TITLE

PREDICTION OF STUDENT ACADEMIC PERFORMANCE USING DECISION TREE ALGORITHM (C4.5)

BY

EZEJA LILIAN CHINENYE

FPI/HND/COM/14/019

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, THE FEDERAL POLYTECHNIC, IDAH KOGI STATE, NIGERIA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE HIGHER NATIONAL DIPLOMA (HND) IN COMPUTER SCIENCE.

DECEMBER, 2016

TITLE

PREDICTION OF STUDENT ACADEMIC PERFORMANCE USING DECISION TREE ALGORITHM (C4.5)

BY

EZEJA LILIAN CHINENYE

FPI/HND/COM/14/019

CERTIFICATION

This is to certify that this academic research was carried out by **EZEJA LILIAN CHINENEYE**, Registration number **FPI/HND/COM/14/019** and submitted to the Department of Computer Science, The Federal Polytechnic, Idah Kogi State, in partial fulfillment of the requirement for the award of Higher National Diploma (HND) in Computer Science.

_____ _____

Mr. Obeta S.C Date

Project Supervisor

_____ _____

Mrs. Odiketa J .C Date

Head of department

DEDICATION

This project work is strictly dedicated to God Almighty the giver of wisdom, knowledge and understanding, without whom my dream wouldn't have become a reality. To him alone is my praises forever Amen.

ACKNOWLEDGMENT

I acknowledge the effort of my Head of Department Mrs. Odiketa J .C, my supervisor Mr. Obeta S.C and other lecturers whom God has used as tool to impact knowledge on me, hence facilitating the fulfillment of my dream in school.

I wish to acknowledge the great effort of my Parents Mr. and Mrs. Ezeja Thomas who has ensured my dream of acquiring Higher National Diploma becomes a reality. This dream wouldn't have been realistic without their support; financially, mentally, spiritually and otherwise.

I wish to acknowledge the effort of my best Igwe Uchenna Otta for his love, care and support towards the accomplishment of my dream.

I acknowledge the contribution of my brothers and sisters especially Uche, Chidi, Nnenna, Nnabuike, Chizoba, Ifeanyi, Chioma, and others who were really instrumental for the successful completion my programme. There supports to ensure my needs in the school are well catered for.

I also acknowledge my son Munachimso for his endurance and patience throughout my stay in school. May God continue to strengthen you.

All these people I have acknowledged and those who have also contributed but I forget to mention their name in this work, I pray Almighty God to grant all their heart desire in Jesus Name

ABSTRACT

The desire of every organization is to extract hidden but useful knowledge from their data through data mining tools. Also, the recent decline in the standard of education in most developing countries has necessitated researches that will help proffer solutions to some of the problems. From the literature, different analysis has been carried out on university data, which includes student's university entrance examination, Ordinary level results and the relationship between these entry results and students' final graduation grades. Therefore, in this work, a new system that will predict students' grades based on the first semester results data using the C4.5 (J48) decision tree algorithm was developed. C4.5 decision tree algorithm was used to train the data of the ND1 first semester result. The knowledge represented by decision trees were extracted and presented in form of IF-THEN rules. The trained data were then used to develop a model for making future prediction of students' first semester grades. The developed system could be very useful in predicting students' final graduation grades even from the point of entry into the university. This will help management staff, academic planners to properly counsel students in order to improve their overall performance.

Keywords: Data mining, Decision trees, Prediction, C4.5 algorithm, Knowledge extraction

TABLE OF CONTENT

CHAPTER ONE

1.0 INTRODUCTION

Over the years, data mining has been used for extraction of implicit, previously undetected and useful information from large amounts of data. It is often used for prediction from the knowledge pattern.

Data Mining is the process of extracting information from large data sets through the use of algorithms and techniques drawn from the field of statistics and Database Management Systems. It also detects hidden knowledge and patterns which were previously unknown from large databases for easy and fast retrieval of data and information. Different analysis has been done on university students' entrance examination and ordinary level result but the relationships between them and the final graduation grades have not received much attention from the research world.

Extraction of hidden knowledge and subsequent management of these much important enrolment data could be of very great importance to the management and stakeholders of academic institutions, most especially in the area of decision making, which would in turn improve students' performance through proper guidance and counseling.

Data mining techniques has also been used in several occasions in solving educational problems and to perform crucial analysis in the educational sector. This is to enhance educational standards and management such as investigating the areas of learning recommendation systems, learning material arrangement, continuous student assessments and evaluation of educational websites. Data Mining discovers relationships among attributes in data set producing conditional statements concerning attribute-values. Classification is one of the most commonly applied data mining technique which uses a set of pre-classified examples to develop a model that can classify the population of records at large (Samrat and Vikesh, 2012). Therefore in this work, a data mining system using the C4.5 decision tree algorithm for classification of data was deployed to mine students' enrolment data and the knowledge gained was used to predict possible student graduation grades.

1.1 BACKGROUND OF THE STUDY

In today's field of computer science, there is an emergence of an interesting and challenging subfield that has made changes in technology in the areas of computational analysis easy. It also offers powerful tools in handling complex programming task which the simple programming approach cannot handle. These subfields are known as data mining and machine learning.

Examples of the complex programming tasks handled by machine learning are clustering, classification, association and prediction. Several of the machine learning algorithms have been developed (Mitchell 1999). Decision tree machine learning algorithm of these several algorithms will be applied to this work to predict student's performance.

1.2 STATEMENT OF PROBLEM

The performance of students in higher institution is becoming less encouraging and the rate at which student carryover, withdrawn, repeats and even drop out is becoming alarming and even those that succeeds ends up with poor result due to the issue that they are not properly guided.

However, the result of this work will help identify those students who need special advising or counseling and the lecturer can develop them with special attention that will help those students with confidence.

1.3 AIMS AND OBJECTIVES

The aim of the work is to develop a model for predicting student's academic performance. This is to aid academic institutions in producing a quality graduates.

Objectives are:

i. To develop a model that will confidently predict performance of students of higher institutions.
ii. To test and validate the model in [i] above

1.4 RESEARCH MOTIVATION

The number of candidate who is admitted into institution of higher learning is on the increase yearly and the number that is graduating successfully is becoming fewer and even those that are successful ends with poor result.

The idea of this work is to assist lecturers and individual students develop strategies that will improve academic performance thereby reducing failing ratio by taking appropriate steps at the right time to improve the quality of education.

1.5 RESEARCH METHODOLOGY

The data collected was course for ND1 first semester in Federal Polytechnic Idah, Kogi State, Nigeria. The grades considered were the semester results for the courses in 2015/2016 academic session. The data collected from the result sheet was entered into Microsoft excel for analysis and then preprocessed in WEKA to eliminate errors. The data set was used to train the model using the training set (data set with class label) in WEKA and the model was built. After which the model was tested based on the training it has received using the testing set (data set without class label). The java programming language was then used to write the program carried out in WEKA for easy access.

1.6 THESIS ORGANISATION

The thesis is organized as follows: chapter two discuses related works on data mining, decision tree algorithm, Iterative Dichotomizer 3 (ID3) algorithms and C4.5 algorithm but the work is focused on C4.5 algorithm. In chapter three I analyze the system problem, gave the details of the methodology, dataset description, feature set description and the experimental setup. In chapter four, I presented my system requirements, implementation detail, results and evaluation then discussion. In chapter five, I present conclusion of the work, recommendation and point out areas for future works.

CHAPTER TWO

2.0 LITERATURE REVIEW

This section examines the concepts and ideas in the literature relating to this research area but the research is focused on C4.5 algorithm.

2.1 DATA MINING (DM)

DM refers to a particular step in the Knowledge discovery process. It consists of particular algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns (models) over the data (Albashiri, 2013

Data mining is also a process of extracting nontrivial, valid, novel and useful information from large databases. DM can be viewed as a kind of search for meaningful patterns or rules from a large search space, that is, the database (Srinivasa et al., 2007). As this knowledge is captured, it can become a very vital tool to gaining a competitive advantage over competitors in an industry. This then creates an important need for tools to analyze and model these data.

According to Rao and Vidyavathi (2010), data mining is a powerful new technology with great potential to help companies focus on the most important information in the data they have collected about the behaviour of their customers and potential customers.

DM also refers to the process of extracting novel, useful, and understandable pieces of information (e.g., patterns, rules, regularities, constraints) from data in databases (Saravanan and Christopher, 2012).

Recently, researchers had also proposed that the future of data mining has a lot to benefit from other technologies such as cloud computing. Cloud computing combined with data mining can provide powerful capacities of storage and computing and an excellent resource management (Qureshi et al., 2013).

Due to the explosive data growth and amount of computation involved in data mining, an efficient and high-performance computing is very necessary for a successful data mining application. Data mining in the cloud computing environment can be considered as the future of data mining because of the advantages of cloud computing paradigm.

Cloud computing provides greater capabilities in data mining and data analytics. The major concern about data mining is that the space required by the operations and item sets is very large. By combining data mining with cloud computing, a considerable amount of space can be saved and the cost of computational resources can be greatly reduced (Pareek and Gupta, 2012).

Data mining is the process of discovering interesting knowledge from large amount of data stored in database, data warehouse or other information repositories.

Data Mining is the process of extracting information from large data sets through the use of algorithms and techniques drawn from the field of Statistics, Machine Learning and Data Base Management Systems (Surjeet & Saurabh, 2012).

Data mining is data analysis methodology used to identify hidden patterns in a large data set. It has been successfully used in different areas including the educational environment.

Educational data mining is an interesting research area which extracts useful, previously unknown patterns from educational database for better understanding, improved educational performance and assessment of the student learning process (Surjeet & Saurabh, 2012).

The main functionality of data mining techniques is applying various methods and algorithms in order to discover and extract patterns of stored data. These interesting patterns are presented to the user and may be stored as new knowledge in knowledge base.

Data mining has been used in areas such as database systems, data warehousing, statistics, machine learning, data visualization, and information retrieval.

Data mining techniques have been introduced to new areas including neural networks, patterns recognition, spatial data analysis, image databases and many application fields such as business, economics and bioinformatics. Some types of data mining techniques are: Clustering, Association Rule Mining, Neural Networks, Genetic

Algorithms, Nearest Neighbor Method, Classification Rule Mining, Decision trees and many others.

2.2 DECISION TREES ALGORITHM

A decision tree is a flow-chart-like tree structure, where each internal node is denoted by rectangles, and leaf nodes are denoted by ovals. All internal nodes have two or more child nodes. All internal nodes contain splits, which test the value of an expression of the attributes. Arcs from an internal node to its children are labeled with distinct outcomes of the test. Each leaf node has a class label associated with it.

Decision tree are commonly used for gaining information for the purpose of decision-making. Decision tree starts with a root node on which it is for users to take actions.

From this node, users split each node recursively according to decision tree learning algorithm.

The final result is a decision tree in which each branch represents a possible scenario of decision and its outcome (Surjeet & Saurabh, 2012).

In data mining, decision trees can be described also as the combination of mathematical and computational techniques to aid the description, categorization and generalization of a given set of data. The four widely used decision tree learning algorithms are: ID3, CART, CHAID and C4.5.

2.3 ID3 (ITERATIVE DICHOTOMIZER 3) ALGORITHM

ID3 is a simple decision tree learning algorithm developed by Ross Quinlan.

The basic idea of ID3 algorithm is to construct the decision tree by employing a top-down, greedy search through the given sets to test each attribute at every tree node. In order to select the attribute that is most useful for classifying a given sets, we introduce a metric - information gain. To find an optimal way to classify a learning set, what we need to do is to minimize the questions asked (i.e. minimizing the depth of the tree). Thus, we need some function which can measure which question

provides the most balanced splitting. The information gain metric is such a function. (Surjeet & Saurabh, 2012).

A decision tree is a tree in which each branch node represents a choice between a number of alternatives, and each leaf node represents a decision. They are commonly used for gaining information for the purpose of decision making. Decision tree starts with a root node on which it is for users to take appropriate actions. From this node, users split each

node recursively according to decision tree learning algorithm. The final result is a decision tree in which each branch represents a possible scenario of decision and its outcome.

ID3 algorithm is one important method in the technology of decision tree Classifications and so is widely applied. ID3 algorithm searches through attributes of the training instances and extracts the attribute that best separates the given examples.

If the attribute perfectly classifies the training sets then ID3 stops; otherwise it recursively operates on the n (where n = number of possible values of an attribute) partitioned subsets to get their "best" attribute. The algorithm uses a greedy search, that is, it picks the best attribute and never looks back to reconsider earlier choices.

The central principle of ID3 algorithm is based on the information theory.

2.4 C4.5 ALGORITHM

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan .C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. It became quite popular after ranking #1 in the Top 10 Algorithms in Data Mining pre-eminent paper published by Springer LNCS in 2008. Algorithm C4.5 builds decision trees from a set of training data in the same way as ID3 , using the concept of information entropy . The training data is a set of already classified samples. Each sample consists of a p- dimensional vector, where they represent attribute values or features of the sample, as well as the class in which falls. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sublists. This algorithm has a few base cases.

All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class. None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using

the expected value of the class. Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

PSEUDO CODE

In pseudo code, the general algorithm for building decision trees is:

1. Check for the above 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. Recur on the sub lists obtained by splitting on A_best, and add those nodes as children of node.

IMPLEMENTATIONS

J48 is an open source Java implementation of the C4.5 algorithm in the Weka data mining tool.

IMPROVEMENTS FROM ID.3 ALGORITHM

C4.5 made a number of improvements to ID3. Some of these are:

- Handling both continuous and discrete attributes – In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it.
- Handling training data with missing attribute values - C4.5 allows attribute values to be marked as "?" for missing. Missing attribute values are simply not used in gain and entropy calculations.
- Handling attributes with differing costs.
- Pruning trees after creation - C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes.

2.5 OTHER RELATED WORKS

These are some of the related work that describes data mining and knowledge discovery in the context of this work.

Surjeet & Saurabh (2012) used student's past academic performance to create a model using ID3 decision tree algorithm for prediction of student's enrolment in MCA course.

Shreenath & Madhu (2012) developed a model for the placement database, so that institutions can use it to discover some interesting patterns that could be analyzed to plan their future activities. The work used Apriori Algorithm to discover student academic behaviours.

Al-Radaideh et al. (2006) applied a decision tree model to predict the final grade of students who studied the C++course in Yarmouk University, Jordan in the year 2005. Three different classification methods namely ID3, C4.5 and the Naïve Bayes were used. The outcome of their results indicated that Decision Tree model had better prediction than other models.

Hijazi & Naqvi (2006) conducted a study on the student performance by selecting a sample of 300 students (225 males, 75 females) from a group of colleges affiliated to Punjab University of Pakistan. The hypothesis that was stated as "Students' attitude towards attendance in class, hours spent in study on daily basis after college, students family income, students mother's age and mother's education are significantly related with student performance" was framed. By means of simple linear regression analysis, it was found that the factors like mother's education and student's family income were highly correlated with the student academic performance.

Khan (2005) conducted a performance study on 400 students comprising 200 boys and 200 girls selected from the senior secondary school of Aligarh Muslim University, Aligarh, India with a main objective to establish the prognostic value of different measures of cognition, personality and demographic variables for success at higher secondary level in science stream. The selection was based on cluster sampling technique in which the entire population of interest was divided into groups, or clusters, and a random sample of these clusters was selected for further analyses. It was found that girls with high socio-economic status had relatively higher academic achievement in science

stream and boys with low socio-economic status had relatively higher academic achievement in general.

Bray (2007) in his study on private tutoring and its implications, observed that the percentage of students receiving private tutoring in India was relatively higher than in Malaysia, Singapore, Japan, China and Sri Lanka.

It was also observed that there was an enhancement of academic performance with the intensity of private tutoring and this variation of intensity of private tutoring depends on the collective factor namely socio-economic conditions.

Oladipupo &Oyelade (2009) analyzed a data using undergraduate students' result in the department

of Computer Science from a university in Nigeria. The department offers two programmes; Computer Science and Management Information Science. A total number of 30 courses for 100 level and 200 level students were considered as a case study. The analysis revealed that there is more to students' failure than the students' ability. It also reveals some hidden patterns of students' failed courses which could serve as bedrock for academic planners in making academic decisions and an aid in the curriculum re-structuring and modification with a view to improving students' performance and reducing failure rate.

Kovacic (2010) presented a case study on educational data mining to identify up to what extent the enrolment data can be used to predict students' success. The algorithms CHAID and CART were applied on student enrolment data of information system students of Open Polytechnic of New Zealand to get two decision trees classifying successful and unsuccessful students. The accuracy obtained with CHAID and CART was 59.4 and 60.5 respectively.

Bharadwaj & Pal (2011) conducted study on the student performance based by selecting 300 students from 5 different degree college conducting BCA (Bachelor of Computer Application) course of Dr. R.M.L. Awadh University, Faizabad, India. By means of Bayesian classification method on 17 attributes, it was found that the factors like students' grade in senior secondary exam, living location, medium of teaching, mothers' qualification, students other habit, family annual income and students' family status were highly correlated with the student academic performance.

Yadav et al. (2011) obtained the university students data like attendance, class test, seminar and assignment marks from the students' database, to predict the performance at the end of the semester using three algorithms ID3, C4.5 and CART and showed that CART is the best algorithm for classification of data.

Ramasubbareddy et al. (2011) proposed associative classification which is a classification of a new tuple using association rules.

Associative classification is a combination of association rule mining and classification. They searched for strong associations between frequent patterns and class labels. The main aim of the paper was to improve accuracy of the classifier. The accuracy could be achieved by producing all types of negative class association rules.

Al´ıpio et al. (2001) studied a new technique called post-bagging, which consists in re-sampling parts of a classification model rather than the data. They did this with a particular kind of model: large sets of classification association rules, and in combination with ordinary best rule and weighted voting approaches. They empirically evaluated the effects of the technique in terms of classification accuracy.

CHAPTER THREE

METHODOLOGY AND DESIGN

3.0 SYSTEM PROBLEM ANALYSIS:

This system solves the problem of how to predict students' grade point average with the use of a machine learning algorithm – a decision tree algorithm called C4.5. (J48 in WEKA, a machine learning platform built in the University of Waikato).

First the system must be trained so as to be able to do the task of predicting grades for students whose records it has not accessed previously. This is going to be achieved by granting the system an opportunity to learn from experience in the form of training instances.

Grade points average (GPA) is the term used to describe the performance of a student at the end of a period of learning, say a semester, and it is computed from the student's scores in each of the courses he or she has studied during the learning period. But this system is not necessarily concerned with the computation of students' scores or their resultant grades, rather, the focus is to gather such details, clean them, and use it to train the system. Then allow the system to predict future grades (GPA) given the needed scores only.

3.1 DETAILS OF THE METHODOLOGY:

3.1.1 Data Preprocessing:

This is the initial task that deals with cleaning the data, normalization, and saving the data in a format that is easily scalable to the tools.

An algorithm such as K-means is able to perform even with noise and irregularities in the dataset, whereas decision trees are sensitive to noise and irregular record sets. However, it is best to ensure that datasets are well prepared before they are loaded. In some cases, datasets need to be discretized before they can be loaded, for instance if one were to be working with an algorithm such as ID3, one needs to discretize numeric or continuous attribute values.

3.1.2 The Basic Decision Trees Technique (C4.5):

Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the sub tree rooted at the new node.

Figure *3.1* illustrates a typical learned decision tree for *Table 3.1* below:

S_NO	COM112	GNS127	STA112	GRADE
1	A	A	A	Distinction
2	A	AB	A	Upper
3	A	AB	AB	Upper
4	BC	C	D	Pass
5	BC	C	A	Upper

Table 3.1: Students' score

The decision tree for this table would classify a set of students' scores in their three main courses as an indicator for obtaining ***Distinction, Upper or Pass*** respectively. For example, the instance ***(COM112 = A, GNS127 = A, STA112 = A)*** would be sorted down the left most branch of this decision tree and would therefore be classified as appropriate (i.e., the tree predicts that ***Grade = Distinction).***

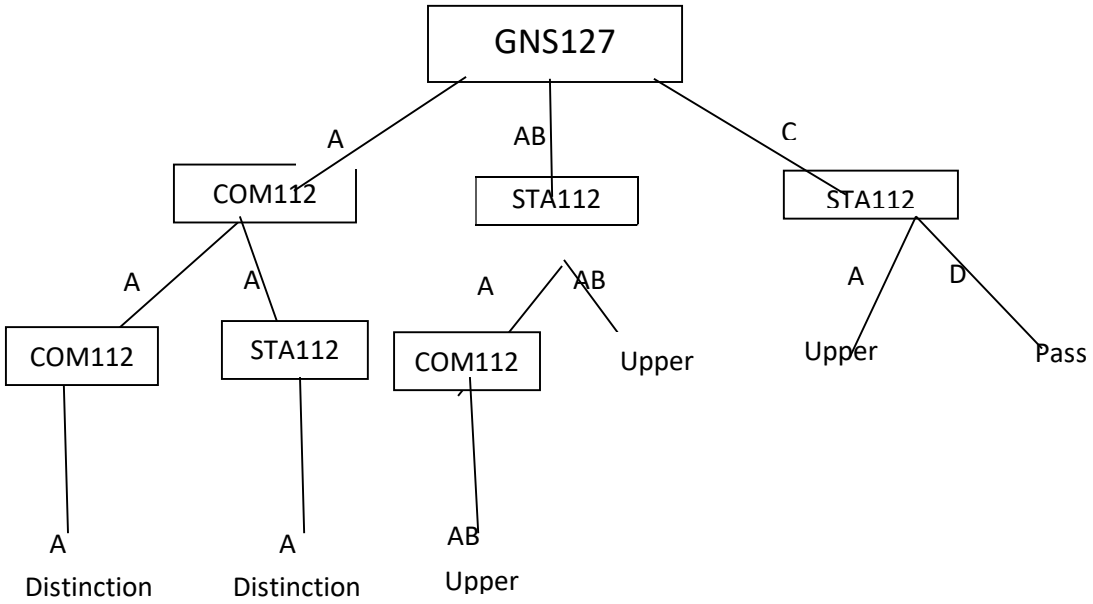


Figure 3.1: Decision tree for students' academic grades

In general, decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances. Each path from the tree root to a leaf corresponds to a conjunction of attribute tests and the tree itself to a disjunction of these conjunctions.

Most algorithms that have been developed for learning decision trees are variations on a core algorithm that employs a top-down, greedy search through the space of possible decision trees. This approach is exemplified by the ID3 algorithm (Quinlan 1986) and its successor C4.5 (Quinlan 1993). In that way, both ID3 and C4.5 operate on the same principle of growing the decision tree in a top-down fashion. Another very important feature of both algorithms is that they both learn decision trees by constructing them top-down, beginning with the question "which attribute should be tested at the root of the tree?" The answer to this question is that, each instance attribute is evaluated using a statistical test to determine how well it alone classifies the training examples. The best attribute is selected and used as the test at the root node of the tree. A descendant of the root node is then created for each possible value of this attribute, and the training

examples are sorted to the appropriate descendant node (i.e., down the branch corresponding to the example's value for this attribute). The entire process is then repeated using the training examples associated with each descendant node to select the best attribute to test at that point in the tree. This forms a greedy search for an acceptable decision tree, in which the algorithm never backtracks to reconsider earlier choices.

The central choice in the ID3 and C4.5 algorithms is selecting which attribute to test at each node in the tree. We would like to select the attribute that is most useful for classifying examples. What is a good quantitative measure of the worth of an attribute? We will define a statistical property, called ***information gain,*** which measures how well a given attribute separates the training examples according to their target classification. ID3 and C4.5 uses this information gain measure to select among the candidate attributes at each step while growing the tree. Information gain calculation is dependent on a measure in Information Theory called Entropy – the measure of purity or impurity in an arbitrary collection of examples (*Tom, M. M., 1997*).

Entropy is given by the relation:

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Where $p_{\oplus}$, is the proportion of positive examples in S and $p_{\ominus}$, is the proportion of negative examples in S. In all calculations involving entropy we define 0 log 0 to be 0.

Also, information gain is defined as the expected reduction in the entropy caused by partitioning the examples according to this attribute. More precisely, the information gain, ***Gain(S, A)*** of an attribute **A,** relative to a collection of examples ***S,*** is defined as

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Where ***Values(A)*** is the set of all possible values for attribute A, and ***S,*** is the subset of S for which attribute A has value v $(\text{i.e., } S_v = \{s \in S | A(s) = v\})$.

Information gain is precisely the measure used by ID3 or its successor C4.5 to select the best attribute at each step in growing the tree. Consider the use of information gain to evaluate the relevance of attributes using a portion of our dataset for student results given in *table 3.1* as summarized in *Figure 3.2.*

From *Table 3.1:*

$Entropy\ (S) = -P_{Distinction}log_2P_{Distinction} - P_{Upper}log_2P_{Upper} - P_{Pass}log_2P_{Pass}$

$= \{[-(^1/_5)*log_2\ (^1/_5)] - [(^3/_5)*log_2\ (^3/_5)] - [(^1/_5)*log_2\ (^1/_5)]\}$

$= 1.371$

$Gain(S, COM112) = \{S - [(|S_A|/5)*Entropy\ (S_A)] - [(|S_{BC}|/5)*Entropy\ (S_{BC})]\}$

$= \{1.371 - [(^3/_5)*(0.918)] - [(^2/_5)*(1)]\}$

$= 0.42$

$Gain(S, GNS127) = \{S - [(|S_A|/5)*Entropy\ (S_A)] - [(|S_{AB}|/5)*Entropy\ (S_{AB})] - [(|S_C|/5)*Entropy\ (S_C)]\}$

$= \{1.371 - [(^1/_5)*(0)] - [(^2/_5)*(0)] - [(^2/_5)*(1)]\}$

$= 0.971$

$Gain(S, STA112) = \{S - [(|S_A|/5)*Entropy\ (S_A)] - [(|S_{AB}|/5)*Entropy\ (S_{AB})] - [(|S_D|/5)*Entropy\ (S_D)]\}$

$= \{1.371 - [(^3/_5)*(0.918)] - [(^1/_5)*(0)] - [(^1/_5)*(0)]\}$

$= 0.82$

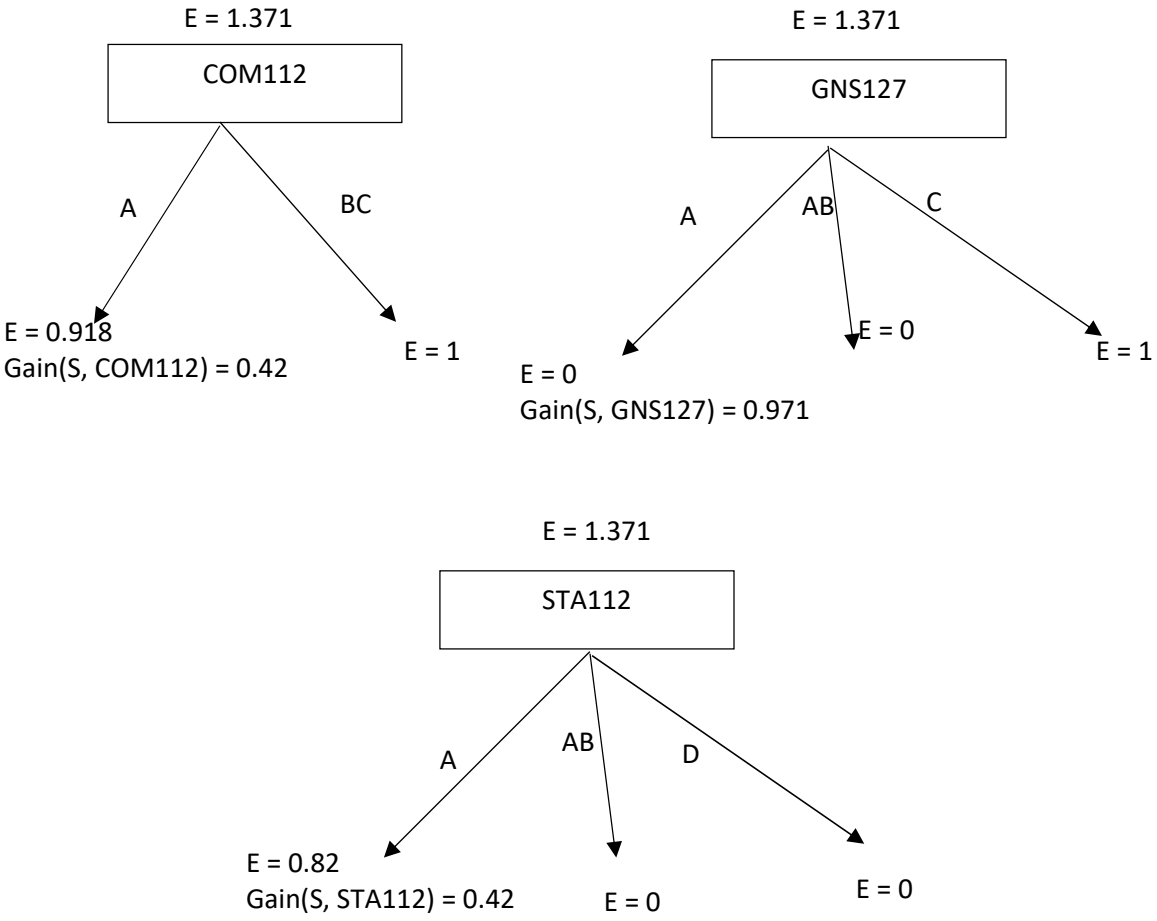


Figure 3.2: Which is the best classifier, COM112, GNS127 or STA112?

J48 decision tree is an implementation of the C4.5 algorithm in WEKA. C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. C4.5 is an extension of Quinlan's earlier ID3 algorithm. C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or on other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen as a leaf to make the decision *(Ji-Wu, J. and Manohar M., 2013)*.

3.2 SYSTEM DESIGN:

This system is designed to reflect the architecture of most decision tree algorithms: it begins with loading the dataset from storage into the decision tree engines, and then the engine builds a model from the training dataset. The model is then being preserved and used to predict new data instances. This is illustrated in *figure 3.3 below*:

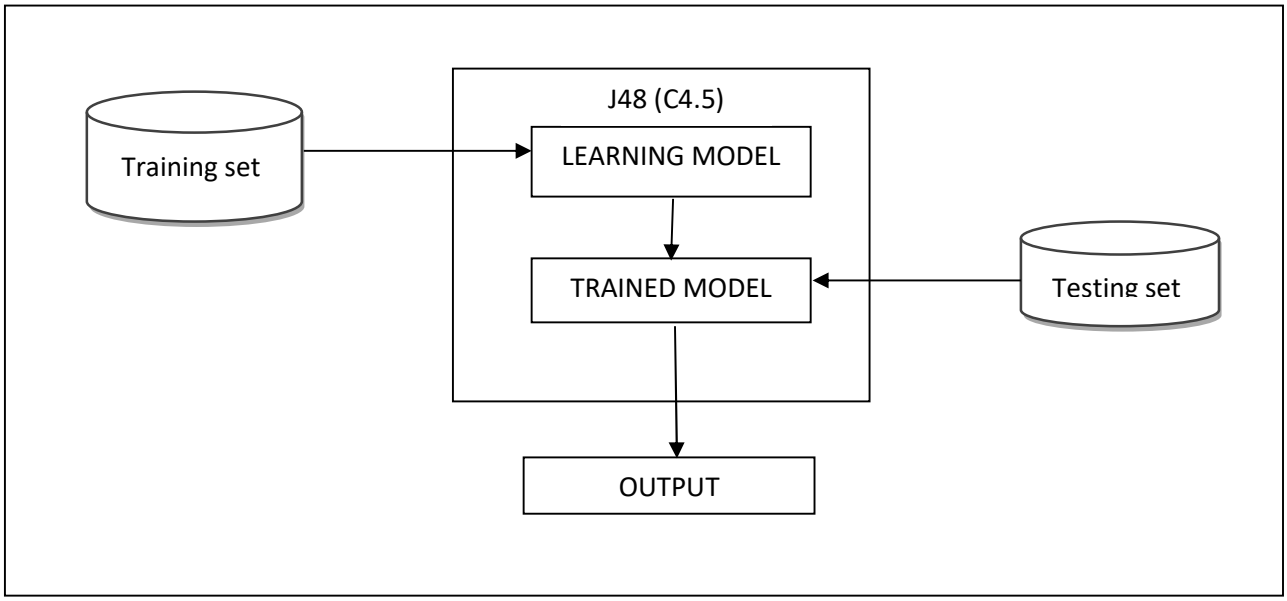


Figure 3.3: Components of the System

The engine room of the system is where the building of the model takes place, i.e. the central core containing the learning model. The components are further explained thus:

3.2.1 Model:

The learning model or learner is the component of the system which collects data sets as input and studies inferences which results from a combination of available factors in the form of attribute values. The learner can also be the segment of the source codes which reads in data from files and learn from the data. The trained model (*figure3.3*) is in practice, not distinct from the learner; the learner learns to become trained. So both learning and trained models forms the same model (J48) that is built to do predictions on test data.

3.2.2 Training and Testing Sets:

These are the rudimentary components. The training dataset is stored in system storage and is read into the system for training. The difference between training and testing dataset is drawn along their class value: both of the datasets are from the same original data source (relational database) except that the testing data sets often have their class values removed or replaced with question mark meaning that they are missing.

3.2.3 Output:

The output is the result that is expected from the system. In this case, it is the model performance information, the predicted class values, the decision tree rules, etc.

3.3 DATASET DESCRIPTION:

The dataset is obtained from students' result board with permission. The students' result contains their grades in each course, and their total grade point average. A total of 270 instances were collected in all. Out of the 270 instances, 260 instances (96%) is used for training while the remaining 10 instances (4%) is for testing.

3.4 FEATURE SET DESCRIPTION:

14 attributes are used in the relation table, which include "com 111", "com112", "com113", "com114", "gls101", "gns127", "ntn111", "ntn112", "otm101", "sta111", "sta112", "gpa", and "class" (the grade); the "gpa" attribute which is also another attribute is to be removed during preprocessing so as not to obscure the objective classification. *Tables 3.1* shows part of the relation table for dataset:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	s/n	com 111	com112	com 113	com 114	gls 101	gns 127	ntn111	ntn 112	otm 101	sta111	sta112	gpa	class
	1	A	A	A	B	A	A	A	A	B	A	A	3.87	distinction
	2	A	A	B	C	A	AB	CD	C	B	B	A	3.08	upper
	3	AB	A	D	BC	A	AB	CD	AB	AB	A	AB	3.39	upper
	4	D	BC	E	E	C	C	E	E	D	AB	D	2.42	pass
	5	BC	BC	CD	C	A	C	BC	C	B	A	A	3.05	upper
	6	AB	A	BC	AB	B	A	BC	A	BC	A	AB	3.54	distinction
	7	A	AB	BC	AB	AB	A	BC	CD	BC	A	A	3.47	upper
	8	A	A	D	AB	A	A	BC	BC	BC	B	A	3.45	upper
	9	C	A	E	E	C	B	D	CD	BC	CD	CD	2.7	lower
	10	BC	A	BC	B	B	A	BC	CD	BC	BC	C	3.14	upper
	11	C	D	CD	D	CD	A	E	F	BC	D	D	2.3	pass
	12	BC	BC	BC	D	A	C	E	D	BC	D	BC	2.8	lower
	13	AB	BC	BC	C	A	A	E	D	B	D	D	2.91	lower
	14	BC	D	CD	CD	AB	AB	E	E	B	E	D	2.59	lower
	15	A	AB	CD	CD	B	AB	E	CD	BC	E	BC	2.95	lower
	16	A	A	AB	AB	A	AB	CD	A	B	A	A	3.7	distinction
	17	C	CD	B	B	A	C	E	E	CD	BC	E	2.57	lower
	19	AB	C	B	B	A	AB	CD	D	C	B	C	3.04	upper
	20	D	A	C	C	AB	B	BC	D	B	C	BC	2.97	lower
	21	E	AB	C	C	AB	B	CD	F	CD	F	BC	2.12	pass
	22	AB	CD	BC	BC	AB	B	E	F	C	D	C	2.47	pass
	23	A	A	A	A	A	A	AB	A	AB	A	C	3.68	distinction
	24	A	A	B	B	A	A	A	A	AB	A	A	3.84	distinction
	25	F	C	E	E	E	F	E	E	F	C	CD	1.79	fail
	26	D	BC	BC	BC	AB	AB	E	F	C	CD	C	2.47	lower
	27	BC	BC	BC	BC	A	B	E	C	BC	A	B	3.12	upper
	28	E	AB	E	CD	BC	C	C	D	BC	C	E	2.53	lower
	29	E	AB	D	F	CD	BC	E	E	C	F	E	2	pass
	30	AB	BC	C	CD	C	BC	BC	AB	C	D	BC	2.95	lower

Table 3.1: Dataset features

3.5 EXPERIMENTAL SETUP:

The dataset is loaded into weka and divided into training set and testing set in the ratio of 80% to 20%, and saved in appropriate format to the system storage. The training set is then read into the C4.5 learning algorithm and used to train the algorithm, and the model is saved. Next the testing set is read into the system and re-evaluated on the training model earlier saved. The result is then printed on display.

Any other number of instances can be saved and re-evaluated each time on the training model to output their predicted class values.

CHAPTER FOUR
SYSTEM IMPLEMENTATION AND RESULT

4.0 INTRODUCTION:

In this chapter, implementation processes of this system are presented and results are evaluated. It discusses the implementation of the C4.5 decision tree algorithm in special software developed for predicting students' grades.

4.1 SYSTEM REQUIREMENTS:

The hardware requirement for the implementation of this system includes:

1. A computer system with not less than 1 GB of available hard disk memory.
2. At least 2GB RAM.
3. A standard Keyboard for input.
4. Mouse for button clicking operations.
5. A display unit.

While software which is necessary for this systems implementation are:

1. Jre – 7 / jdk-7u71 for java or later versions.
2. Netbeans IDE.
3. Microsoft Office package and Text editor for building the dataset (MS Excel), source coding, as well as viewing arff files.
4. WEKA package for automated data preprocessing and model building, etc.

4.2 IMPLEMENTATION DETAILS:

This system implementation was made as simple as possible, yet it accomplishes complex tasks. Java programming language was adopted in coding the program. Java is object-oriented making it easy to use; it is also robust and flexible, as it can interface with many platforms (especially WEKA in this case).

After the dataset preprocessing was completed using WEKA, the rest of the tasks were implemented in three steps, which include (i) reading in the training dataset, (ii) training the model, and (iii) supplying the testing dataset for the model to predict what the *expected* results will be.

4.2.1 Reading in the dataset:

The training and testing datasets are parts of the entire dataset which has been saved separately to be used for different purposes in the system. Even though the original dataset was prepared in MS Excel and saved in comma separated value (csv) file format, the training and testing data are taken from it and saved in attribute relations file format (arff) each. It is this arff dataset that is being used during system execution. The system takes in the training and testing datasets which have to be specified within the system source code. The method of specifying the directory at which the datasets reside is given thus:
"C:/Users/Computer_Name/Location_in_the_computer/Folder_Name/Sub_folder_name/File_name.arff". Example: ***"C:/Users/LILIAN/Desktop/Project stuffs/Data segment/ScoreTrain.arff".*** And for the testing dataset: ***"C:/Users/LILIAN/Desktop/Project stuffs/Data segment/ScoreTest.arff"***

The user should get the basic idea of how this happens, since he or she would need to store his or her dataset in storage in order to get the system to read from it. It is also worth noting that, the system can read data from other file formats other than arff; it can read csv file as well, but arff is preferred in this work.

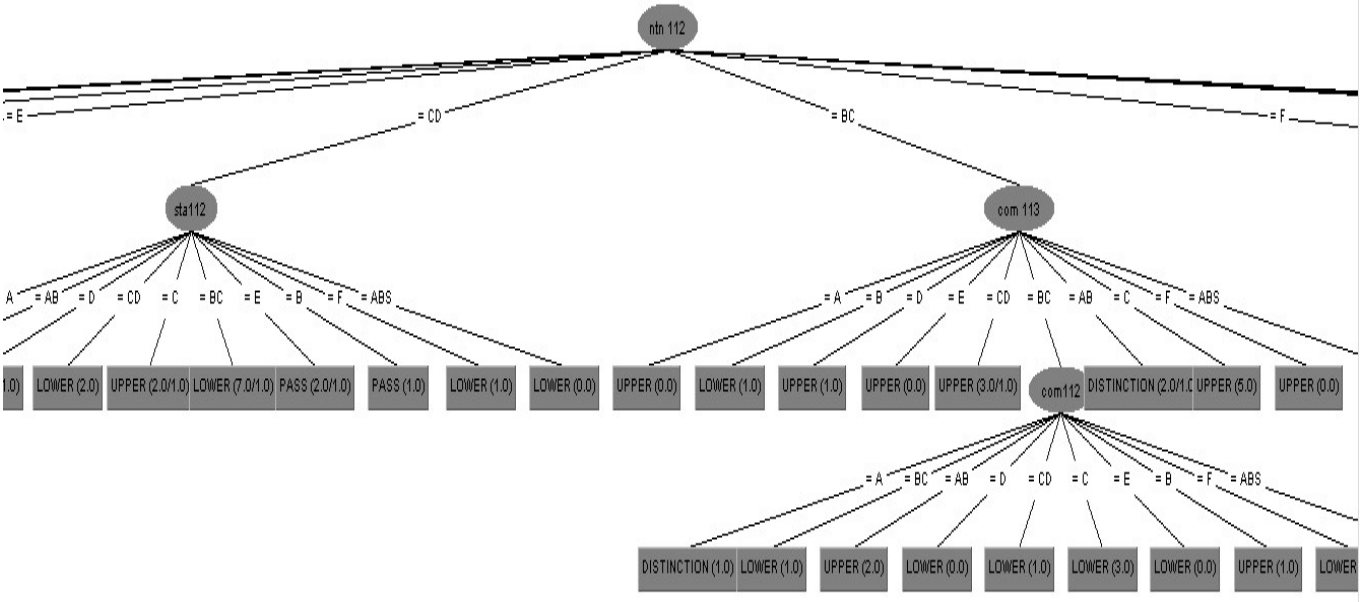
4.2.2 Building the Model:

At this point, the system trains the J48 model on the training examples. The training is inductive and automatic. The decision tree model (J48) learns from each example and its decision attribute (i.e. each combination of factors which give rise to each inference). For instance, an instance in the training data shows *student A who* sat for the following courses and obtained the attached scores, ***com 111 (A), com112 (A), com 113 (B), com114 (C), gls101 (A), gns127 (AB), ntn111 (CD), ntn112 (C), otm101 (B), sta111 (B), and sta112 (A),*** resulting in an overall grade point average (GPA) of ***UPPER CREDIT*** tells the model exactly how the *apportioned* inference (Upper credit) is arrived at. However, the model has to keep reading each instance in the training set, and keep updating the idea it has learned about the relationship between conditions and their resulting inference from a large set of instances in the training set.

The learning process by the model can be evaluated. This system makes it possible for the model to be assessed over its learning process by means of the model performance information which is displayed alongside other results about the system. Also, decision tree algorithms (J48 in this case) are capable of forming *IF-THEN-ELSE* rules from their learning; these rules are also shown in the output as a ***tree***. The tree can equally be represented as a digraph.

Example: the tree-rules below is also being drawn in *figure 4.1:*

```
J48 pruned tree
------------------

ntn 112 = CD
|   sta112 = A: UPPER (4.0)
|   sta112 = AB: UPPER (2.0)
```



| sta112 = D: PASS (2.0/1.0)
| sta112 = CD: LOWER (2.0)
| sta112 = C: UPPER (2.0/1.0)
| sta112 = BC: LOWER (7.0/1.0)
| sta112 = E: PASS (2.0/1.0)
| sta112 = B: PASS (1.0)
| com 113 = E: UPPER (0.0)
| com 113 = CD: UPPER (3.0/1.0)
| com 113 = BC
| | com112 = A: DISTINCTION (1.0)
| | com112 = BC: LOWER (1.0)
| | com112 = AB: UPPER (2.0)
| | com112 = D: LOWER (0.0)
| | com112 = CD: LOWER (1.0)
| | com112 = C: LOWER (3.0)
| sta112 = F: LOWER (1.0)
| sta112 = ABS: LOWER (0.0)
ntn 112 = BC
| com 113 = A: UPPER (0.0)
| com 113 = B: LOWER (1.0)
| com 113 = D: UPPER (1.0)
| | com112 = E: LOWER (0.0)
| | com112 = B: UPPER (1.0)
| | com112 = F: LOWER (0.0)
| | com112 = ABS: LOWER (0.0)
| com 113 = AB: DISTINCTION (2.0/1.0)
| com 113 = C: UPPER (5.0)
| com 113 = F: UPPER (0.0)
| com 113 = ABS: UPPER (0.0)

Figure 4.1: The Centre branches of J48 pruned tree diagram (see also ntn112=CD & ntn112=BC of the tree-rules above)

4.2.3 Testing the Model:

The dataset for the testing is in practice, not taken in at run-time, but the testing dataset directory, as mentioned earlier on, is specified in the program source codes; so the system already knows where the file resides, and picks it up as soon as it reaches the time for it to try its knowledge on new instances which may not have their decision attribute values being specified.

The separate dataset for training and testing gives the user an ample opportunity to view what exactly the system model is doing. This is because, the model will assign class values to those that do not have any values given; this is an advantage over systems which do not use separate datasets for training and testing.

4.2.4 The Output:

The system is built to display results on console. This is because, the results of the system are in text format, which can be so large depending on the number of test instances available. At the same time, results can be displayed for multiple test instances at a time.

The result shown for this system include basic model information like correctly and incorrectly classified instances, Kappa statistics, K&B information score, mean absolute error, total number of instances, etc., as well as confusion matrix, pruned tree, and of course, the prediction on test instances are shown in the output buffer.

4.3 RESULTS AND EVALUATION:

The results of this system collected from the output buffer are displayed piece-by-piece in *figures 4.2, 4.3, and 4.4:*

4.3.1 Pruned Tree:

```
J48 pruned tree                  |   sta112 = A: UPPER
------------------               (4.0)
ntn 112 = CD
```

| sta112 = AB: UPPER (2.0)
| sta112 = D: PASS (2.0/1.0)
| sta112 = CD: LOWER (2.0)
| sta112 = C: UPPER (2.0/1.0)
| sta112 = BC: LOWER (7.0/1.0)
| sta112 = E: PASS (2.0/1.0)
| com 113 = E: UPPER (0.0)
| com 113 = CD: UPPER (3.0/1.0)
| com 113 = BC
| | com112 = A: DISTINCTION (1.0)
| | com112 = BC: LOWER (1.0)
| | com112 = AB: UPPER (2.0)
| | com112 = D: LOWER (0.0)
| | com112 = CD: LOWER (1.0)
| | com112 = C: LOWER (3.0)
| sta112 = B: PASS (1.0)
| sta112 = F: LOWER (1.0)
| sta112 = ABS: LOWER (0.0)
ntn 112 = BC
| com 113 = A: UPPER (0.0)
| com 113 = B: LOWER (1.0)
| com 113 = D: UPPER (1.0)
| | com112 = E: LOWER (0.0)
| | com112 = B: UPPER (1.0)
| | com112 = F: LOWER (0.0)
| | com112 = ABS: LOWER (0.0)
| com 113 = AB: DISTINCTION (2.0/1.0)
| com 113 = C: UPPER (5.0)
| com 113 = F: UPPER (0.0)
| com 113 = ABS: UPPER (0.0)

Figure 4.2: Tree-rules

The pruned tree is an IF-ELSE-THEN statement or rule that can also be built into a digraph tree (see *figure 4.1*). It is formed by the model from its learning from the training information. It uses these rules to search for the value that must match with any new data without class values or whose class values are randomly (carelessly) assumed.

4.3.2 Confusion Matrix / Other Information:

```
=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 12  1  0  1  0 |  a = DISTINCTION
  5 20  2 13  0 |  b = UPPER
  1  1 37 19 14 |  c = PASS
  0 16 16 51  0 |  d = LOWER
  0  0 20  8 23 |  e = FAIL


Model Information
=================

Correctly Classified Instances         143               55      %
Incorrectly Classified Instances       117               45      %
Kappa statistic                          0.4024
K&B Relative Info Score              11277.1187 %
K&B Information Score                  241.8746 bits      0.9303 bits/instance
Class complexity | order 0            557.3957 bits      2.1438 bits/instance
Class complexity | scheme           59267.2761 bits    227.9511 bits/instance
Complexity improvement     (Sf)     -58709.8804 bits   -225.8072 bits/instance
Mean absolute error                      0.2053
Root mean squared error                  0.3798
Relative absolute error                 67.7539 %
Root relative squared error             97.6405 %
Total Number of Instances              260
```

Figure 4.3: Model performance information

Each element in the matrix (*figure 4.3 above*) is a count of instances. Rows represent the true classes, and columns represent the predicted classes. As can be seen, all 12 out of the 14 *DISTINCTION* were correctly classified. 20 out of 37 *UPPER* were correctly classified while 17 of them were wrongly classified. Also, 37 out of 72 *PASS* were correctly classified while 35 of them was wrongly classified as some other grades, and so on.

Worth mentioning too are the correctly and incorrectly classified instances found in the model information screenshot (also in *figure 4.3*); 143 out of the 260 instances (55%) were correctly classified while only 117 of the instances (being 45%) were incorrectly classified.

4.3.4 Prediction:

```
Attribute value                  Predicted class
================================================
A,B,BC,F,B,BC,A,C,A,C,BC,? =====> LOWER
BC,A,CD,E,C,BC,B,BC,B,BC,B,? =====> UPPER
C,E,E,CD,CD,E,E,CD,BC,BC,C,? =====> UPPER
E,A,B,CD,AB,CD,A,C,BC,AB,AB,? =====> UPPER
D,D,CD,D,C,D,C,D,BC,E,D,? =====> PASS
E,CD,B,E,C,CD,BC,BC,C,BC,C,? =====> LOWER
D,E,E,F,F,CD,E,E,E,F,F,? =====> PASS
CD,CD,E,D,CD,BC,E,C,A,BC,CD,? =====> LOWER
D,D,E,E,D,D,E,D,C,CD,E,? =====> PASS
D,D,E,E,D,D,E,D,C,CD,E,? =====> PASS
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 4.4: Predicted results

As can be seen in the screenshot above (*figure 4.4*), the system is also able to output predicted values for the testing dataset instances which were left with missing class values (denoted with *?*).

4.4 DISCUSSION:

The system is a simplified system which however does a complex tasks; Java's robust nature was being explored to achieve this.

The results portray that the system is effective to some extent. The model has a performance that ranks up to 55%. It can also be seen clearly that the model was able to classify each test instances as if it can really see their actual values before they were removed; by comparison, this is apparent as illustrated in *figure 4.5* below:

262	A	B	BC	F	B	BC	A	C	A	C	BC	LOWER
263	BC	A	CD	E	C	BC	B	BC	B	BC	B	LOWER
264	C	E	E	CD	CD	E	E	CD	BC	BC	C	PASS
265	E	A	B	CD	AB	CD	A	C	BC	AB	AB	UPPER
266	D	D	CD	D	C	D	C	D	BC	E	D	PASS
267	E	CD	B	E	C	CD	BC	BC	C	BC	C	LOWER
268	D	E	E	F	F	CD	E	E	E	F	F	FAIL
269	CD	CD	E	D	CD	BC	E	C	A	BC	CD	LOWER
270	D	D	E	E	D	D	E	D	C	CD	E	PASS
271	D	D	E	E	D	D	E	D	C	CD	E	PASS

```
Attribute value                  Predicted class
================================================
A,B,BC,F,B,BC,A,C,A,C,BC,? =====> LOWER
BC,A,CD,E,C,BC,B,BC,B,BC,B,? =====> UPPER
C,E,E,CD,CD,E,E,CD,BC,BC,C,? =====> UPPER
E,A,B,CD,AB,CD,A,C,BC,AB,AB,? =====> UPPER
D,D,CD,D,C,D,C,D,BC,E,D,? =====> PASS
E,CD,B,E,C,CD,BC,BC,C,BC,C,? =====> LOWER
D,E,E,F,F,CD,E,E,E,F,F,? =====> PASS
CD,CD,E,D,CD,BC,E,C,A,BC,CD,? =====> LOWER
D,D,E,E,D,D,E,D,C,CD,E,? =====> PASS
D,D,E,E,D,D,E,D,C,CD,E,? =====> PASS
```

Figure 4.5: Comparison of the original and predicted classes

The screenshots in *figure 4.5* are for a part of the original dataset that was used for the testing set (last 10 instances) and the prediction output for the same last 10 instances. Note that the class values were replaced with question mark to indicate they are missing, and are expected to be predicted. The predicted values are shown on the right after the ===> symbol, and they corresponds exactly to the original class values except for few cases of disparity.

CHAPTER FIVE

5.0 SUMMARY

The various data mining techniques can be effectively implemented on educational data. From the above results it is clear that classification techniques can be applied on educational data for predicting the student's outcome and improve their results. The predictions obtained from the system can help the tutor to identify the weak students and improve their Performance. Since the application of data mining brings a lot of advantages in higher learning institution, these techniques can be applied in the other areas of education to optimize the resources, to predict the number of students who are likely to get admission etc.

5.1 CONCLUSION

A study was conducted on the prediction of students' graduation grades using the C4.5 (J48) decision tree algorithm with student's first semester course grades which were trained to generate a model for predicting students' grades. It could be concluded from the study that the developed system will be very useful in any academic institution for the prediction of students' final graduation grades. This will also help management staff, academic planners and level/course advisers and even parents to properly counsel the students, most especially the academically weak ones in order to improve their performance. Academically sound students could also be advised based on the results of this prediction most especially when they begin to fall below their expected grades at any point in time. The system will generally help students to benchmark their graduation grades from their entry point into the university, thereby helping them to work harder in order to achieve this. Finally, the developed system would help to significantly reduce the overall failure rate in most academic institutions as students can be well guided and counseled.

5.2 RECOMMENDATION

1. The management should take the full advantage of this research by using the program in identifying those students with the likelihood of failure in school, putting measures in place to ensure those identified student are properly guided and counseled, thereby increasing their likelihood of success.
2. The management should make available wireless network in the school to enable all the students access to internet to consult educational materials that will further enhance the knowledge impacted to them by the lecturers
3. The management should provide a conducive learning environment for student to read as this could facilitate the rate at which they assimilate what they are thought in the school

5.3 FUTURE WORK

From the class wise accuracy, it is clear that the true positive rate of the model is 70%.
The future work would include applying data mining techniques on an expanded data set with more distinctive attributes to get more accurate results. Also, a comparative analysis of these results would be carried out based on other experiments results gotten from using other types of decision tree algorithms such as CHAID and CART. Design an interface for the model, incorporation of decision tree machine.

REFERENCE

AI-Radaideh O.A., AI-Shawakfa E.W. and AI-Najjar M.I (2006). Mining student's data using decision tree, International Arab conference on information technology (ACIT 2006) yarmouk University, Jordan

Baradwaj, Brijesh Kumar and Saurabh pal (2012) . mining Educational Data to analyze students performance. arXiv preprint arXiv: 1201.3417.

Esposito F., Malerba D. and Semeraro G.,(1997). A Comparative Analysis of Methods for Pruning Decision Trees. EEE Transactions on Pattern Analysis and Machine Intelligence, 19(5):476-492.

Fayyad U., and Irani K. B., (1992). The attribute selection problem in decision tree generation. In proceedings of Tenth National Conference on Artificial Intelligence, pp. 104–110, Cambridge, MA: AAAI Press/MIT Press.

Hijazi S. T. and Naqvi,R. S., (2006).“Factors affecting student‟s performance: A Case of Private Colleges”, Bangladesh *e-Journal of Sociology, Vol. 3*, No. 1.

Hijazi, S., & Naqvi, R. (2013). Factors affecting students' performance: A Case of Private Colleges. *Bangladesh e-Journal of Sociology, Vol. 3, No. 1*.

Khan, Z. (2015). Scholastic achievement of higher secondary students in science stream. *Journal of Social Sciences, Vol. 1, No. 2*, pp. 84-87.

Kovačić Z. J. (2010) "Early prediction of student success: Mining students enrolment data,"Presented at the Informing Science & IT Education Conference.

Luan, Jing (2002) Data mining and it's application in higher education. New direction for instution research, No.133, springer.

Ogunde A.O and Ajibade D.A. (2014). A data mining system for predicting university student's graduation grades using ID3 decision tree algorithm, Journal of computer science and information technology, vol. 2, No. 1, pp.21-46.

Oladipupo, O. O., & Oyelade, O. J. (2008). Knowledge Discovery from Student's Repository: Association rule mining Approach. *International Journal of Computer Science & Security (IJCSS) Vol.4 Issue 2*, pp 199-206.

Pandey U. K. and Pal S. (2011) “Data Mining: A prediction of performer or underperformer using classification”, (IJCSIT) International Journal of Computer Science and Information Technology, Vol. 2(2), pp.686-690, ISSN:0975-9646.

Quinlan J. R., (1993) *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers,.

Rokach , Lior Maimon, D (2008). Data mining with decision tree theory and applications, World Scientific pub.co.inc

Samrat, S., & Vikesh, K. (2012). Classification of Student’s data Using Data Mining Techniques for Training & Placement Department in Technical Education. *International Journal of Computer Science and Network (IJCSN), Volume 1, Issue 4*.

Shamuga Piya K. and Senthil Kumar A.V (2013). Improving the student’s performance using education data mining. “international journal for advanced networking and application”.vol.04 Issue :04 pg 1680-1685.

Tsai, C.F, Tsai C.T ,hung C.S, Hwang ps (2011). Data mining techniques for identifying student’s risk of failing a computer proficiency test requires for graduating. Australian Journal of educational technology 27(3): 481-498

Yadav, S., Bharadwaj, B., & Pal, S. (2011). Data Mining Applications: A comparative study for Predicting Students’ Performance. *International Journal of Innovative Technology and Creative Engineering (IJITCE), Vol. 1, No. 12,* pp. 13-19.

APPENDIX I

```
/*
* To change this license header, choose License Headers in Project Properties.
* To change this template file, choose Tools | Templates
* and open the template in the editor.
*/
package weka.api;
import java.util.Random;
import weka.classifiers.Evaluation;
import weka.core.Instance;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;
import weka.classifiers.trees.J48;
import weka.api.ClassifyInstance;
/**
*
* @author lilian
*/
public class ClassifierPrediction extends javax.swing.JFrame {
/**
* Creates new form ClassifierPrediction
*/
public ClassifierPrediction() {
initComponents();
}
```

```
/**
* This method is called from within the constructor to initialize the form.
* WARNING: Do NOT modify this code. The content of this method is always
* regenerated by the Form Editor.
*/
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {
jPanel1 = new javax.swing.JPanel();
jPanel2 = new javax.swing.JPanel();
jButton1 = new javax.swing.JButton();
jMenuBar1 = new javax.swing.JMenuBar();
jMenu1 = new javax.swing.JMenu();
jMenu2 = new javax.swing.JMenu();
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
jPanel1.setBackground(new java.awt.Color(102, 102, 255));
jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "PREDICTION OF STUDENT
PERFORMANCE USING DECISSION TREE ALGORITHM(J48)",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new java.awt.Font("Tahoma",
3, 14), new java.awt.Color(51, 102, 255))); // NOI18N
jPanel2.setToolTipText("");
jButton1.setText("Predict");
jButton1.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
jButton1ActionPerformed(evt);
}
```

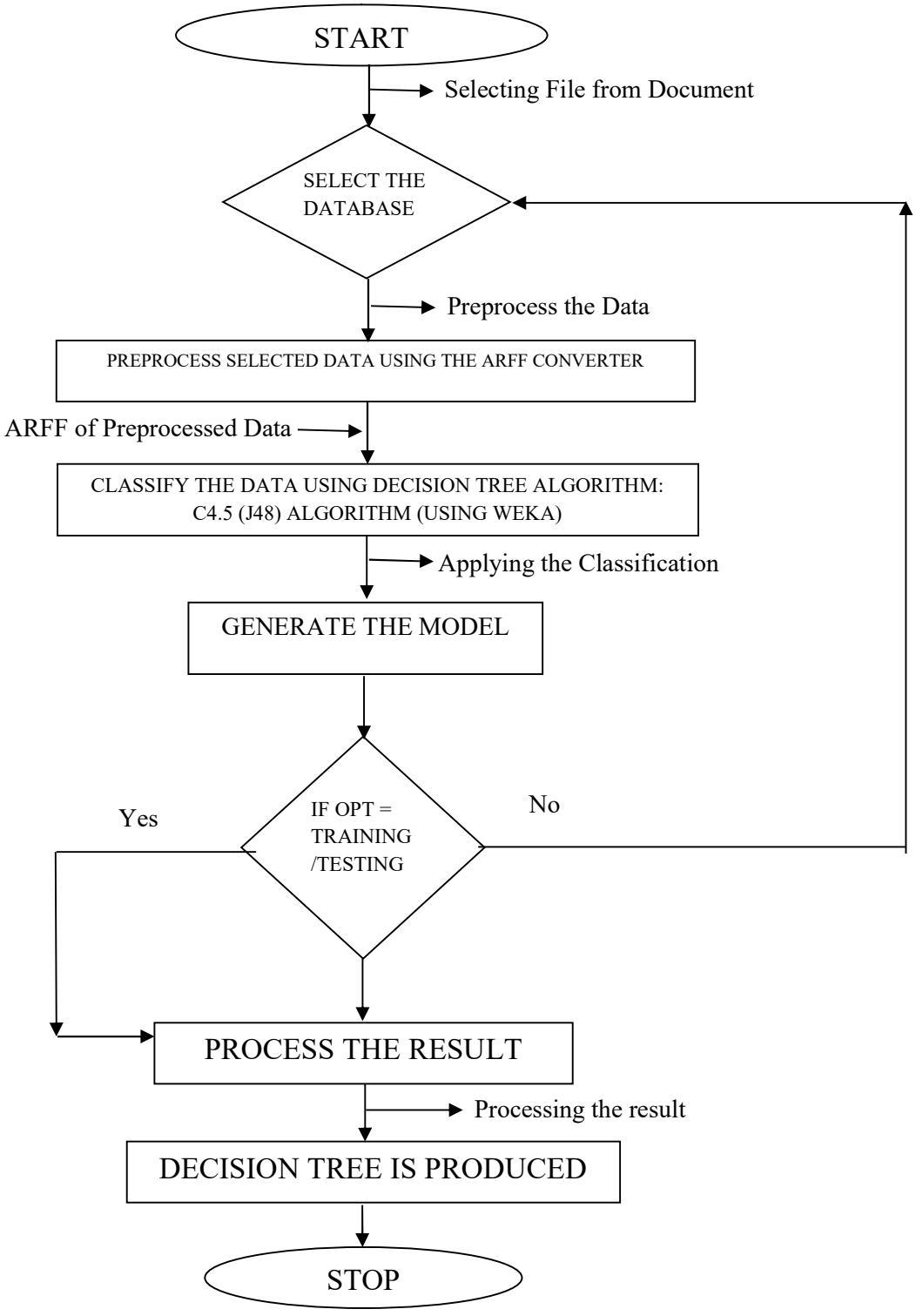
```
});

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(0, 528, Short.MAX_VALUE))

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(0, 80, Short.MAX_VALUE))

);javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()
```

```
.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(0, 161, Short.MAX_VALUE))

);

jMenu1.setText("File");

jMenuBar1.add(jMenu1);

jMenu2.setText("Edit");

jMenuBar1.add(jMenu2);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(0, 200, Short.MAX_VALUE))

);

pack();
```

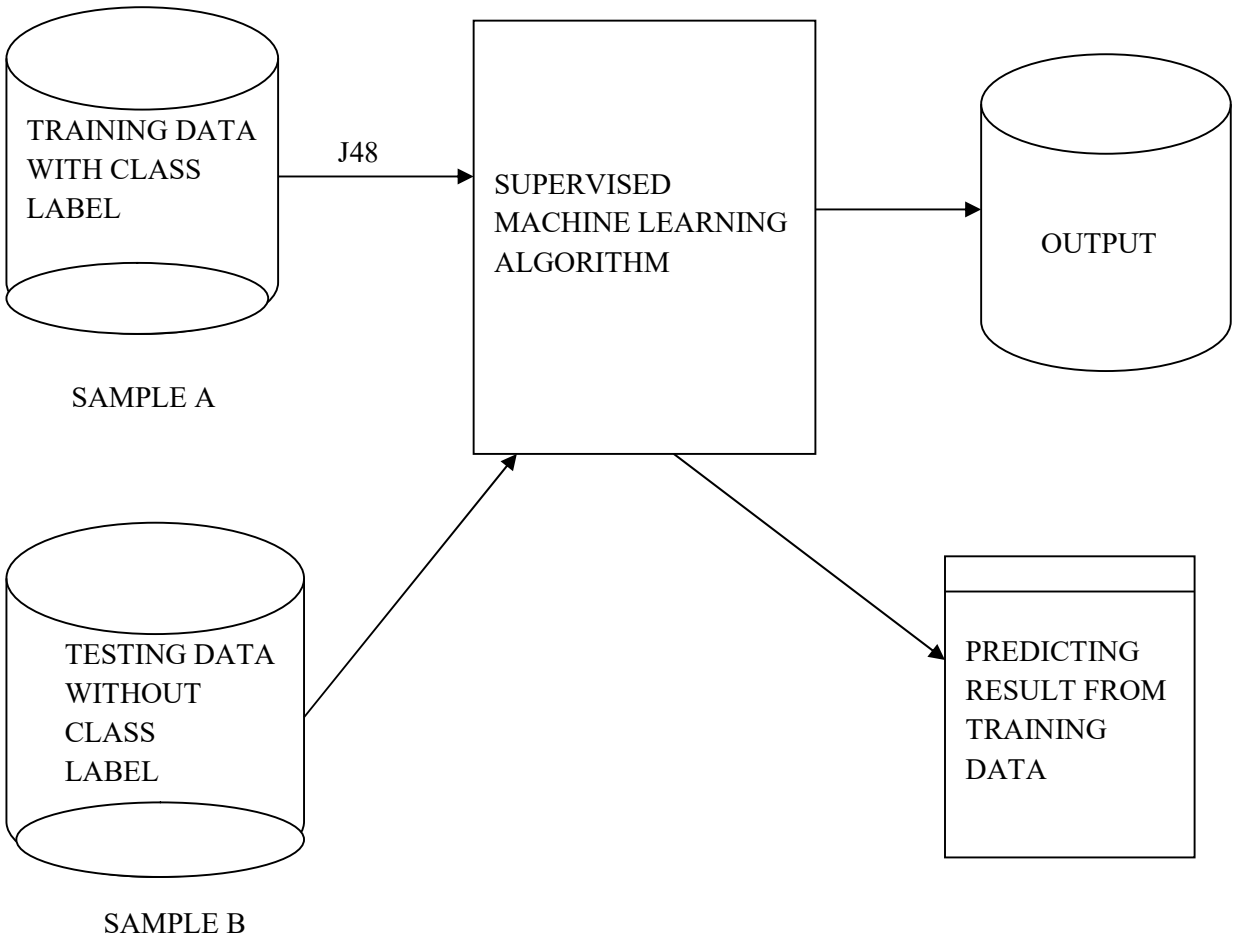
```
}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

new ClassifyInstance().getClass();

// TODO add your handling code here:
}

/**

* @param args the command line arguments

*/

public static void main(String args[]) {

/* Set the Nimbus look and feel */

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(ClassifierPrediction.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);

} catch (InstantiationException ex) {
```

```
java.util.logging.Logger.getLogger(ClassifierPrediction.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(ClassifierPrediction.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(ClassifierPrediction.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);

}

//</editor-fold>

/* Create and display the form */

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new ClassifierPrediction().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JMenu jMenu1;

private javax.swing.JMenu jMenu2;

private javax.swing.JMenuBar jMenuBar1;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

// End of variables declaration

}
```

APPENDIX II



System flow chart for the training/testing of data



System flow chart