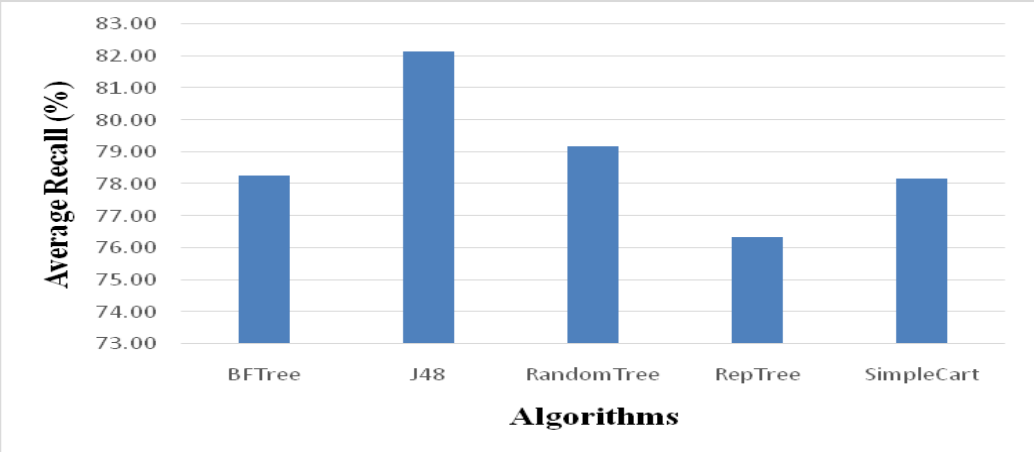


Figure 4.3: Graph of table 4.5

4.4.6. F-Measure

The F-score or F-Measure also refers to F-measures combines the both the measures Precision and Recall as the harmonic mean

$$F-Measure = \frac{2 \times Precision \times Recall}{Precesion + Recall}$$ ----------Equation 4.4

		F-Measure (%)				
SNO	**Relation**	**BFTree**	**J48**	**RandomTree**	**REPTree**	**SimpleCart**
1	Anneal	95.50	95.60	94.50	92.70	95.70
2	Audiology	71.30	74.50	65.00	65.20	72.00
3	Balance-scale	77.90	76.10	78.30	74.90	76.40
4	Breast-cancer	63.50	69.10	63.60	63.00	64.00
5	Credit-g	70.40	72.40	65.90	69.40	72.50
6	Diabetes	72.80	71.30	72.70	72.20	72.20
7	Glass	70.60	65.70	67.10	65.30	65.30
8	Heart-statlog	76.70	73.20	77.70	79.80	77.20
9	Irish	96.00	96.00	95.30	94.70	95.30
10	Labor	75.10	77.10	85.80	65.50	75.10
11	Lymph	79.40	80.40	76.40	74.90	78.50
12	Multiplexor	83.50	82.10	80.10	81.30	82.70
13	Mushroom	55.80	62.70	66.85	54.96	5.80
14	Soybean	99.90	100.00	100.00	100.00	99.90
15	Students	94.90	97.40	94.90	89.40	94.90
16	Vote	95.20	96.60	94.50	95.60	95.60
17	Zoo	23.40	92.00	66.31	23.40	23.40
	Average	76.58	81.31	79.12	74.25	73.32

Table 4.6: Result taking F-Measure

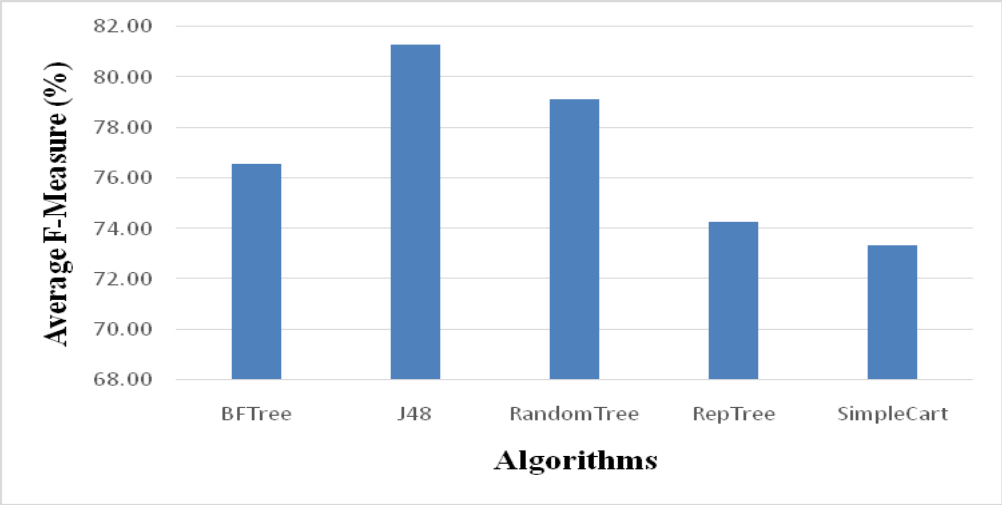


Figure 4.4: Graph of table 4.6

4.4.7. TreeSize

		Tree Size				
SNO	**Relation**	**BFTree**	**J48**	**RandomTree**	**REPTree**	**SimpleCart**
1	Anneal	49	92	278	64	51
2	Audiology	43	54	365	28	43
3	Balance-scale	161	103	349	59	25
4	Breast-cancer	05	06	444	32	01
5	Credit-g	77	140	1073	96	13
6	Diabetes	05	39	271	49	05
7	Glass	21	59	97	19	15
8	Heart-statlog	41	35	131	11	31
9	Irish	11	09	17	05	09
10	Labor	13	05	30	09	03
11	Lymph	17	34	134	17	17
12	Multiplexor	25	23	85	15	19
13	Mushroom	13	30	169	38	13
14	Soybean	07	08	42	05	07
15	Students	77	35	153	21	29
16	Vote	71	11	143	3	11
17	Zoo	01	17	101	01	01
	Average	37.47	41.18	228.35	27.76	17.24

Table 4.7: Result taking Tree Size

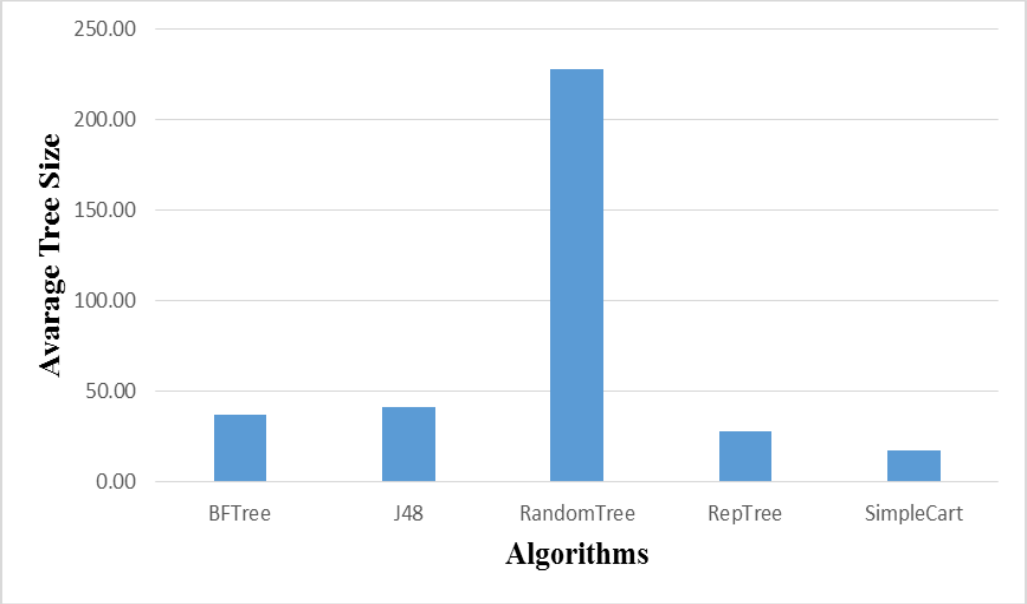


Figure 4.5: Graph of table 4.7

4.4.8. Averages of Evaluation Metrics

Evaluation Metric / Algorithms	Accuracy	Precision	Recall	F-Measure	TreeSize
BFTree	78.27	76.17	78.28	76.58	37.47
J48	82.16	81.71	82.15	81.31	41.18
RamdomTree	79.19	79.17	79.19	79.12	228.35
REPTree	76.33	73.91	76.33	74.25	27.76
SimpleCart	78.58	75.67	78.18	73.32	17.24

Table 4.8: Result taking averages of Evaluation Metrics

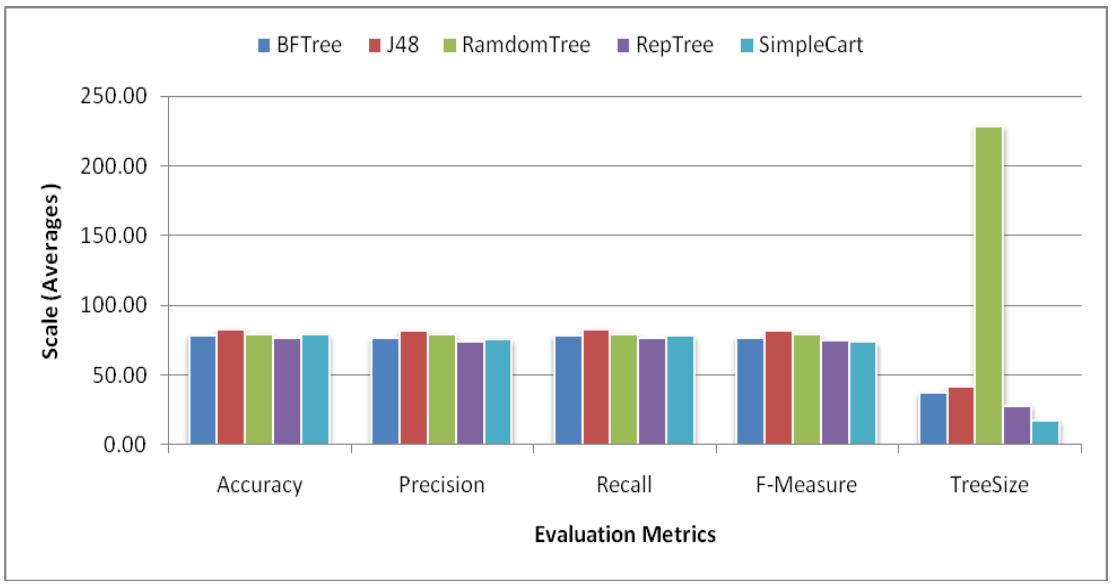


Figure 4.6: Graph of table 4.8

4.5. Context Analysis

The tables 4.3, 4.4, 4.5, 4.6, 4.7, 4.8 and the graph shown in figures 4.1, 4.2, 4.3, 4.4, 4.5 and 4.6 are the results of the simulations. They demonstrate the performance of decision tree Classification algorithms.

Table 4.3 showed the accuracy observed by implemented decision tree algorithms and figure 4.1 showed the averaged accuracy of table 4.3 where it ranged from 76.33% to 82.16%. Among all the algorithms J48 had got rich as well as motivating and encouraging result with 82.16% and REPTree was less capable to classify with accuracy of 76.33%.

Figure 4.2 showed the averaged precision of table 4.4 observed by implemented decision tree algorithms where it ranged from 73.91% to 81.71%. Among all the algorithms J48 had got good precision level of 81.71% whereas REPTree got less precision level of 73.91%.

Figure 4.3 showed the averaged recall of table 4.5 observed by implemented decision tree algorithms where it ranged from 76.33% to 82.15%. Among all the algorithms J48 had got again encouraging TP rate of 81.15% whereas REPTree got minimum TP rate of 76/33%.

Figure 4.4 showed the averaged F-measure of table 4.6 observed by implemented decision algorithms where it ranged from 73.32% to 81.31%. Again J48 had got victory over compared algorithms with the value of 81.31% and SimpleCard had got minimum value of 73.32%.

Figure 4.5 showed the averaged tree Size of table 4.7 observed by implemented decision algorithms where it ranged from 17.24 to 228.35. SimpleCart had got victory over compared algorithms with the tree size 17.24 and RandomTree had got large tree size 228.35.

Figure 4.6 showed that the comparison between all the evaluation metrics of all the implemented algorithms and from that comparison; J48 had got rich as well as motivating and encouraging performance in every aspects excepts tree size. Since SimpleCard had got minimum tree size for the searching SimpleCard is better and in other aspects J48 better.

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1. Conclusion

The comparison of classification algorithm is a complex task and it is an open problem. First, the notion of the performance can be defined in many ways: accuracy, speed, cost, reliability, etc. Second, an appropriate tool is necessary to quantify this performance. Third, a consistent method must be selected to compare with the measured values. The selection of the best classification algorithm for a given dataset is a very widespread problem. In this sense it requires to make several methodological choices. So this research focused in the analysis of decision tree classification algorithm in different datasets of multiple attributes and multiple instances. Where analysis was done among five decision tree classification algorithms (BFTree, J48, RandomTree, REPTree and SimpleCart).

From the context analysis it was seen that J48 was able to classify 82.16% of the data correctly which was best among all in comparison to results of evaluation metrics (Accuracy, Precision, Recall and F-Measure) and SimpleCart was able to build decision tree with small tree size of 17.24 (averaged value). In a nut shell, the experiment result showed that J48 had got about 3.89% better accuracy than BFTree, 2.97% better accuracy than RandomTree, 5.83% better accuracy than REPTree and 3.58% better accuracy than SimpleCart.

5.2. Future Works

Directions for future works are:

- Research can be done by comparing other decision tree classification algorithms.
- Research can be done by taking large datasets with more attributes values.
- Research can be done by comparing more evaluation metrics.

REFERENCES

[1] H. Jawei, K. Micheline, P. Jain, "Data Mining: Concepts and Techniques, Third Edition", Morgan Kaufmann Publishers an imprint of Elsevier, 225 Wyman Street, Waltham, MA 02451, USA 2012.

[2] T. Miranda Lakshmi, A. Martin, R. Mumtaj Begum, and V. Prasanna Venkatesan, "An analysis on performance of decision tree algorithms using student's Qualitative data." *International Journal of Modern Education and Computer Science (IJMECS)* 5, no. 5 (2013): 18.

[3] J. R. Quinlan, "C4.5: Programs for Machine Learning, Morgan Kauffman", 1993.

[4] Wu. Xindong, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, Geoffrey J. McLachlan et al., "Top 10 algorithms in data mining." *Knowledge and Information Systems* 14, no. 1 (2008): 1-37.

[5] Microsoft Press, "Microsoft® Computer Dictionary Fifth Edition", Microsoft Press a division of Microsoft corporation, one Microsoft way, Redmond, Washington 980852-6399, ISBN: 0-7356-1495-4, May 01, 2002. (Available at http://www.assaabloyamericasuniversity.com/Other/AssaAbloyAmericasUniv/Library/Reference%20Docs/Computer%20Dictionary.pdf).

[6] http://www.cs.waikato.ac.nz/ml/weka/ (accessed on Oct 6, 2013).

[7] Ian H. Witten, Eibe Frank, Mark A Hall, "Data Mining: Practical Machine Learning Tools and Techniques Third Edition", Morgan Kaufmann Publishers, an imprint of Elsevier, 30, Corporate Drive, Suite 400, Burlington, MA 01803, USA, ISBN: 978-0-12-374856-0, 2011.

[8] http://www.cs.waikato.ac.nz/ml/weka/arff.html (accessed on Oct 6, 2013).

[9] S. Ravichandran, V. B. Srinivasan and C. Ramasamy, "Comparative Study on Decision Tree Techniques for Mobile Call Detail Record." *Journal of Communication and Computer* 9, no. 12 (2012): 1331-1335

[10] W. Rui, Jing Lu. "The Improvement of Replacement Method for Web Caching." In *proceedings of the Third International Symposium on Computer Science and Computational Technology (ISCSCT' 10)*, Jiaozuo, P.R China, 14-15, August 2010.

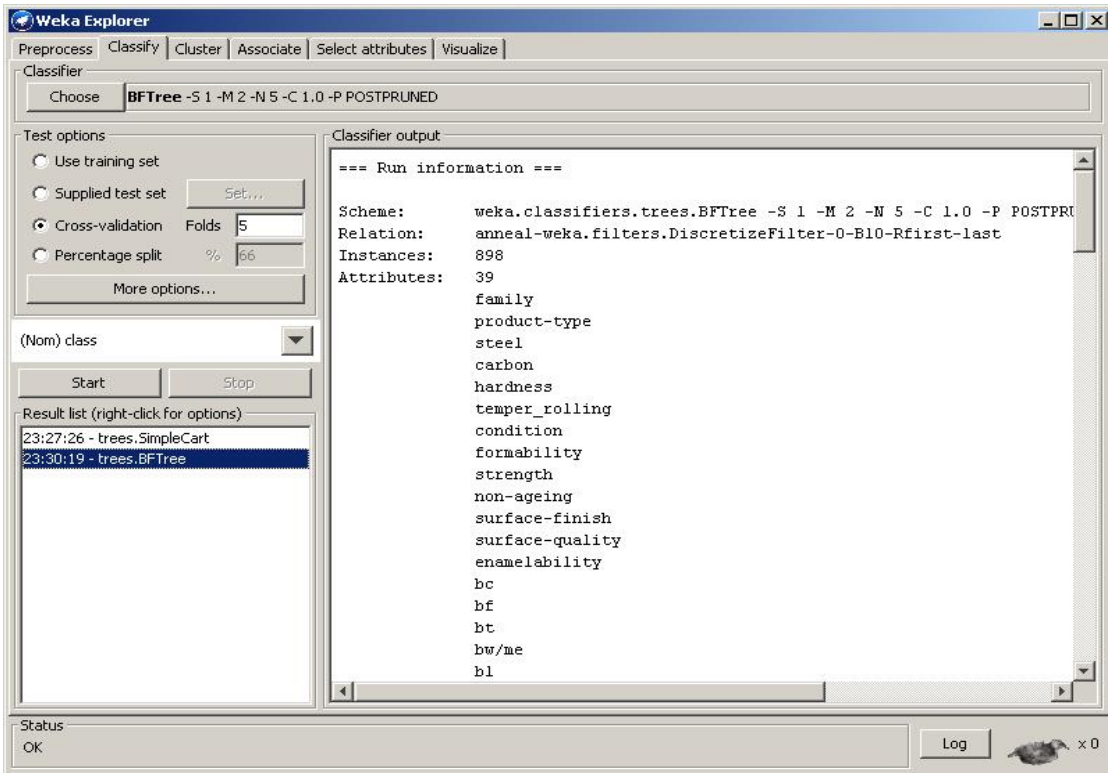
[11] S. Sarina, S. M. Shamsuddin, A. Abraham. "Intelligent web caching using adaptive regression trees, splines, random forests and tree net." In *Data Mining and Optimization (DMO), 2011 3rd Conference on*, pp. 108-114. IEEE, 2011.

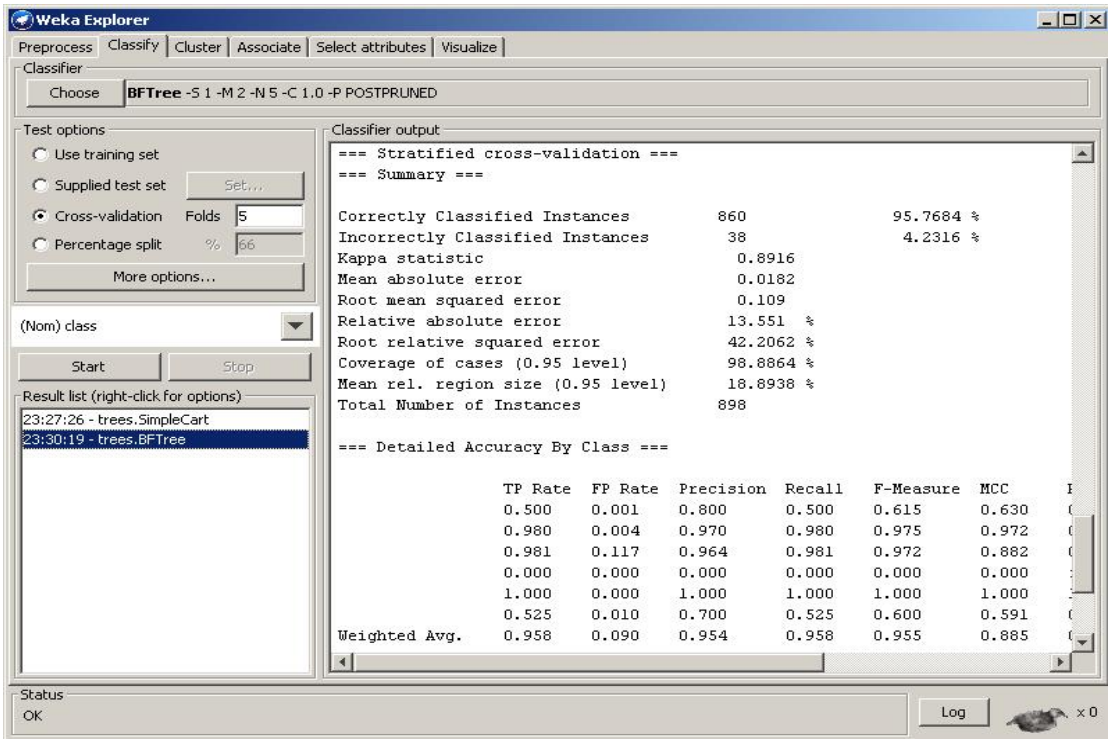
[12] A. Papagelis, D. Kalles, "GATree: genetically evolved decision trees", in *proceeding of 12th IEEE International Conference on Tools with Artificial Intelligence*, 2000. (Available at http://www.gatree.com/?page_id=46).

[13] A. Papagelis, D. Kalles, "Breeding Decision Trees Using Evolutionary Techniques", in p*roceeding of 18th International Conference on Machine Learning*, 2001. (Available at http://www.gatree.com/?page_id=46).

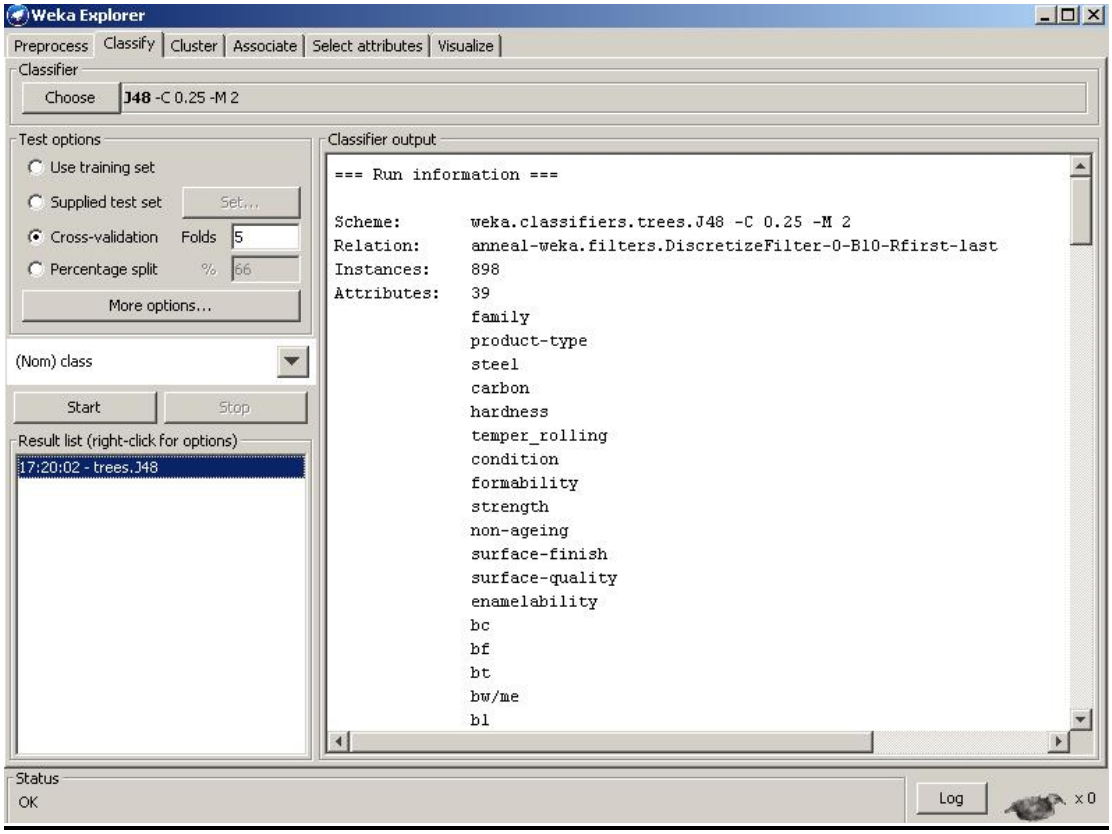
APPENDIX

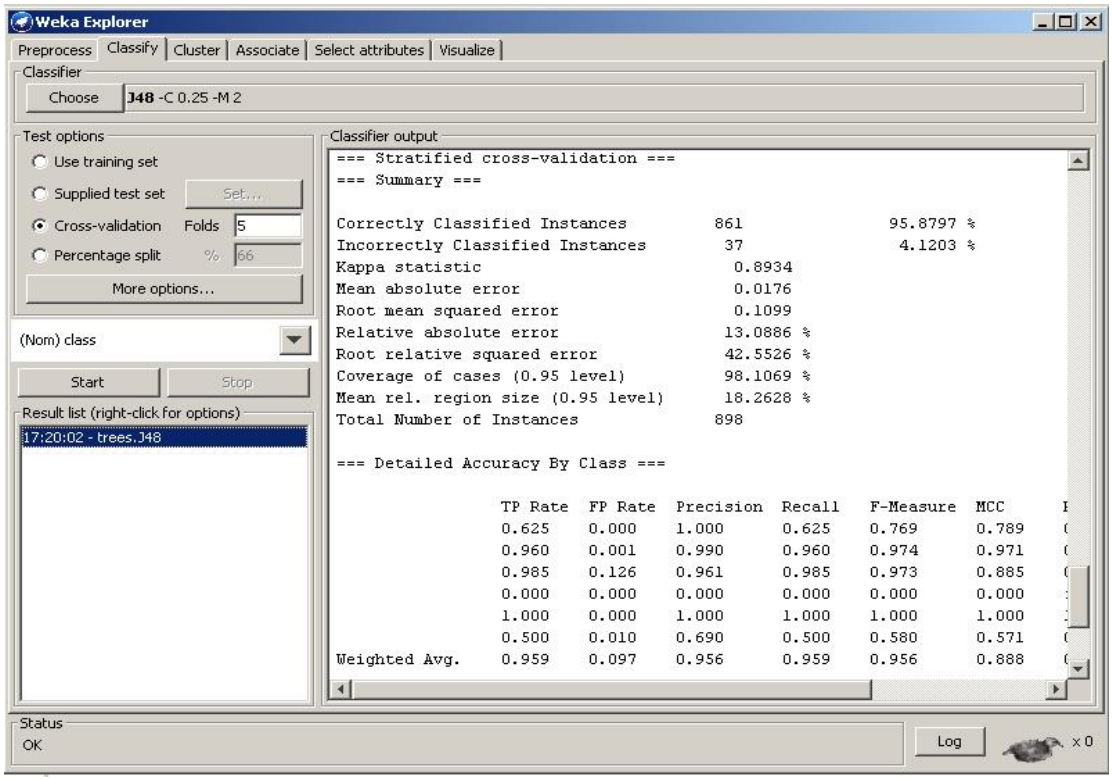
BF Tree



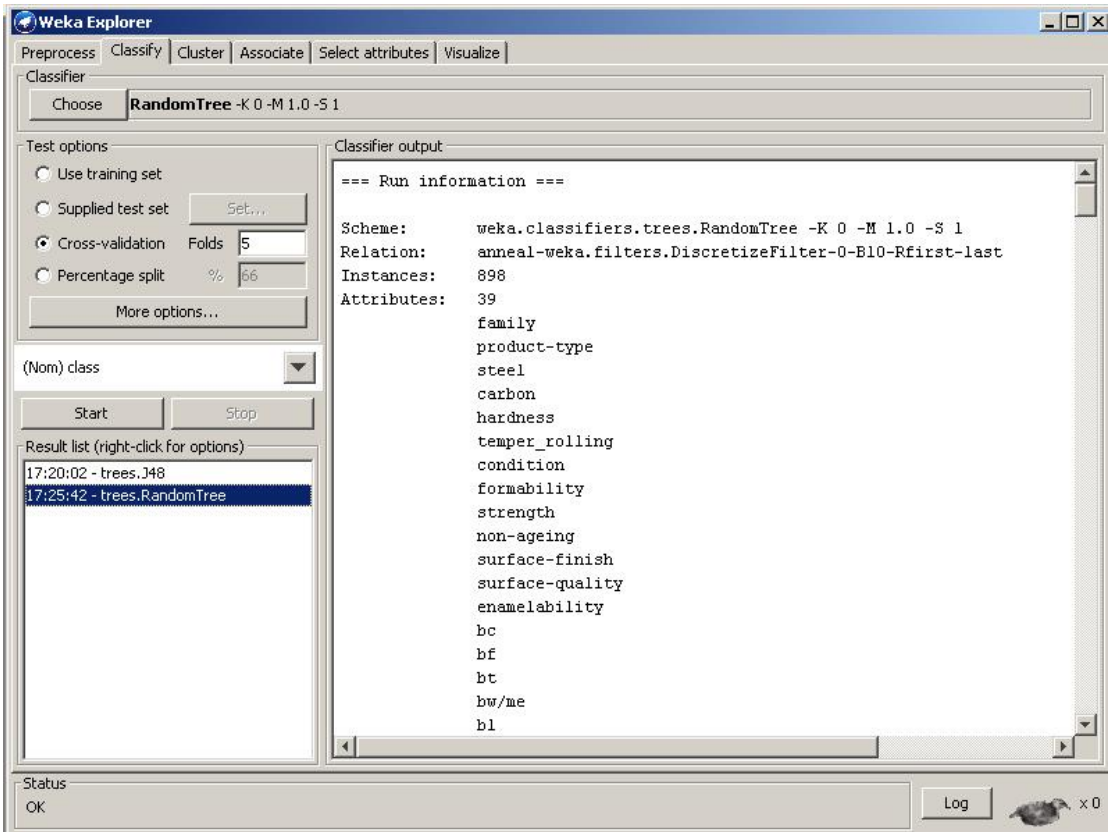


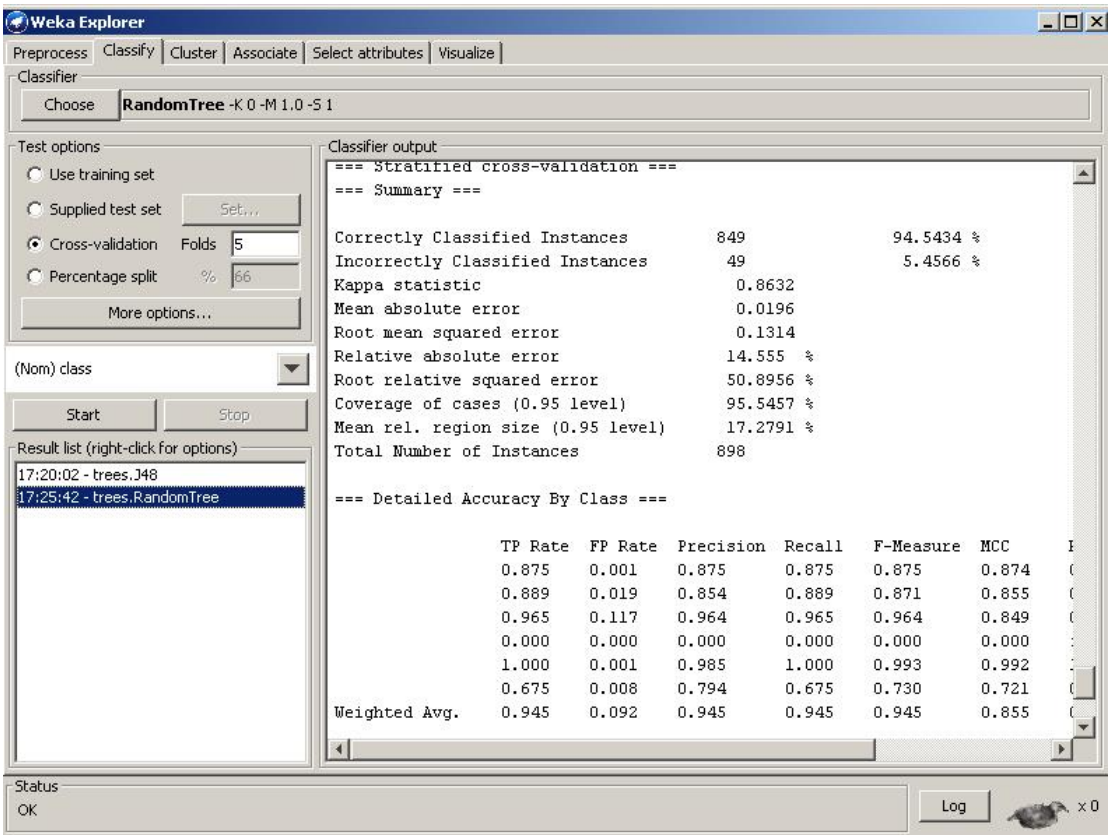
J48


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: anneal-weka.filters.DiscretizeFilter-O-B10-Rfirst-last
Instances: 898
Attributes: 39
family
product-type
steel
carbon
hardness
temper_rolling
condition
formability
strength
non-ageing
surface-finish
surface-quality
enamelability
bc
bf
bt
bw/me
bl
Status
OK
Log
x 0

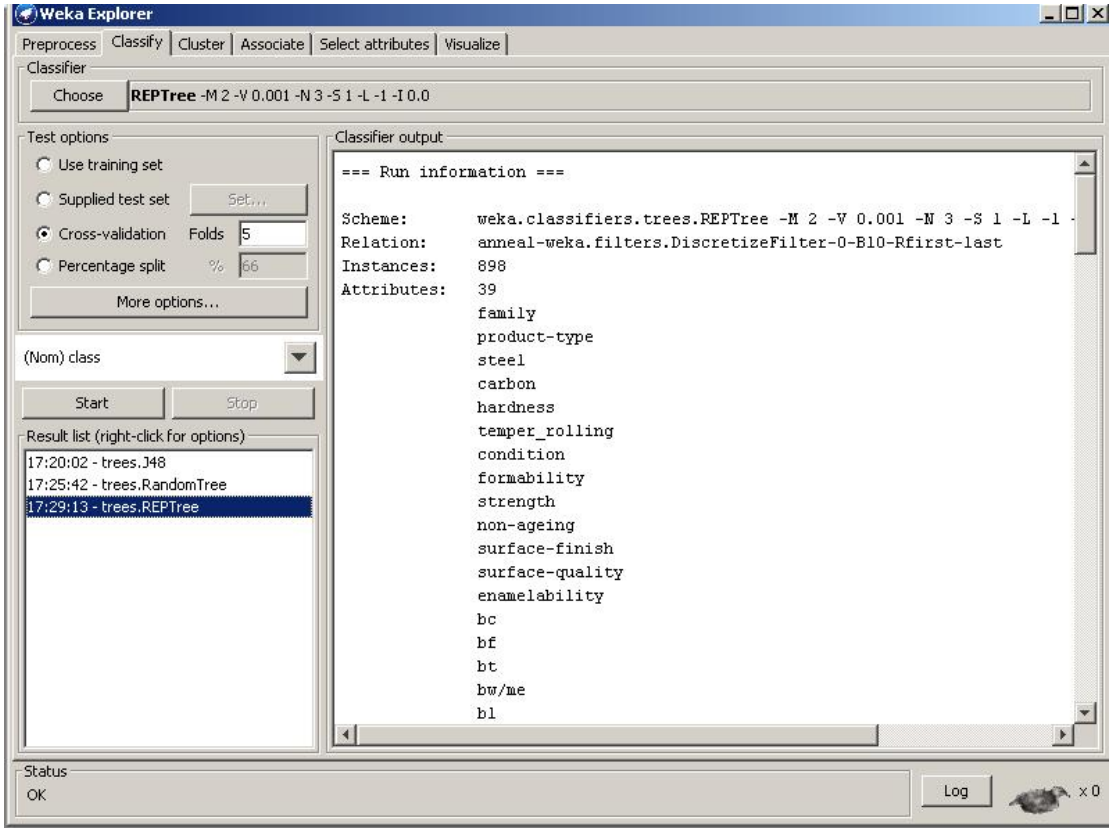

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 861 95.8797 %
Incorrectly Classified Instances 37 4.1203 %
Kappa statistic 0.8934
Mean absolute error 0.0176
Root mean squared error 0.1099
Relative absolute error 13.0886 %
Root relative squared error 42.5526 %
Coverage of cases (0.95 level) 98.1069 %
Mean rel. region size (0.95 level) 18.2628 %
Total Number of Instances 898
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC
0.625 0.000 1.000 0.625 0.769 0.789
0.960 0.001 0.990 0.960 0.974 0.971
0.985 0.126 0.961 0.985 0.973 0.885
0.000 0.000 0.000 0.000 0.000 0.000
1.000 0.000 1.000 1.000 1.000 1.000
0.500 0.010 0.690 0.500 0.580 0.571
Weighted Avg. 0.959 0.097 0.956 0.959 0.956 0.888
Status
OK
Log
x 0

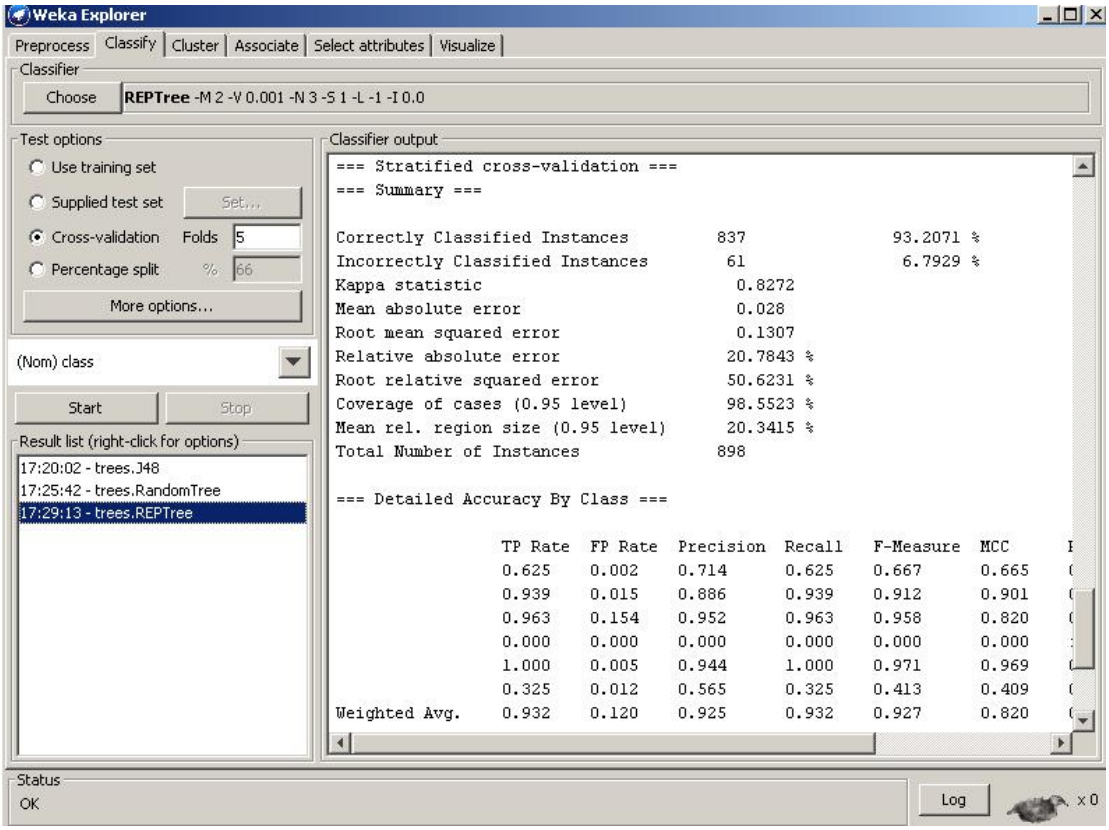
Random Tree


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
RandomTree -K 0 -M 1.0 -S 1
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.RandomTree -K 0 -M 1.0 -S 1
Relation: anneal-weka.filters.DiscretizeFilter-O-B10-Rfirst-last
Instances: 898
Attributes: 39
family
product-type
steel
carbon
hardness
temper_rolling
condition
formability
strength
non-ageing
surface-finish
surface-quality
enamelability
bc
bf
bt
bw/me
bl
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
17:25:42 - trees.RandomTree
Status
OK
Log
x 0

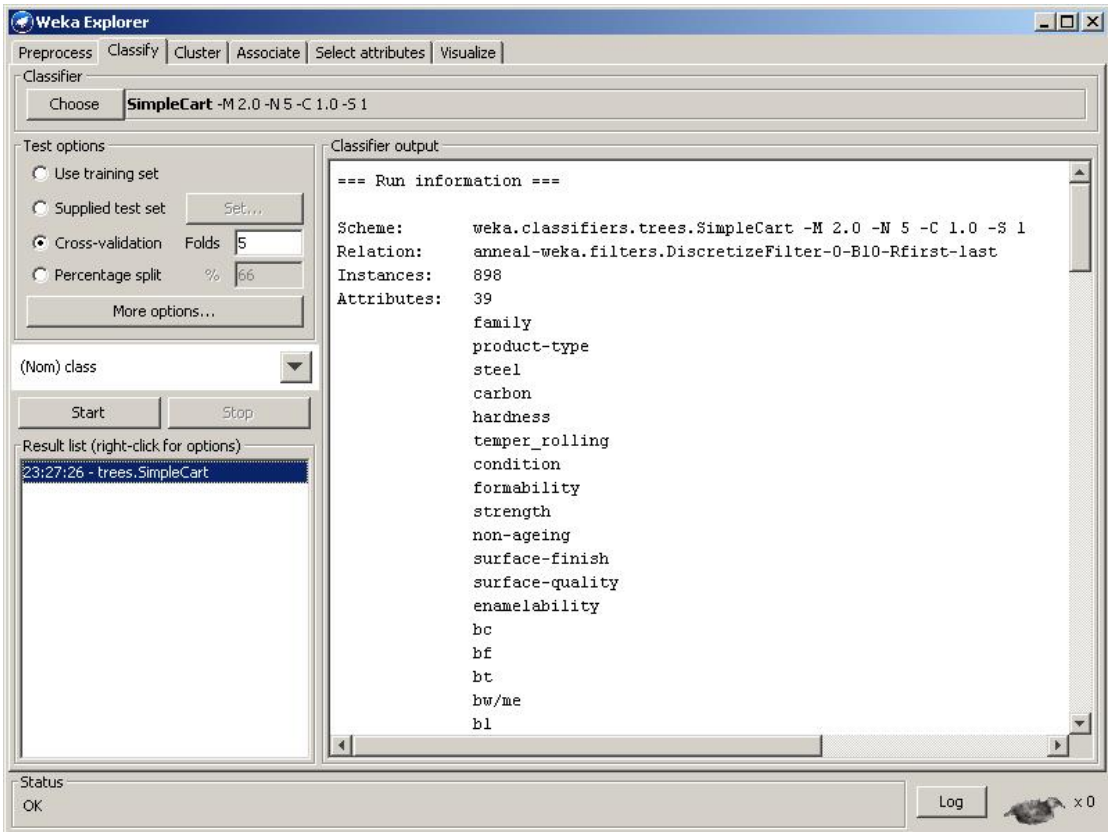

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
RandomTree -K 0 -M 1.0 -S 1
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 849 94.5434 %
Incorrectly Classified Instances 49 5.4566 %
Kappa statistic 0.8632
Mean absolute error 0.0196
Root mean squared error 0.1314
Relative absolute error 14.555 %
Root relative squared error 50.8956 %
Coverage of cases (0.95 level) 95.5457 %
Mean rel. region size (0.95 level) 17.2791 %
Total Number of Instances 898
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC
0.875 0.001 0.875 0.875 0.875 0.874
0.889 0.019 0.854 0.889 0.871 0.855
0.965 0.117 0.964 0.965 0.964 0.849
0.000 0.000 0.000 0.000 0.000 0.000
1.000 0.001 0.985 1.000 0.993 0.992
0.675 0.008 0.794 0.675 0.730 0.721
Weighted Avg. 0.945 0.092 0.945 0.945 0.945 0.855
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
17:25:42 - trees.RandomTree
Status
OK
Log
x 0

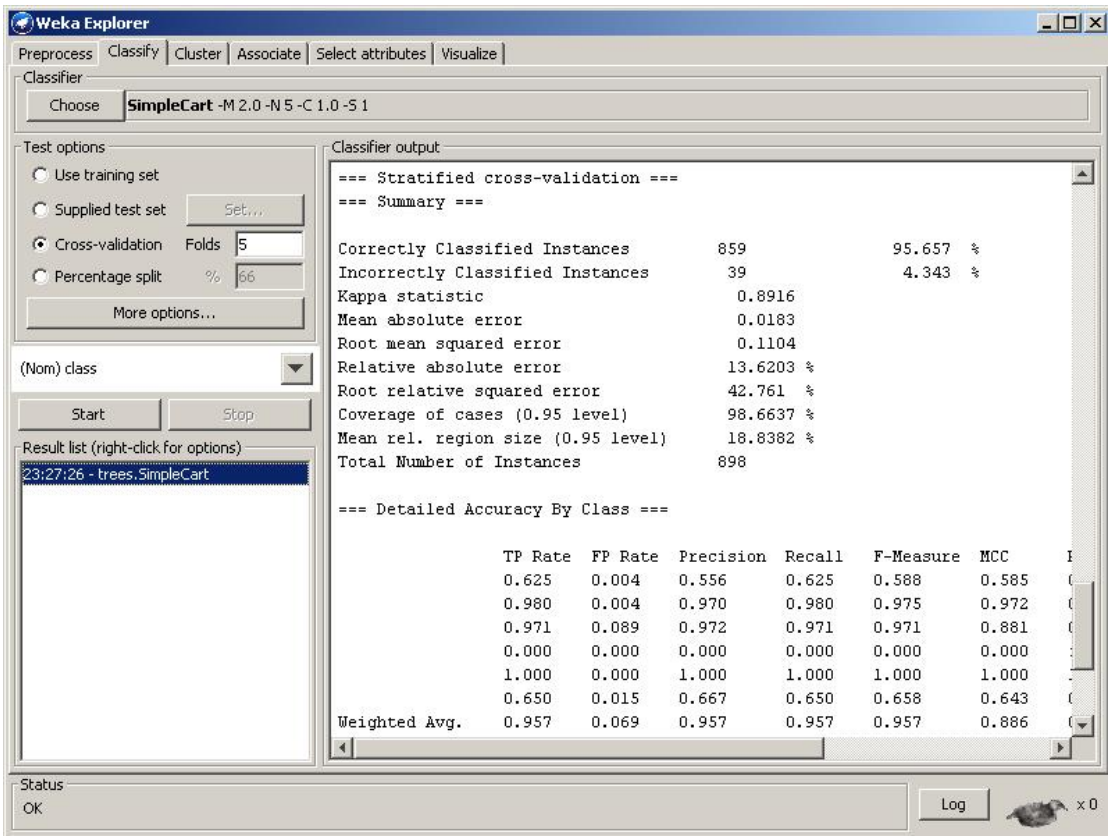
Rep Tree


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
17:25:42 - trees.RandomTree
17:29:13 - trees.REPTree
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -
Relation: anneal-weka.filters.DiscretizeFilter-O-B10-Rfirst-last
Instances: 898
Attributes: 39
family
product-type
steel
carbon
hardness
temper_rolling
condition
formability
strength
non-ageing
surface-finish
surface-quality
enamelability
bc
bf
bt
bw/me
bl
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
17:20:02 - trees.J48
17:25:42 - trees.RandomTree
17:29:13 - trees.REPTree
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 837 93.2071 %
Incorrectly Classified Instances 61 6.7929 %
Kappa statistic 0.8272
Mean absolute error 0.028
Root mean squared error 0.1307
Relative absolute error 20.7843 %
Root relative squared error 50.6231 %
Coverage of cases (0.95 level) 98.5523 %
Mean rel. region size (0.95 level) 20.3415 %
Total Number of Instances 898
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC
0.625 0.002 0.714 0.625 0.667 0.665
0.939 0.015 0.886 0.939 0.912 0.901
0.963 0.154 0.952 0.963 0.958 0.820
0.000 0.000 0.000 0.000 0.000 0.000
1.000 0.005 0.944 1.000 0.971 0.969
0.325 0.012 0.565 0.325 0.413 0.409
Weighted Avg. 0.932 0.120 0.925 0.932 0.927 0.820
Status
OK
Log
x 0

Simple Cart


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
SimpleCart -M 2.0 -N 5 -C 1.0 -S 1
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
23:27:26 - trees.SimpleCart
Classifier output
=== Run information ===
Scheme: weka.classifiers.trees.SimpleCart -M 2.0 -N 5 -C 1.0 -S 1
Relation: anneal-weka.filters.DiscretizeFilter-O-B10-Rfirst-last
Instances: 898
Attributes: 39
family
product-type
steel
carbon
hardness
temper_rolling
condition
formability
strength
non-ageing
surface-finish
surface-quality
enamelability
bc
bf
bt
bw/me
bl
Status
OK
Log
x 0


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
SimpleCart -M 2.0 -N 5 -C 1.0 -S 1
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
5
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
23:27:26 - trees.SimpleCart
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 859 95.657 %
Incorrectly Classified Instances 39 4.343 %
Kappa statistic 0.8916
Mean absolute error 0.0183
Root mean squared error 0.1104
Relative absolute error 13.6203 %
Root relative squared error 42.761 %
Coverage of cases (0.95 level) 98.6637 %
Mean rel. region size (0.95 level) 18.8382 %
Total Number of Instances 898
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC
0.625 0.004 0.556 0.625 0.588 0.585
0.980 0.004 0.970 0.980 0.975 0.972
0.971 0.089 0.972 0.971 0.971 0.881
0.000 0.000 0.000 0.000 0.000 0.000
1.000 0.000 1.000 1.000 1.000 1.000
0.650 0.015 0.667 0.650 0.658 0.643
Weighted Avg. 0.957 0.069 0.957 0.957 0.957 0.886
Status
OK
Log
x 0