DESIGN AND IMPLEMENTATION OF FACULTY SUPPORT SYSTEM TO REDUCE COURSE DROPOUT RATES

A Thesis Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Rohith Jidagam

May 2016

DESIGN AND IMPLEMENTATION OF FACULTY SUPPORT SYSTEM TO REDUCE COURSE DROPOUT RATES

_____

Rohith Jidagam

APPROVED:

_____

Dr. Christoph F. Eick
Department of Computer Science

_____

Dr. Nouhad Rizk
Department of Computer Science

_____

Dr. Tammy Tolar
Department of Psychological Health and Learning Sciences

_____

Dr. Weidong Shi
Department of Computer Science

_____

_____

Dean, College of Natural Sciences and Mathematics

Acknowledgements

First and foremost, I would like to thank and convey my sincere gratitude to my graduate advisors, Dr. Christoph F. Eick, and Dr. Nouhad Rizk for giving me an opportunity to work in the data mining team. This opportunity eventually led to the work done in this thesis. Their advice, guidance, and support were instrumental throughout the course of this study. It is difficult to envisage this work without their help and insight.

I would also like to thank Dr. Weidong Shi and Dr. Tammy Tolar for agreeing to be a part of my Thesis committee. My appreciation also goes to the Educational Data Mining team, led by Dr. Nouhad Rizk, for their continuous efforts in obtaining the real datasets, conducting survey and collecting information.

I would like to specially acknowledge the significant role played by my friends at the University of Houston, for making sure that I ensconce in my personal as well as academic life at the University of Houston. Also, I would like to take this opportunity to thank my family and my girlfriend for encouraging and motivating me in all my endeavors. Finally, I would like to express my deepest gratitude towards my parents and my brother for their unparalleled love and support. They continue to be the greatest source of inspiration for me.

DESIGN AND IMPLEMENTATION OF FACULTY SUPPORT SYSTEM TO REDUCE COURSE DROPOUT RATES

An Abstract of a Thesis

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Rohith Jidagam

May 2016

Abstract

The primary goal of educational systems is not only to provide quality of education but also to make sure that students graduate with a strong academic standing. One specific challenge that universities face is high course drop rates. An early prediction of students' failure may help to identify students who need special attention to reduce course drop rates by providing appropriate interventions, such as continuous mentoring and conducting review sessions. To address this problem, a new framework called *Faculty Support System (FSS)* is proposed that learns different classification models to predict student course performance based on his/her attendance, and performance in assignments, quizzes, in-class group projects, and exams. The investigated approaches for this task include Naïve Bayes, Multi-Layer Neural Networks, Decision Trees, and Random Forests. Next, using these models potentially low-performing students will be selected for interventions. Finally, data related to the performance of particular interventions and the employed classification models will be collected at the end of the semester. The proposed FSS framework is evaluated on two different real-world datasets that were obtained during two different semesters for two Computer Science courses at University of Houston, Texas.

Our experimental results reveal that Multi-Layer Neural Networks performed the best, and the proposed modelling approach can efficiently identify students at risk, and recommend interventions to enhance their performance before the final exam of the semester. The evaluation of different classifiers on educational datasets gave some insights into how different data mining algorithms predict student performance and enhance student retention. Moreover, the experiments created valuable data about the performance of different interventions.

Table of Contents

1. Introduction 1

2. Background & Related Work 5

2.1 Naïve Bayes 5

2.2 Decision Trees 6

2.3 Random Forests 6

2.4 Multiple Layer Neural Network 7

2.5 Related Work 10

3. Faculty Support System 14

3.1 Preparatory Analysis for FSS 15

3.2 Faculty Support System Architecture 17

3.2.1 Pre-Data Mining Phase 18

3.2.2 Intervention Phase 19

3.2.3 Post-Data Mining Phase 24

4. Technical Details of FSS 26

4.1 A Naïve Bayes Approach for Future Grade Prediction 26

4.1.1 Final Grade Prediction using Naïve Bayes 29

4.2 Evaluation Methods and Metrics 36
4.2.1 Evaluation Methods 36
4.2.2 Evaluation Metrics 37
4.3 Weka Workbench 39
4.4 Parameter Selection for Different Classification Models 40
4.4.1 Parameter Selection of Multiple Layer Neural Networks 40
4.4.2 Parameter Selection of Decision Trees 42
4.4.3 Parameter Selection of Random Forests 44
5. Experimental Results 46
5.1 Objectives and Overview of Experiments 46
5.2 Data Sets 47
5.2.1 Data Pre-Processing 47
5.3 Experiment 1: Prediction of Final Examination Grade Using Naïve Bayes 50
5.4 Experiment 2: Finding the Best Performing Classifier 51
5.5 Exploratory Data Analysis of Intervention Phase Data 55
5.5.1 Determining the Proportion of Students in each Predicted Category 56
5.5.2 Determining the Proportion of Students in each Category Based on Gender 57
5.5.3 Determining the Proportion of Students in each Category Based on Ethnicity 57
5.5.4 Determining the Proportion of Students in each Category Based on Faculty Support 58

5.5.5 Determining the Proportion of Students in each Category Based on Learning Method..... 59

5.6 Experiment 3: Evaluating the pre-Data Mining Classification Model..... 60

6. Conclusion and Future Work 63

References 65

List of Figures

Figure 1.1: Multiple Layer Neural Network 8

Figure 3.2: The Faculty Support System 18

Figure 4.1: Architecture of Implemented Naive Bayes Classifier 34

Figure 4.2: ROC curve with performance comparison..... 39

Figure 5.1: Histogram of proportion of predicted students in each category..... 56

Figure 5.2: Histogram of proportion of students in each category based on gender 57

Figure 5.3: Histogram of proportion of students in each category based on ethnicity..... 58

Figure 5.4: Histogram of proportion of students in each category based on faculty support.....59

Figure 5.5: Histogram of proportion of students in each category based on learning method.... 60

List of Tables

Table 3.1: Social factors collected from the students 22

Table 3.2: Structural factors collected from the students 22

Table 3.3: Institutional factors collected from the students 23

Table 3.4: Personal factors collected from the students 23

Table 3.5: Learning factors collected from the students 24

Table 4.1: Frequencies with conditional probabilities for attribute attendance 30

Table 4.2: Frequencies with conditional probabilities for attribute quizzes 30

Table 4.3: Frequencies with conditional probabilities for attribute assignments 31

Table 4.4: Frequencies with conditional probabilities for attribute class projects 31

Table 4.5: Frequencies with conditional probabilities for attribute exams 31

Table 4.6: Prior Probability of response variables 32

Table 4.7: Test Instance with no class label 32

Table 4.8: Confusion matrix of a 2-class classification model 38

Table 4.9: Performance metrics 38

Table 4.8: Parameters selection in Weka Tool for Multiple Layer Neural Networks 41

Table 4.9: Parameters selection in Weka Tool for Decision Trees 42

Table 4.10: Parameters selection in Weka Tool for Random Forests 44

Table 5.1: Data sets used to evaluate FSS model 47

Table 5.2: Different variables and their values used in FSS model 49

Table 5.3: Records of final data set after pre-processing 50

Table 5.4: Predicted number of students in each category for dataset 1 51

Table 5.5: Predicted number of students in each category for dataset 2 51

Table 5.6: Confusion matrix for different classifiers using dataset 1 52

Table 5.7 Performance metrics for classifiers using 10-fold cross validation on dataset 1..... 53

Table 5.8: Confusion matrix for different classifiers using dataset 2 54

Table 5.9 Performance metrics for classifiers using 10-fold cross validation on dataset 2..... 55

Table 5.11: Decomposition of students based on predicted and actual grades on dataset 1 61

Table 5.12: Decomposition of students based on predicted and actual grades on dataset 2 61

1. Introduction

Educational Data Mining (EDM) [1] is an emerging discipline, concerned with developing methods for analyzing different types of data that come from educational settings, and to employ data mining methods to better understand students and the setting which they learn in. The increase in instrumented educational software, as well as databases of student test scores, has created large repositories of data reflecting how students learn. EDM focuses on collection, archiving, and analysis of data related to students' learning and assessment. EDM is poised to leverage an enormous amount of research from the data mining community and apply that research to educational problems in learning, cognition, and assessment. As EDM is concerned with the application of Data Mining and Machine Learning to educational datasets, different methods like classification, clustering, regression, factor analysis, association rule mining and sequential pattern mining can be applied to educational data. For example, researchers have used educational data mining to:

- Detect affect and disengagement
- Guide student learning efforts
- Develop or refine student models
- Measure the effect of individual interventions

- Assess approaches for improved teaching support
- Predict student performance and behavior

However, these techniques when integrated into a model could achieve greater use and bring wider benefits to the university management. Moreover, there is a need to develop standard data formats, so that they can be more easily share data and conduct meta-analysis across tutoring systems, and determine which data mining techniques are most appropriate for the specific tasks in EDM, and how these techniques can be used on a wide scale.

In the recent years, extensive research has been conducted in different educational institutions to identify the factors that cause students to drop a course, and factors that determine student performance in a course. However, it is a difficult task to find such factors as they are hidden in large repositories of students' data. To identify such factors has been a major focus of EDM.

In this research, we propose a new framework called *Faculty Support System (FSS)* that would enable an instructor to predict a student's performance in a course. The rationale behind FSS is to build an automated system that will help in the early prediction of students' failure and provide different types of intervention such as continuous mentoring and conduct review sessions to reduce drop-out rate and enhance students' performance in the course.

Moreover, data related to the performance of particular interventions and the employed grade prediction models will be collected and summarized at the end of the semester. In this research, we used educational data of students in the computer science department at University of Houston, Texas, to predict students' final exam grades and to evaluate FSS. The proposed framework, FSS employs a modeling technique that can be used within any educational institution as a way of increasing awareness of focusing on low performing students, of combating the dropout crisis and of investing in strategies to ensure students success. The application of FSS in the educational system is an interactive cycle of prediction, intervention, verification, and validation. The proposed framework will generate preliminary but useful results that can advance our knowledge of how to appropriately conduct educational data mining projects and extend the field in new directions. The detailed FSS framework is explained in chapter 3.2.

In this research, we used different types of data mining algorithms, such as Naive Bayes, Multiple Layer Neural Network, Random Forests, and Decision Trees to build classification models on different sets of students' data. Another goal of this research is to compare these algorithms, to assess their performance, and to identify the best performing algorithm. To compare and improve classified models that estimate knowledge of students, it is necessary to use metrics that measure the quality of employed models. Experimental results will be presented that compare the four employed classification approaches, based on different performance metrics.

The rest of the thesis is organized as follows. In Chapter 2, the background of the algorithms used in this research and the relevant educational data mining literature are discussed. The Faculty Support System Framework is explained in detail in Chapter 3. Chapter 4 discusses how different data mining algorithms are used in Faculty Support. Experimental results that evaluate the FSS and the employed classification algorithms are presented in Chapter 5. Finally, the thesis is concluded in Chapter 6 by providing the summary of the research findings and by discussing future work.

2. Background & Related Work

This chapter provides background knowledge concerning the classification models that are used in this research (Sections 2.1 through 2.4) and discusses related work in the educational data mining field in Section 2.5.

2.1 Naïve Bayes

Naïve Bayes [2] is a classification algorithm based on Bayes rules of simple conditional probability that estimates the likelihood of class given a set of input data. Naïve Bayes assumes that the effect that an attribute plays on a given class is independent from the values of other attributes. Bayesian classifiers are popular classification algorithms due to their simplicity, computational efficiency, and good performance for real-world problems. Bayesian classifiers are statistical classifiers, as they predict the probability that a given sample belongs to a particular class. The technical details and implementation of the Naïve Bayes approach are explained in Chapter 4.1.

2.2 Decision Trees

Decision trees [3] are powerful and popular tools for classification. A decision tree partitions the training data set into disjoint subsets so that each subset is as pure as possible – most examples belong to the same class. The learning of a tree relies on a

divide-and-conquer strategy that recursively partitions the data to produce the tree. In the beginning, all the examples are at the root. As the tree grows, the examples are subdivided recursively. The algorithm stops when all the training examples in the current data are of the same class, or when the remaining examples can no longer be split. In tree learning, each successive recursion chooses the best attribute to partition the data at the current node according to the values of the attributes. The best attribute to split is selected minimizing an objective function; popular objective functions include gini index, information gain, and information gain ratio. The decision tree learning algorithm is a greedy algorithm that does not backtrack: once a node test is created, it will not be revisited.

2.3 Random Forests

Random forests [4] use an ensemble of simple decision trees as classification models. An ensemble approach constructs a set of base classifiers from training data and performs classification by taking a vote from the predictions made by each base classifier.

Random Forests use an ensemble learning approach called boosting to obtain a set of base classifiers by generating different training sets from the training data using weighted sampling with replacement approach. Boosting assigns a weight to each training example and which adaptively changes at the end of each boosting round. AdaBoost [5] is the most popular boosting algorithm. The AdaBoost algorithm initially assigns equal weights to the training examples. A training set is generated by drawing examples from the training set.

Next, a classifier is induced from the training set and used to classify all the examples in the original data. The weights of the training examples are updated at the end of each boosting round. Examples that are classified incorrectly will have their weights increased while those that are classified correctly will have their weights decreased. This forces the classifier to focus on examples that are difficult to classify in subsequent iterations which is important to obtain a diverse set of base classifiers.

2.4 Multiple Layer Neural Network

Multiple Layer Neural Networks [6] are classification algorithms based on Neural Networks. A Neural network is a set of connected input/output units and each connection has a weight associated with it. During the learning phase, the network learns by adjusting weights, reducing the error to predict the correct class labels for the examples in the training set.

In a multilayer neural network, the neurons are aligned in layers and any two adjacent layers the neurons are connected with weighted edges. A typical multilayer neural network consists of at least three layers of neurons, including one input layer, one or more hidden layers, and one output layer. The number of neurons in the input layer is determined by the number of input attributes, the number of neurons in the output layer is determined by the number of classes. As for a hidden layer, the number of neurons is a design issue. If too few neurons are used, the model will not be able to learn complex decision boundaries. On the other hand, if too many neurons are used the generalization

capability of the model will decrease and lead to overfitting. An example of multiple layer neural networks with one input layer, one hidden layer, and one output layer is seen in Figure 2.1.

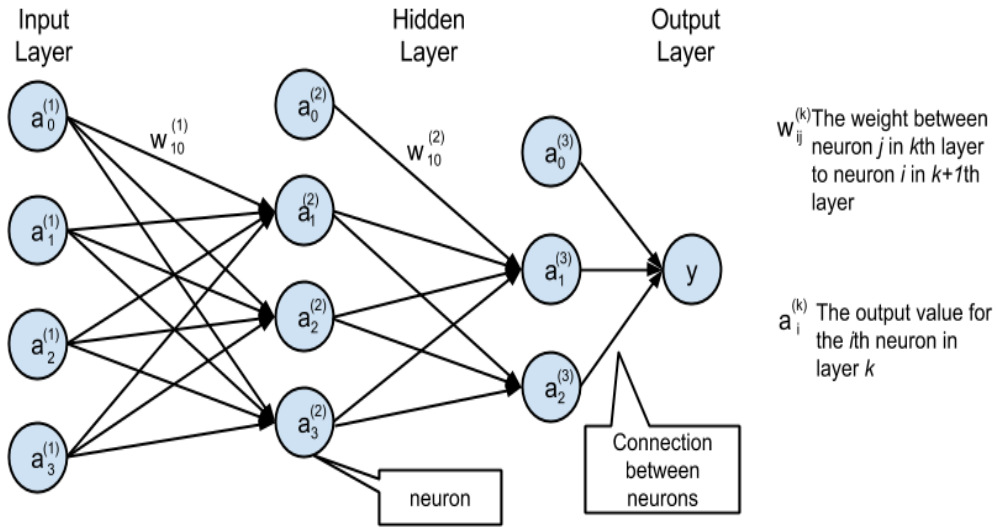

Figure 1.1: Multiple Layer Neural Network

In general, we use the multiple layer neural network by feeding the input features to the input layer and get the result from the output layer. The results are calculated in a feed-forward approach, from the input layer to the output layer. For each layer except the input layer, the value of the current neuron is calculated by taking the linear combination of the values output by the neurons of the previous layer, where the weight determines the contribution of a neuron in the previous layer to the neuron of the subsequent layer, given by equation 2.1.

$$z = w^T a = \sum_{i=1}^{k} w_i a_i \tag{2.1}$$

Obtaining the linear combination result, z, a nonlinear squashing or activation function is used to constrain the output into a restricted range. Typically, different activation functions are used by neural networks such as unit step, sigmoid, and gaussian. The role of the activation function in a neural network is to produce a non-linear decision boundary through non-linear combinations of the weighted inputs.

A key feature of neural networks is an iterative learning process in which records from the dataset are presented to the network one at a time, and the weights associated with the input values are adjusted each time. After all cases are presented the process often starts over again. During this learning phase, the network learns by adjusting the weights to predict the correct class label of input samples. Once a network has been structured for a particular application, that network is ready to be trained. To start this process, the initial weights are chosen randomly. Then the training, or learning of the neural network, begins. The network processes the records in the training data one at a time, using the weights and functions in the hidden layers, then compares the resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights so that the error is reduced. During the training of a network, the same set of data is processed many times as the connection weights are continually refined until some convergence is accomplished.

2.5 Related Work

EDM has gained increased interest in recent years. Predicting students' performance has been an issue studied previously in educational data mining research in the context of student attrition. In this section, we briefly discuss approaches that most closely relate to our proposed approach. Han and Kamber [7] describe an approach that uses association rule mining to identify the factors influencing the success of students, and decision tree models are used to predict student performance. Priya et al. [8] applied a decision tree classification technique to the end of semester mark leading to the discovery of valuable knowledge that can be used to improve students' performances. Galit [9] presented a case study that used students data to analyze their learning behavior to predict the results and to warn students at risk before their final exams.

Goyal and Vohra [10] showed that if data mining techniques such as clustering, decision tree, and association analysis were applied to higher education processes, it helped to improve student performance, their life cycle management, selection of courses, and predict student retention rate. Surjeet Kumar et al. [11] used decision tree classifiers to predict student drop-out rates. Pathan et al. [12] developed a decision tree based mining model to improve students' programming skills in the C language where they collected data from 70 students of Structured Programming Language (SLP) course and generated two datasets.

Minaei-Bidgoli [13] used a combination of multiple classifiers to predict student grades based on features extracted from logged data in an education web-based system. Furthermore, they tried to learn an appropriate weighting of the features using a genetic algorithm approach, which further improved prediction accuracy. Pittman [14] performed a study to explore the effectiveness of data mining methods in identifying success or failure of students in a course and found that SVM-based approaches and random forests methods accomplished the highest accuracies. Romero et al. [15] compared different, data mining methods for classifying students based on their Moodle (e-learning system) usage data and the final marks obtained in their respective programs and they observed that decision trees were the most suitable approach for this task. Nguyen et al. [16] compared the accuracy of the decision tree and Bayesian network models for predicting the academic performance of undergraduate and postgraduate students and observed that decision tree models accomplished better accuracy than a Bayesian network classifier.

Al-Radaideh et al. [17] proposed a classification model to enhance the quality of the higher educational system by evaluating students' data that may affect the students' performance in courses. They used three different classification methods ID3, C4.5, and the Naïve Bayes. The results indicated that the decision tree model had better prediction accuracy than the other models. Muslihan et al. [18] compared artificial neural network and the combination of clustering and decision tree classification techniques for

predicting and classifying student's academic performance and found that combination of clustering and decision tree techniques achieved high accuracy. Ramaswami and Bhaskaran [19] have constructed a predictive model called CHAID with 7-class response variables by using predictive variables obtained through feature selection, to evaluate the academic achievement of students at higher secondary schools in India. The accuracy of the present model was compared with other models, and it has been found to be satisfactory. Tripti et al. [20] used J48 decision trees and random forests to predict the performance of Masters of Computer Applications (MCA) students in their work and observed that random tree has better accuracy, and it consume less time than the J48 decision tree algorithm.

Our proposed approach is different from what has been proposed in the following aspects. The previous work discovers valuable knowledge that a faculty or an educational system can use to improve student performance, whereas our proposed approach identifies students that are more likely to fail so that corrective measures can be taken before failure happens. Moreover, the proposed post-data mining phase measures the effectiveness of the predicted results by comparing them with real outcomes. Additionally, the previous work compares different algorithms without focusing on different performance metrics. The important aspect of this research is to use a smart combination of performance metrics to examine the efficiency and performance of any educational model, such as the Faculty Support System, and provide a more

comprehensive evaluation of data mining algorithms. Most of the available studies in the literature use only accuracy as a unique criterion to compare the performance of data mining algorithms and ignore the other performance metrics.

3. Faculty Support System

The monitoring and support of the students in a course is a problem that is considered crucial for many educational institutions. The specific objective of the proposed research is to reduce the drop-out rate of the students in a course and support them in obtaining good grades. This is achieved by identifying the low scoring students early in the semester. The identification process is done with the use of data mining techniques that predicts the students' final exam scores based on the course performance and other factors. Next, the identified students are exposed to interventions, such as review sessions, continuous mentoring, and rigorous training. The additional tutoring gives low-scoring students a chance to improve their performance and complete the course with a good grade. Also, in the intervention phase, the students' learning and behavioral characteristics are collected in the form of a survey to analyze the factors that affect student performance. In particular, the research goals are as follows:

1. Generation of datasets of predictive variables from the students' records and their on-going performance in the course.
2. Construction of a model to predict low-scoring students.
3. Conduct a survey of students in the course, to obtain their behavioral and social characteristics.

4. Perform an exploratory data analysis and identify factors that affect student performance.
5. Evaluation and enhancement of the constructed classification models.

The research goals are achieved by proposing a framework called *Faculty Support System (FSS),* which will be explained in detail in section 3.2. In the process of designing and implementing Faculty Support System Framework, a preparatory analysis phase was conducted which is explained as follows.

3.1 Preparatory Analysis for FSS

During this phase, an extensive literature review was performed to study the existing problems at higher educational institutions that have been addressed by the application of data mining techniques and methods in previous research projects. In the meanwhile, formal interviews with representatives of the university management, faculty, and departmental levels were conducted. The major objective of these interviews was to find out the specific problems related to the students' performance at the university, which has not yet been solved but is considered crucial to the improvement of student performance as well as university performance. Some insights were gathered from informal talks with lecturers, students, and representatives of the university administrative staff. Based on the outcomes of these activities, the project goals, and objectives, and the main research questions were formulated.

The survey results showed most of the faculty and university management reported that the high course drop-out rate is a serious problem for the University of Houston. There are three reasons for dropping a course: First, students enrolling for difficult courses are not able to handle the course workload. Second, the students take too many courses in a semester. Finally, there is a lack of intervention to help students with difficulties in completing courses. The current research focuses on these factors and provides a solution to reduce the drop-out rate.

The main challenge for modern universities is to deeply analyze students' performance, to identify their uniqueness, and to build a strategy for further development and future actions. The goals and objectives of the research are achieved by identifying patterns in the educational data that are useful for predicting student performance based on internal assessments like class projects, exams, quizzes, attendance, and assignments. The university management would like to know which features are the strongest predictors of course performance. Moreover, universities are also interested in collecting data on their students, course performance and factors that potentially affect student success.

During this phase, we reviewed different methods for building a model that would classify the students into the First, Second, Third, and Fail categories; depending on their course performance and the data collected in the survey.

3.2 Faculty Support System Architecture

The architecture of the Faculty Support System is shown in Figure 3.1. The Faculty Support System model employs 3 phases.

1. Pre-data mining phase
2. Intervention phase
3. Post-data mining phase

In the first phase, the pre-data mining phase, a classifier is learned with the data collected from the previous course to predict the course performance of the students. The pre-data mining phase predicts a possible final grade for each student in the first 4-6 weeks of the semester before the course drop date. The main goal of this phase is to identify students that risk failure and to expose them to additional interventions. The intervention phase is where the Center of Excellence team collects learning and behavioral data on the students. The Center of Excellence team analyzes those characteristics and selects a specific intervention catered for each student. Possible interventions are review sessions, counselling and mentoring with the goal to enhance their performance. In the post-data mining phase, different models are learned, evaluated, and compared that classify students into different classes based on their final exam grade. Moreover, the post-data mining phase conducts a detailed analysis of student' proficiency broken down into

student subgroups and grade levels, analyzes the success of particular interventions, and analyzes the importance of various factors on student course performance.

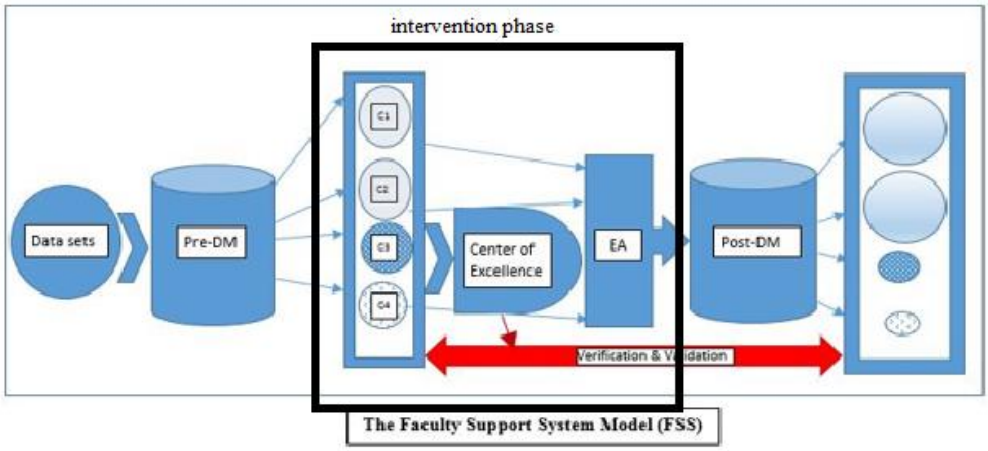

Figure 3.1: The Faculty Support System

The classification model built in the pre-data mining phase is verified by comparing the predicted outcomes of the model in pre-data mining phase with the actual student course performance. The following sub-sections discuss the Faculty Support System phases in more detail.

3.2.1 Pre-Data Mining Phase

The goal of this phase is to predict the final exam grade of the students before the final examination. In this phase, a classifier is trained using data collected from the previous teaching of the same course. The training data has five explanatory variables of internal assessments: in-class projects, quizzes, programming assignments, exams, and

attendance and one response variable, final exam grade. In this research, we used Naïve Bayes classifier for modelling this task. After Naïve Bayes Model has been obtained, as discussed in Section 4.2 - we feed the current semester data into the classifier which predicts the performance of those students (assigning them to 4 groups: First, Second, Third, Fail). The students that belong to the Fail and Third category are considered to be low-scoring ones and will be focused in the intervention phase.

3.2.2 Intervention Phase

The goal of this phase is to collect and analyze the data regarding student characteristics, and support the low-scoring students to improve their performance by providing different intervention methods. A group of professionals called the *Center of Excellence Team* will be responsible for the guidance and support of the selected low-performing students. The Center of Excellence team consists of group of decision makers, and professional teachers to support, promote, and enhance teaching effectiveness and student learning. In fact, the Center of Excellence team implements intervention strategies such as peer tutoring, supplemental instruction, academic counselling, walk-in and individual training, study through games, mentoring on note taking, time management, exam preparation, and stress management.

Peer Tutoring: Peer tutoring is a method of instruction that involves students teaching other students. Students learn more and demonstrate mastery when they can comprehensively explain a subject. Vice versa, when a student is struggling, having

someone who is on the same age level is helpful as they help to create bridges in the learning gaps. A struggling student can greatly benefit from preparing and teaching the topic they are studying, to a tutor within the same age group.

Supplemental Instruction: Supplemental Instruction is a form of tutoring that focuses on collaboration, group study, and interaction for assisting students in undertaking challenging courses. Supplemental Instruction targets courses with a minimum 30% drop, withdraw or fail rate and provides a trained peer who has successfully negotiated the course to assist future students.

Academic Advising: Academic Advising is an opportunity to exchange information designed to help students reach their educational and career goals. Advising is a shared responsibility between an adviser and the student. Ultimately, it is the responsibility of the student to make decisions about his/her life goals by creating a plan. Academic advisers can assist in this process by helping the student understand options, determine resources, and, when necessary, identify alternatives. While students are urged to keep parents informed of plans and progress, the advising relationship uniquely is between the academic adviser and the student.

Walk-in and individual training: In this kind of training the students with low-academic performance are trained individually by the Center of Excellence team. The Center of Excellence team takes special focus on these students by providing daily assignments,

helping them to solve problems, and provide necessary feedback. The Center of Excellence team conducts sessions and the students are provided with course-specific learning and study strategies, note taking and test taking skills, as well as the opportunity for a structured study time with other students.

Study through games: In this method, special games are designed that are integrated with the course material. Now-a-days most of the students show interest in playing games rather than studying. The games are designed in such a way that they need to answer course related questions, to make progress. Also, the level of the game can be changed by answering more difficult questions rather than easier ones. Also, the Center of Excellence team conducts sessions for the students on note taking, exam preparation, and stress management.

In this study, we focused on two methods, supplemental instruction and study through games. The supplemental instruction starts by planning review sessions that includes everyone but focusing on the low-performing students. In fact, the study through games method attracts most of the students where continuous improvement is seen. Moreover, the Center of Excellence team collects information about all the students by conducting a survey. The survey collects data related to social factors, structural factors, policy factors, institutional factors, personal factors, and learning factors. These factors are important to analyze the learning and social behavior of the students.

Social factors: The social factors include how well parents support the students' education and the degree to which the students are involved in organizational events conducted in the university. Also, these factors include how well the students can work in a group or share ideas with their friends. So, keeping all these in mind the social factors obtained are given in Table 3.1.

Social Factors
Does your parents support the major you choose in the university?
Did you join any organization on campus?
Is the organization related to your major?
Did you have a study group for your class?
How often do you meet with the study group?

Table 3.1: Social factors collected from the students

Structural Factors: The structural factors include details about the family's income. These factors are important, since studies reveal that students with less financial support from their parents, are often those receiving low grades. Gender and race information are also collected. The structural factors are given in Table 3.2.

Structural Factors
What is your family income?
What percent of your family's income is spent on your education?
Race
Gender

Table 3.2: Structural factors collected from the students

Institutional Factors: The institutional factors are mostly related to the faculty and the teaching assistants. The details like how well the faculty is teaching the course by providing sufficient material, real time examples, and in-depth knowledge of the course

are collected. Also, the factors like how well the teaching assistants are supporting the students and the frequency of interaction between students and teaching assistants or faculty are collected. The institutional factors are given in Table 3.3.

Institutional Factors
What do you think about the workload of the course?
How are the teaching assistants in the course?
Does the teaching assistants support you well?
How often do you meet with the advisor/faculty?
How often do you go to the tutoring center?
How many years will it take to complete your degree?

Table 3.3: Institutional factors collected from the students

Personal Factors: The personal factors are related to the students' health as the students with certain health problem may not able to focus on studies. Information like sleeping habits and diet of the students are also collected. Also, the student learning time is recorded in these factors. Finally, the details regarding the students' work are collected as the students may waste most of their time in working rather than studying which is the primary focus. The personal factors are given in Table 3.4.

Personal Factors
Do you have any health issues?
Do you work off campus/on campus?
Does the work relate to your studies?
How many hours do you work a week?
How many hours do you sleep a day?
How many meals do you have per day?

Table 3.4: Personal factors collected from the students

Learning Factors: The learning factors of the students include how many hours the students take to complete their homework or assignment and how many hours the students study for an exam. Also, the information regarding the students' interest in learning i.e. what do the students expect from the faculty and the kind of teaching are collected. The learning factors are given in Table 3.5.

Learning Factors
How many hours do you spend on assignment/homework?
How many hours do you study for the exam?
Do you have enough knowledge in your majoring field?
How many times do you skip a class?
What learning method do you prefer?

Table 3.5: Learning factors collected from the students

Policy factors: The policy factors include details about the financial assistance received by the students. These factors are considered because studies reveal that the students with financial support may have less pressure and can focus on studies without interruptions.

Finally, at the end of the semester, the Center of Excellence team creates a report to the university officials that summarizes the student data collected in the intervention phase, along with the provided interventions.

3.2.3 Post-Data Mining Phase

The goals and objectives of the post-data mining phase include:

1. Learning of classification models which subdivide students into different categories (First, Second, Third, and Fail) based on the final exam grade.

2. Evaluation of the prediction model obtained in pre-data mining phase by comparing the actual final exam grade and predicted final exam grade.
3. Creation of a report to the university officials identifying the internal assessments of the students, factors obtained in the survey in intervention phase, and types of intervention associated with low-scoring students in the final exam.
4. Evaluation and comparison of different classification algorithms using different performance metrics to predict student performance.

In this research, we investigated Decision Trees, Random Forests, Naïve Bayes and Multiple Layer Neural Network classifiers to predict student course performance. The performance of different classification approaches were evaluated using 10-fold cross-validation method and different metrics such as accuracy, precision, recall, and ROC curves. Finally, we computed useful statistics in the post-data mining phase, such as the percentage increase or decrease in the number of students predicted as low-scoring or high-scoring in the pre-data mining phase. The generated report helps university officials to identify factors that cause students to perform poorly in the course and to assist the usefulness of interventions to enhance student performance.

4. Technical Details of FSS

This chapter explains how the data mining algorithms are used and implemented in the Faculty Support System. To identify students for the intervention phase, a Naïve Bayes classifier was implemented. To place students into different groups based on the final exam grade, four classification models of the Weka data mining tool are used and compared: Naïve Bayes, J48 Decision Trees, Random Forests, and Multiple Layer Neural Networks. The parameters selection process for learning the four different classification models using Weka is also explained in detail.

The rest of this chapter is organized as follows: First the Naïve Bayes approach used to predict student performance is discussed along with its implementation. Next, different performance metrics used to evaluate classification models are explained. Next, we provide a brief summary of Weka tool. Finally, we discussed how the model parameters for the four different classification models were selected.

4.1 A Naïve Bayes Approach for Future Grade Prediction

As seen in the Section 2.1, Bayesian classification is based on Bayes Theorem. Let X denote evidence, which is described in our approach using a set of attribute-value pairs. Let C

refers to some hypothesis. In our case, we are interested to determine if an object belongs to class C based on evidence X, i.e. we want to determine P(C|X), the probability that the hypothesis C holds given the evidence X. This is known as posterior probability and can be calculated using Bayes' theorem from the equation 4.1.

$$P(C \mid X) = \frac{P(X \mid C) \times P(C)}{P(X)} \tag{4.1}$$

where, P(C|X) is the posterior probability of C conditioned on X, P(C) is the prior probability of C, P(X) is the prior probability of X, and P(X|C) is the posterior probability of X conditioned on C.

We assume that the evidence X is in the form given in equation 4.2.

$$X = X_1 \wedge X_2 \wedge \ldots\ldots \wedge X_n \tag{4.2}$$

Let D be a training set of tuples and their associated class labels. Each tuple is represented by an n attribute value-pairs, $X = \{X_1, X_2, \ldots, X_n\}$, which represent n measurements, for the attributes $A_1, A_2, \ldots, A_m$ for a student. Moreover, we assume that there are m classes to choose from: $C_1, C_2, \ldots, C_m$. Given evidence X, the task of Naïve Bayes is to predict the class having the highest posterior probability, conditioned the attribute value pairs $X_1, X_2, \ldots, X_n$ in X. That is, the Naïve Bayes classifier determines the class C_i for which equation 4.3 is satisfied for $1 \leq j \leq m$, $j \neq i$.

$$P(C_i \mid X) > P(C_j \mid X) \tag{4.3}$$

The class C_i with maximum $P(C_i|X)$ is called maximum posteriori hypothesis. Using Bayes theorem, we compute the maximum posteriori hypothesis using the equation 4.4.

$$P(C_i \mid X) = \frac{P(X \mid C_i) \times P(C_i)}{P(X)} \tag{4.4}$$

Given data sets with many attributes, it is computationally expensive to acquire all the necessary probabilities to compute $P(X|C_i)$. To simplify the computing process of $P(X|C_i)$, an assumption of class conditional independence is made (see equation 4.5) which assumes that the probabilities of different attributes have specific values are conditionally independent of one another, and we obtain:

$$\begin{aligned} P(X|C_i) &= \prod_{k=1}^{n} P(X|C_i) \\ &= P(X_1 \wedge X_2 \wedge \ldots \wedge X_n \mid C_i) \\ &= P(X_1|C_i) \times P(X_2|C_i) \times \ldots \times P(X_k|C_i) \end{aligned} \tag{4.5}$$

With the conditional independence assumption, we only need to estimate the conditional probability of each X_i, given C and the prior probabilities of X_i and C instead of knowing the class conditional probabilities for every combination of X. To classify a test record, the Naïve Bayes classifier computes the posterior probability for each class C using the formula given in equation 4.6.

$$P(C \mid X) = \frac{P(C)\prod_{i=1}^{d} P(X_i \mid C)}{P(X)} \tag{4.6}$$

Since P(X) does not depend on the class membership, it is fixed for each class. Therefore, it is sufficient to choose the class that maximizes the numerator term.

$$P(C)\prod_{i=1}^{d} P(X_i \mid C) \tag{4.7}$$

From the equations 4.6 and 4.7, we can deduce equation 4.8.

$$P(C \mid X) \propto P(C)\prod_{i=1}^{d} P(X_i \mid C) \tag{4.8}$$

where $\propto$ represents proportional.

4.1.1 Final Grade Prediction using Naïve Bayes

The Naïve Bayes classifier is learned using training data. In this research, we used 241 instances of a course data from the previous semester. In the training phase, we calculated the posterior probabilities $P(C_i|X_i)$ for every combination of values for each attribute X_i and every class C_i based on information gathered from the training data.

The prior and conditional probabilities are calculated by constructing frequency tables for each attribute. Next, conditional probabilities are computed from the frequency tables. The frequency tables and the conditional probabilities for each attribute are given in Tables 4.1 to 4.5.

Attendance (ATT)				
	First	Second	Third	Fail
Good	75	66	21	10
Average	12	17	8	5
Poor	5	4	7	11
Total	92	87	36	26
Conditional Probability P(ATT=value|Class)				
Good	75/92	66/87	21/36	10/26
Average	12/92	17/87	8/36	5/26
Poor	5/92	4/87	7/36	11/26

Table 4.1: Frequencies with conditional probabilities for attribute attendance

From the Table 4.1, the conditional probability, P(ATT = Good | class = First) is equal to the number of students with ATT = Good divided by total number of students belonging to class = First: 75/92. Likewise, conditional probabilities for all the other attributes were computed.

Quizzes (QZ)				
	First	Second	Third	Fail
Good	13	2	1	0
Average	59	55	5	2
Poor	20	30	30	24
Total	92	87	36	26
Conditional Probability P(QZ=value|Class)				
Good	13/92	2/87	1/36	0/26
Average	59/92	55/87	5/36	2/26
Poor	20/92	30/87	30/36	24/26

Table 4.2: Frequencies with conditional probabilities for attribute quizzes

Assignments (ASS)				
	First	Second	Third	Fail
Good	70	49	9	3
Average	18	21	8	1
Poor	4	17	19	22
Total	92	87	36	26
Conditional Probability P(ASS=value|Class)				
Good	70/92	49/87	9/36	3/26
Average	18/92	21/87	8/36	1/26
Poor	4/92	17/87	19/36	22/26

Table 4.3: Frequencies with conditional probabilities for attribute assignments

Class Projects (CP)				
	First	Second	Third	Fail
Good	57	42	7	3
Average	23	24	8	8
Poor	12	21	21	15
Total	92	87	36	26
Conditional Probability P(CP=value|Class)				
Good	57/92	42/87	7/36	3/26
Average	23/92	24/87	8/36	8/26
Poor	12/92	21/87	21/36	15/26

Table 4.4: Frequencies with conditional probabilities for attribute class projects

Exams (EX)				
	First	Second	Third	Fail
Good	38	6	3	1
Average	52	53	7	3
Poor	2	28	26	22
Total	92	87	36	26
Conditional Probability P(EX=value|Class)				
Good	38/92	6/87	3/36	1/26
Average	52/92	53/87	7/36	3/26
Poor	2/92	28/87	26/36	22/26

Table 4.5: Frequencies with conditional probabilities for attribute exams

Finding Prior probabilities of Response Class:

The prior probability of the classes are computed from the training set by simply counting the percentage of instances that belong to each class, as given in the Table 4.6.

Prior Probabilty of Response Variables				
	First	**Second**	**Third**	**Fail**
No of Instances	92	87	36	26
Probability	92/241	87/241	36/241	26/241

Table 4.6: Prior Probability of response variables

Testing phase:

During testing phase, the posterior probabilities have to be computed using equations 4.6 and 4.7; for example, for the specified evidence X given in Table 4.7.

ATT	**QZ**	**ASS**	**CP**	**EX**	**Grade**
Good	Average	Average	Poor	Poor	????

Table 4.7: Test Instance with no class label

Using the prior and conditional probabilities calculated in the training phase, we compute the posterior probability of each of the four classes using equation 4.6. For example, the posterior probability P(First|X) was computed as follows:

X = {ATT=Good Λ QZ=Average Λ ASS=Average Λ CP=Poor Λ EX=Poor}

Grade = {First, Second, Third, Fail}

P(X) = P(ATT=Good Λ QZ=Average Λ ASS=Average Λ CP=Poor Λ EX=Poor)

= P(ATT=Good) * P(QZ=Average) * P(ASS=Average)

* P(CP=Poor) * P(EX=Poor)

$$= \frac{75}{172} * \frac{59}{121} * \frac{18}{48} * \frac{12}{69} * \frac{2}{78}$$

= 0.000355

P(First | X) = P(First| ATT=Good ∧ QZ=Average

∧ ASS=Average ∧ CP=Poor ∧ EX=Poor)/P(X)

= {P(Grade = First) * P(ATT = Good | Grade = First)

* P(QZ = Average | Grade = First)

* P(ASS = Average | Grade = First)

* P(CP = Poor | Grade = First)

* P(EX = Poor | Grade = First)}/P(X)

$$= \frac{\left(\frac{92}{241} * \frac{75}{92} * \frac{59}{92} * \frac{18}{92} * \frac{12}{92} * \frac{2}{92}\right)}{0.0003555}$$

= 0.311

Similarly, we computed the posterior probabilities for the other three classes Second, Third, and Fail and we obtain:

P(Grade = Second | X) = 0.202

P(Grade = Third | X) = 0.344

P(Grade = Fail | X) = 0.143

Since the posterior probability for the class Third is higher than other classes, the evidence X in Table 4.7 is classified belong to class Third.

We implemented a Java program that computes the class label of a test instance using the methodology described above. Figure 4.1 gives the architecture of the program.

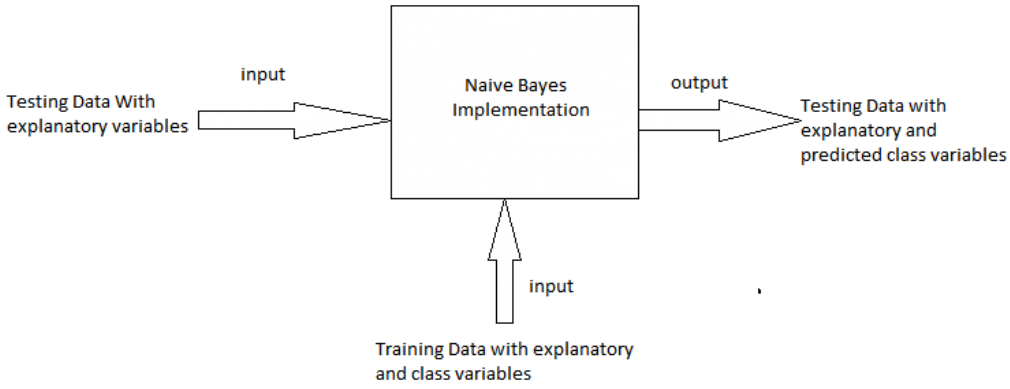

Figure 4.1: Architecture of Implemented Naive Bayes Classifier

The architecture of the Naïve Bayes implementation can be explained as follows:

1. There are 2 kinds of input to the program:
    a. A training data set with the explanatory variables and class labels
    b. A testing data set with only explanatory variables that need to be classified.
2. The classifier is learned from training data using the following steps:
    a. Construct the frequency tables that consists of counts for each value of the attribute for each class variable.
    b. Compute the likelihood of each attribute from the frequency tables and store them in a separate Hash Map. Hash Map is a key value pair data structure used to store the likelihood probabilities of each attribute to speed up retrieving their values.
    c. Compute the prior probabilities of class variable and store them in a separate Hash Map.
3. The testing set is read line by line, for each attribute-value pair its likelihoods and prior probability is retrieved from the stored hash maps. Next we compute the posterior probability of all classes. Finally, we insert the predicted class label into the output file.

4.2 Evaluation Methods and Metrics

This section explains the different evaluation methods and metrics used in Faculty Support System. Data mining metrics are used for the quantitative assessment of the performance of a data mining algorithm.

4.2.1 Evaluation Methods

The performance of classification models is evaluated with different evaluation methods. This evaluation is important to assess the quality of the model, for selecting parameters in the iterative process of learning and for selecting the most appropriate model from a given set of models. As far as classification models are concerned, the performance of a classifier is measured in terms of error-rate; different approaches to compute error rates are discussed in the following sections.

4.2.1.1 Hold out Method

The Hold out method assess training performance using a single data split. The data is split into two separate datasets where one data set is used for training and the other is used for testing. The model is learned with the training data and used to predict the output values in the testing data.

4.2.1.2 K-fold Cross Validation Method

Cross Validation is a popular technique for predicting the generalization performance of a data mining model. In this method, the dataset is divided into k subsets and the hold

out method is repeated for k times. Each time, one of the k subsets is used as the testing set and the union of other folds is used as the training set. The error rate is computed by adding the number of errors in each fold. One advantage of the k-fold cross validation method is that it is less sensitive to the arrangement of the test sets and the training sets. Another advantage of this method is that all observations are used for both training and testing but each observation is used for testing exactly once.

4.2.2 Evaluation Metrics

Metrics summarize performance of a model and give a simplified view of a model behavior. Thus, using several performance metrics helps for better understanding the model behavior, as different aspects of learning performance are captured in different metrics. Perfromance metrics are divided into qualitative understanding of errors, and visual metrics.

4.2.2.1 Qualitative understanding of errors

These metrics are based on qualitative understanding of errors, i.e. whether the prediction is correct or incorrect. They are commonly used to assess the performance of classification models. The most common qualitative performance metrics are: accuracy, sensitivity, specificity, precision, and F- Measure. They are computed based on confusion matrix (Table 4.8) and their formulas are given in Table 4.9.

		Actual Class	
		Yes	No
Predicted Class	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

Table 4.8: Confusion matrix of a 2-class classification model

Metric	Equation
Accuracy	$\frac{TP+TN}{TP+FP+FN+TN} \times 100\%$
Precision	$\frac{TP}{TP+FP} \times 100\%$
Recall (Sensitivity)	$\frac{TP}{TP+FN} \times 100\%$
F-Measure	$\frac{2 \times Precision \times Recall}{Precision+Recall}$
Specificity	$\frac{TN}{FP+TN} \times 100\%$

Table 4.9: Performance metrics

4.2.2.1 Visual metrics (ROC)

Receiver Operating Characteristics (ROC) graph is a useful technique for visualizing classification model performance. This approach relies on ranking of predictions. The ROC curve summarizes the qualitative error of the prediction model over all possible thresholds. The curve has false positive rate (specificity) on the x-axis and true positive rate (sensitivity) on the y-axis. Each point of the curve corresponds to a choice of a

different threshold. The area under the ROC curve (AUC) provides a summary of performance measure across all possible thresholds. It is equal to the probability that a randomly selected positive observation has higher predicted score than a randomly selected negative observation. As can be seen in Figure 4.2, the classifiers with high AUC values are preferred.

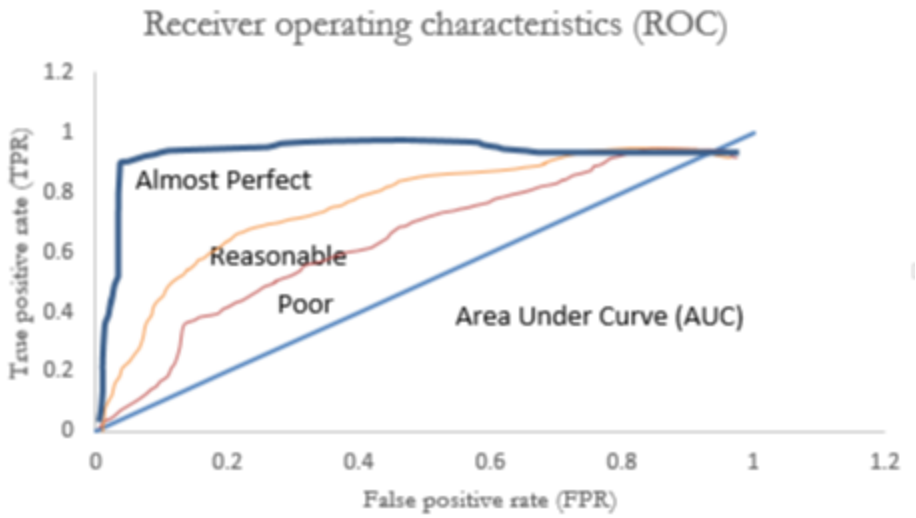

Figure 4.2: ROC curve with performance comparison

4.3 Weka Workbench

Weka provides implementations of learning algorithms and a programming environment that facilitates the application of learning algorithms to datasets. It also includes variety of tools for pre-processing datasets. We can preprocess a dataset, feed it into a learning algorithm that creates a classification model, and analyze the resulting classifier and its

performance without having to do any programming. All algorithms take their input in the form of a single relational table in the ARFF format or a CSV format.

4.4 Parameter Selection for Different Classification Models

This section explains in detail the parameter selection process while building different classification models like Multiple Layer Neural Networks, Decision Trees, and Random Forests using Weka. In contrast to other classification models, the Naïve Bayes classification model does not require any input parameters.

4.4.1 Parameter Selection of Multiple Layer Neural Networks

This section explains the parameter selection process while building a Multiple Layer Neural Network classifier using Weka. The different parameters and their values are given in Table 4.8.

The debug parameter is set to false, which indicates that classifier does not output any additional information to the console. The decay parameter is crucial in building Multiple Layer Neural Network. The true value of the decay parameter may cause the learning rate to decrease. It will divide the starting learning rate by the epoch number to determine the current learning rate. The hiddenLayers parameter determines the number of hidden layers of the neural network.

Parameter	Value
debug	False
decay	False
hiddenLayers	a
learningRate	0.3
momentum	0.2
normalizeAttributes	True
nominalToBinaryFilter	True
reset	False
seed	0
trainingTime	500
validationSetSize	0
validationThreshold	20
activationFunction	sigmoid

Table 4.8: Parameters selection in Weka Tool for Multiple Layer Neural Networks

The parameter learningRate determines how quickly the neural network updates its weights. If the learning rate is too low the network adjust its weights very slowly and if the learning rate is too large, weights might oscillate, leading to relatively poor solutions and sometimes do not even converge. The role of the momentum parameter is to speed up convergence and avoid local minima. It ranges between 0 and 0.9. Selecting a good combination of learning rate and momentum is important as this will speed up convergence and helps avoid local minima, leading to a better prediction performance.

The parameter nominalToBinaryFilter will preprocess the instances with the filter when dealing with nominal attributes. The true value of the parameter normalizeAttributes will normalize the attributes to make them equally important. If the parameter reset is set to true, the network will reset to a lower learning rate. The seed parameter is used to

initialize the random number generator, as it is important to memorize the seed to reproduce learning results. The trainingTime parameter determines the number of epochs to train. The validationSetSize parameter determines the size of the validation set. The training will continue until the error on the validation set shows no improvement, or until a training time-limit is reached.

The validationThreshold parameter is used to decide when to terminate training. Finally, the parameter activationFunction determines the type of activation function used in model. Some of the critical parameters such as learningRate and momentum were selected using a trial and error method. We tried multiple combinations of parameter values, and selected the parameter combination which minimized the error rate. For the other parameters, default values provided by Weka were used in the experiments.

4.4.2 Parameter Selection of Decision Trees

This section explains the parameter selection process while building a Decision Tree classifier using Weka. The different parameters and their values are given in Table 4.9.

Parameter	Value
confidenceFactor	0.25
minNumObj	2
numFolds	3
unpruned	False

Table 4.9: Parameters selection in Weka Tool for Decision Trees

There are usually two criteria for the quality of decision trees: classification accuracy and decision tree size, expressed as the number of nodes in the tree. Too complex decision trees usually over fit the data and leads to a non-optimal generalization error. In contrast, very small decision trees have a low training and generalization error. In summary, smaller trees are often preferred to larger ones, as they do not overfit the training set and are less sensitive to noise. The size of decision trees can be controlled during the decision tree construction process by pruning. There are two approaches to decision tree pruning:

1. Pre-pruning - terminating the subtree construction during the tree-building process.
2. Post-pruning - reducing the size of an already constructed tree.

Post-pruning tends to give better results than pre-pruning because it makes pruning decisions based on a fully grown tree, unlike pre-pruning, which can suffer from premature termination of the tree-growing process. As given in Table 4.9, the two parameters confidenceFactor and numFolds play a vital role in post-pruning. Lowering the confidence factor decreases the amount of post-pruning. We tested the J48 classifier with confidence factor ranging from 0.1 to 0.5 by an increment of 0.1 and recorded the tree size and the error rate. We selected 0.25 for the confidenceFactor parameter as this setting lead to the lowest testing error. The number of minimum instances per node (minNumObj) was set to 2, and cross-validation folds for the Testing Set (numFolds) was

set to 3 during confidence factor testing. The unpruned parameter was set to false, to initiate post-pruning.

4.4.3 Parameter Selection of Random Forests

This section explains the parameter selection process while building a Random Forest classifier using Weka. The different parameters and their values taken are given in Table 4.10.

Parameter	Value
maxDepth	0
numFeatures	2
numTrees	100
seed	1

Table 4.10: Parameters selection in Weka Tool for Random Forests

The forest error rate depends on two things:

1. The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.
2. The error rate of each tree in the forest. A tree with a low error rate is a strong classifier. Increasing the accuracy of the individual trees decreases the forest error rate.

The maxDepth parameter determines the depth of the trees that form the random forest. We choose 0 for this parameter, as we did want to impose any constraints on tree sizes in the forest. The seed attribute is used to initialize the random-number generator.

Random numbers are used for the weighted sampling with replacement in boosting algorithm - memorizing the seed is important for reproducibility of learning results.

The important parameters when building a Random Forest are numTrees and numFeatures given as the number of trees used in the forest and the number of attributes used in each tree respectively. To find the best settings for those two parameters we set the numFeatures parameter to the default value (square root of the number of all predictors) and evaluated different numTrees values: {100, 200, 300….,1000}. We build ten Random Forest classifiers for each numTrees value, recorded the error rate and picked the parameter value with the lowest error rate. Next, we varied the numFeatures value, keeping the best numTree value fixed. Based on this approach, the numTrees was set to 100 and the numFeatures was set to 2, as this setting leads to the lowest error rate.

5. Experimental Results

5.1 Objectives and Overview of Experiments

This chapter discusses the various experiments that were carried out to evaluate different functionalities of the Faculty Support System. We evaluated different classification algorithms, namely Naïve Bayes, Decision Trees, Random Forests, and Multi-Layer Neural Networks on real-world course data sets to determine the most suitable approach for the post-data mining phase. Moreover, we evaluated the Naïve Bayes classifier that was used in pre-data mining phase to identify low-scoring students based on the actual results. Finally, an exploratory data analysis was conducted on the student data, intervention data (collected in the intervention phase), along with course performance data. In short, this chapter will try to establish the usefulness of the functionalities provided by FSS.

This chapter is organized as follows: First, we provide details of the datasets that are used to conduct the experiments and describe how they are obtained. Next, results of the experiments are discussed that evaluate and compare the employed classification algorithms. Finally, exploratory data analysis results for the data collected during the intervention phase are presented.

5.2 Data Sets

The datasets used in this study were obtained from two different courses of computer science department at University of Houston, Texas (USA). The details of the datasets are given in Table 5.1.

Data Set	Course	Semester	Size
1	COSC 1410	Spring 2015	111
2	COSC 2410	Fall 2015	130

Table 5.1: Data Sets used to evaluate FSS model

The datasets contain data related to the internal assessments of the students taking the course and their final exam grade. More specifically, six attributes are associated with each dataset: performance in class projects, quizzes, programming assignments, exams, attendance, and final exam grade.

5.2.1 Data Pre-Processing

Before applying data mining algorithms, it is necessary to pre-process the collected course dataset. In particular, the data we collected for different courses were pre-processed to deal with missing values. Moreover, multiple attributes were aggregated into single attributes, and the datasets were standardized to obtain compatibility of data collected from different courses.

Initially, the datasets contained missing values; for example, some students may not have taken the second exam due to sickness or some other reason. Also, there were several exams conducted across the semester. We combined all the attributes into a single attribute by aggregating all attribute values associated with the particular course activity. For example, in a course, there are five assignments conducted over the semester; the obtained five assignment scores were aggregated into a single assignment score, by taking the weighted average of the five scores. Similarly, the exams, class projects, quizzes, and attendance were also aggregated into single value attributes. As far as the missing values are concerned, we replaced those with attribute averages.

Data standardization is important because different courses have different course structures and different evaluation metrics to measure student performance. To successfully develop a generalized model for student performance assessment, the course data needs to be standardized into a generic and unified course format. The data standardization was achieved by aggregating multiple scores into singles scores for each category and by transforming the continuous variables into discrete variables. In particular, the numerical values of scores in each attribute were transformed into categorical values in the following way:

- **QZ:** Quizzes conducted during the course. It was divided into three classes: Poor = <60%, Average = 60% to 80% and Good = 80%.
- **ASS:** Programming Assignments in the course. It was divided into three classes: Poor = <60%, Average = 60% to 80% and Good = 80%.

- **CP:** In-class Group projects in the course. It was divided into three classes: Poor = <60%, Average = 60% to 80% and Good = 80%.
- **ATT:** Attendance of Student across the semester. It was divided into three classes: Poor = <60%, Average = >60% and <80%, Good = 80%.
- **EX:** Exams conducted during the semester. Exams were divided into two classes: Poor = <60%, Average = > 60% and <80%, Good = >80%.
- **FEG:** End Semester Marks obtained in the final examination of the semester. It was split into four classes: First = >60%, Second = >45% and <60%, Third = >36% and < 45%, Fail = < 40%.

Table 5.2 gives variable names and domain values of the course datasets that were generated for the two courses and Table 5.3 gives an example of a course dataset after pre-processing.

Variable	Description	Possible Values
QZ	Quizzes	Poor, Good, Average
ASS	Assignments	Poor, Good, Average
EX	Exams	Poor, Good, Average
CP	Class Projects	Poor, Good, Average
ATT	Attendance	Poor, Good, Average
FEG	Final Exam Grade	First, Second, Third, Fail

Table 5.2: Different variables and their values used in FSS model

ATT	QZ	ASS	CP	EX	Grade
Good	Average	Good	Good	Average	First
Good	Average	Good	Good	Average	First
Good	Average	Good	Good	Good	First
Poor	Average	Average	Good	Average	First
Poor	Poor	Poor	Poor	Poor	Third
Good	Poor	Poor	Poor	Poor	Fail
Good	Average	Good	Good	Average	Second
Average	Average	Good	Average	Average	Second
Poor	Average	Good	Poor	Average	First
Good	Average	Average	Good	Good	First

Table 5.3: Records of final data set after pre-processing

5.3 Experiment 1: Prediction of Final Examination Grade Using Naïve Bayes

This experiment focuses on the prediction of the final examination grade of the students in the pre-data mining phase based on different explanatory variables like quizzes, exams, assignments, class projects, and attendance. The Naïve Bayes approach explained in Chapter 4.1 was used for this purpose. The Naïve Bayes algorithm was trained with the training data, obtained from the previous semesters of the course and predicted the final exam grade for the current semester data.

The predicted grades of students for the first dataset are given in Table 5.4. It can be seen from the Table 5.4 that the employed Naïve Bayes classifier predicted 8 students in the

Third grade and 18 students in the Fail grade. Consequently, 8 + 18 = 26 students were identified to be at risk in failing the course.

Data Set 1 with 111 instances	
Predicted Grade	**Number of Students**
First	46
Second	39
Third	**8**
Fail	**18**

Table 5.4: Predicted number of students in each category for dataset 1

Table 5.5 shows the results for the second dataset, here the Naïve Bayes classifier classified 25 students as Third grade and 20 students as Fail grade.

Data Set 2 with 130 instances	
Predicted Grade	**Number of Students**
First	47
Second	38
Third	**25**
Fail	**20**

Table 5.5: Predicted number of students in each category for dataset 2

5.4 Experiment 2: Finding the Best Performing Classifier

This experiment is focused on comparing different classifiers namely Naïve Bayes, J48 Decision Trees, Random Forests, and Multiple Layer Neural Networks. The classifiers were evaluated using 10-fold cross validation method, explained in Section 4.2.1 and the results

were compared using the evaluation metrics, introduced in Chapter 4.2.2. Classification models were learned using the parameter settings described in Chapter 4.4.

The experimental results for the four different classifiers are summarized in Table 5.6 for the dataset 1. The table gives the confusion matrix for each classifier. The performance of the classifier can be easily evaluated by inspecting the confusion matrix. The rows of the table are the actual class label of an instance, and the columns of the table are the predicted class label of an instance.

	First	Second	Third	Fail		First	Second	Third	Fail
	J48 Decision Trees					**Naïve Bayes**			
First	27	15	2	1		30	12	2	1
Second	18	15	2	4		13	21	0	5
Third	5	2	0	7		3	3	1	7
Fail	1	2	2	8		1	2	4	6
	Random Forests					**Multiple Layer Neural Network**			
First	32	10	2	1		33	8	4	0
Second	15	17	4	3		12	20	4	3
Third	6	1	0	7		3	2	2	7
Fail	1	5	3	4		1	2	4	6

Table 5.6: Confusion Matrix for different classifiers using dataset 1

Based on the confusion matrix, performance metrics such as precision, recall, accuracy, sensitivity, and specificity were computed and presented in Table 5.7 for dataset 1. As seen in Table 5.7, the time taken to build the J48 Decision Trees classifier was 0.01 seconds and it was the fastest classifier when compared to other methods. On the other hand, it took 1.12 seconds to train the Multiple Layer Neural Network classifier.

Performance Metrics	Naïve Bayes	Multiple Layer Neural Network	J48	Random Forest
Test Mode	10 fold Cross Validation			
Time taken to build Classifier	0.01 seconds	1.12 seconds	0.04 seconds	0.5 seconds
Correctly Classified Instances	58 (52.2 %)	61 (54.9 %)	50 (45.1 %)	53(47.7%)
Incorrectly Classified Instances	53 (47.7 %)	50 (45.1 %)	61 (54.9 %)	58(52.2%)
Mean Absolute Error	0.268	0.23	0.297	0.261
Root Mean Squared Error	0.394	0.39	0.417	0.390
Relative Absolute Error	78.1 %	68.36 %	86.7 %	76.2%
Root Relative Squared Error	95.3 %	94.73 %	100.8 %	94.3%
True Positive (TP) Rate	0.523	0.55	0.45	0.477
False Positive (FP) Rate	0.211	0.184	0.262	0.238
Precision	0.508	0.555	0.416	0.452
Recall	0.523	0.55	0.45	0.477
F-Measure	0.512	0.549	0.429	0.461
ROC Area (AUC)	0.747	0.779	0.644	0.755

Table 5.7 Performance Metrics for classifiers using 10-fold cross validation on dataset 1

Multiple measures were used to assess accuracy: correctly classified instances; the error rate was given by the mean absolute error and root mean squared error, and for the ROC measure, the area under curve (AUC) was reported. It can be seen that the accuracy of Multiple Layer Neural Network is 54.9% while J48 Decision Trees, Naïve Bayes, and Random Forests have 45%, 52%, and 47.7%, respectively. The Multiple Layer Neural Network achieved an accuracy that is 3% higher than those of other classifiers. As far as the AUC score is concerned, the Multiple Layer Neural Network has 77% of the area under the curve, whereas the AUC of other classifiers is at least 3% lower.

The confusion matrix for the four different classification approaches for dataset 2 is given in Table 5.8.

	First	Second	Third	Fail		First	Second	Third	Fail
	J48 Decision Trees					**Naïve Bayes**			
First	45	2	0	0		38	9	0	0
Second	13	33	2	0		11	35	2	0
Third	0	9	7	6		0	7	11	4
Fail	0	0	2	11		0	0	8	5
	Random Forests					**Multiple Layer Neural Network**			
First	10	2	2	0		39	8	0	0
Second	6	4	3	2		13	35	4	0
Third	1	1	1	3		0	6	11	5
Fail	0	0	3	0		0	0	2	11

Table 5.8: Confusion Matrix for different classifiers using dataset 2

Table 5.9 reports the results for various evaluation measures for dataset 2. As seen in Table 5.9, the time taken to build the J48 Decision Trees classifier was 0.01 seconds and is the fastest classifier when compared to other classifiers. On the other hand, it took 0.91 seconds to train the Multiple Layer Neural Network. It can be seen that the accuracy of the Multiple Layer Neural Network is 73.3% while J48 Decision Trees, Naïve Bayes, and Random Forests have 70.7%, 70%, and 68.4% accuracy respectively. The Multiple Layer Neural Network achieved an accuracy that is 3% higher than other classifiers. Moreover, the Multiple Layer Neural Network has 89% of the area under the curve, whereas the AUC of other classifiers is at least 3% lower.

Performance Metrics	Naïve Bayes	Multiple Layer Neural Network	48	Random Forest
Test Mode	10-fold cross validation			
Time taken to build Classifier	0.5 seconds	0.91 seconds	0.01 seconds	0.05 seconds
Correctly Classified Instances	89 (68.4 %)	96 (73.3 %)	92 (70.7 %)	91 (70 %)
Incorrectly Classified Instances	41 (31.5 %)	34 (26.6 %)	38 (29.2 %)	39 (30 %)
Mean Absolute Error	0.206	0.161	0.169	0.198
Root Mean Squared Error	0.317	0.305	0.335	0.3227
Relative Absolute Error	59.1 %	52.0%	48.5%	50.8 %
Root Relative Squared Error	76.1 %	79.5%	80.5 %	74.7 %
True Positive (TP) Rate	0.685	0.738	0.708	0.70
False Positive (FP) Rate	0.139	0.118	0.131	0.135
Precision	0.678	0.73	0.703	0.688
Recall	0.685	0.735	0.708	0.70
F-Measure	0.679	0.72	0.704	0.69
ROC Area (AUC)	0.843	0.893	0.864	0.863

Table 5.9 Performance Metrics for classifiers using 10-fold cross validation on dataset 2

In summary, from the experiments with the two different course data sets, we observe that Multiple Layer Neural Networks achieved significantly higher accuracy than Naïve Bayes, J48 Decision Trees, and Random Forests.

5.5 Exploratory Data Analysis of Intervention Phase Data

This section describes the results of exploratory data analysis (EDA) conducted on the data collected during the intervention phase. The primary interest in the intervention phase is to support the low-scoring students that have been identified in the pre-data mining phase. The Center of Excellence team performs EDA with the data related to

student predicted grades, final grades, and the data collected during the intervention phase. The goal of the exploratory data analysis is to:

1. maximize insights into the data collected
2. preliminary selection of appropriate types of intervention
3. determining relationships among the explanatory variables collected

5.5.1 Determining the Proportion of Students in each Predicted Category

Figure 5.1 shows the histogram of the proportion of students in each predicted category (First, Second, Third, and Fail). There are around 28% students in the Third and Fail class. These students fall into a low scoring group and are primary interest for the intervention phase.

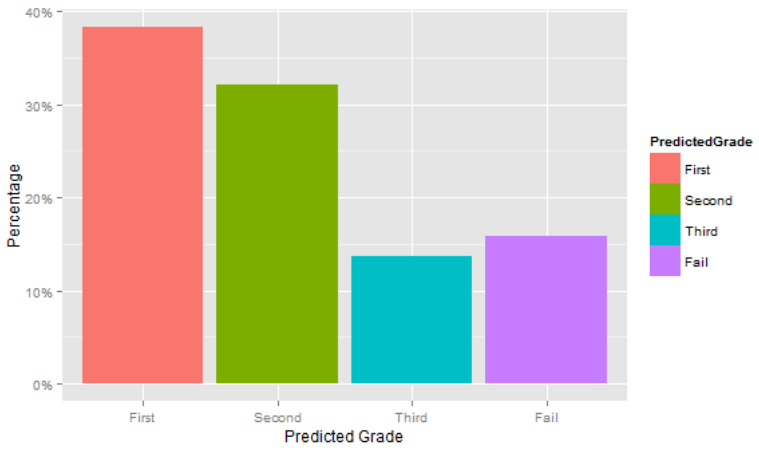

Figure 5.1: Histogram of proportion of predicted students in each category

5.5.2 Determining the Proportion of Students in each Category Based on Gender

Figure 5.2 gives the detailed decomposition of students into each category based on gender. If we look at the proportions of the grade distribution, it can be seen that the male students perform much worse than the female students; particularly, there are significantly more number of male students in the Third category and also a higher percentage of male students in the Fail category.

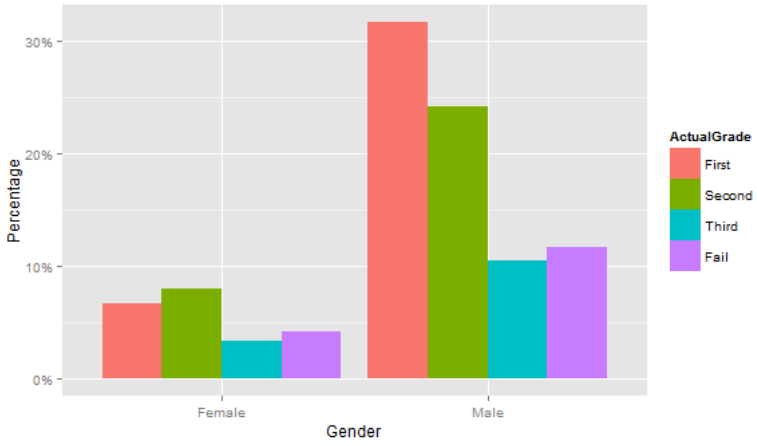

Figure 5.2: Histogram of proportion of students in each category based on gender

5.5.3 Determining the Proportion of Students in each Category Based on Ethnicity

Figure 5.3 shows the decomposition of students based on their ethnicity or race. If we look at the proportions of the ethnicity distribution, it can be seen that Asians perform

well in the course followed by Americans, Hispanic, Others, and Africans.

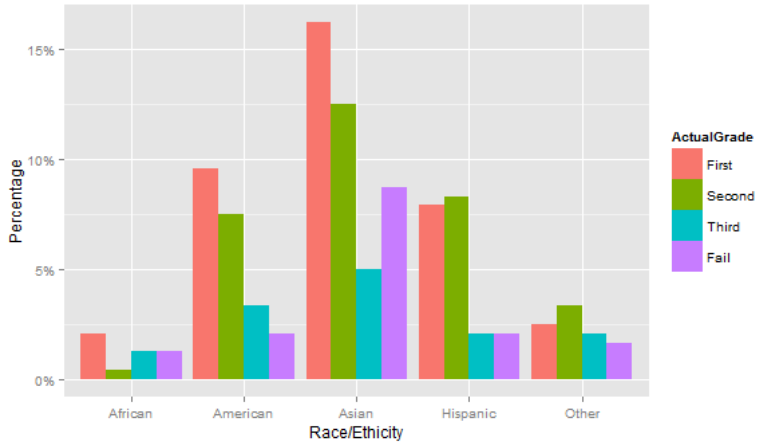

Figure 5.3: Histogram of proportion of students in each category based on ethnicity

5.5.4 Determining the Proportion of Students in each Category Based on Faculty Support

Figure 5.4 assess the faculty support in helping students. Although some of the students claim that faculty are very helpful, there is also an equal proportion of students who claim that the faculty are somewhat helpful. In summary, the results are inconclusive since the grade proportions do not vary much between groups.

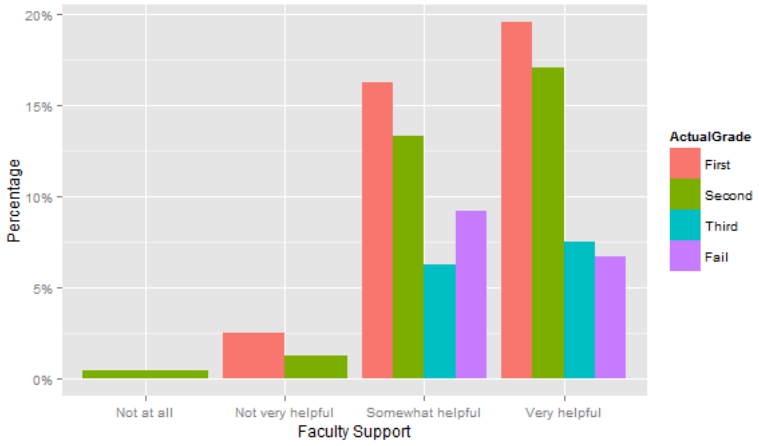

Figure 5.4: Histogram proportion of students in each category based on faculty support

5.5.5 Determining the Proportion of Students in each Category Based on Learning Method

Figure 5.5 shows the proportion of students who are interested in different kinds of learning methods. Most of the students claim that they are interested in kinesthetic learning i.e. learning by practice. Also, a small proportion of students claim that they are interested in visual learning i.e. learning by observations while only a few students claim that they are interested in auditory learning i.e. learning by listening. This study gives the Center of Excellence team an insight of what types of intervention should they provide in the intervention phase.

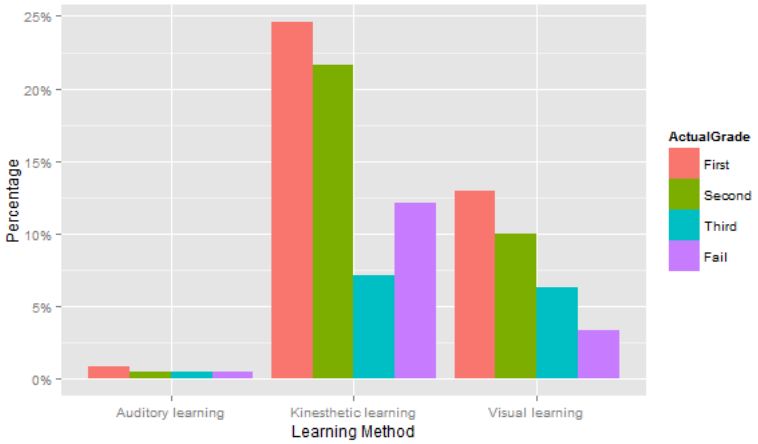

Figure 5.5: Histogram of proportion of students in each category based on learning method

Based on the identified factors, the Center of Excellence team takes necessary actions and adopt different kinds of intervention as explained in Chapter 3.2.2.

5.6 Experiment 3: Evaluating the pre-Data Mining Classification Model

In this experiment, the classification model used in pre-data mining phase was evaluated based on the actual performance of the students. The predicted grade in the pre-data mining phase was compared to the actual final exam grade and different statistics were evaluated and explained in the following sections. The detailed decomposition of the students based on predicted and actual grades are given in Tables 5.11 and 5.12 for each dataset.

		Actual				
		First	**Second**	**Third**	**Fail**	**Total**
predicted	**First**	31	12	3	0	46
	Second	12	22	2	3	39
	Third	1	2	4	1	8
	Fail	1	3	5	9	18
	Total	45	39	14	13	111

Table 5.11: Decomposition of students based on predicted and actual grades on dataset 1

Interpreting the results summarized in Table 5.11, we observe that out of 18 students who were predicted to fail, 1 student moved to the first group, 3 students moved to the second group, and 5 students moved to the third group. Hence, out of 18 students in the predicted Fail group, 9 students have improved their performance. Similarly for students that were predicted Third grade, out of 8 students, 3 students have improved their performance, and 4 students remained in the same grade and 1 student failed. Moreover, there are 3 students who were predicted Second grade but actually failed in the final examination.

		Actual				
		First	**Second**	**Third**	**Fail**	**Total**
predicted	**First**	35	12	0	0	47
	Second	12	23	3	0	38
	Third	0	13	11	1	25
	Fail	0	0	8	12	20
	Total	47	48	22	13	130

Table 5.12: Decomposition of students based on predicted and actual grades for dataset 2

Interpreting the results summarized in Table 5.12, we observe that out of 20 students who were predicted to fail, 8 students moved to the third group. In summary, 40% of the students who were predicted to fail did not fail the course. Hence, out of 20 students in the predicted Fail group, 8 students have improved their performance. Similarly for predicted Third grade, out of 25 students, 13 students have improved their performance, and 11 students remained in the same category, and 1 student performed worse.

The results of our analysis suggest that the type of intervention provided in the intervention phase helped students and a significant percentage of them actually improved their performance. Finally, a report was created to the university officials that includes final exam grade, predicted grade, and the data collected in the intervention phase and a summary concerning the type of intervention used for low-scoring students.

6. Conclusion and Future Work

The overall goal of this research is to reduce the course drop-out rate of students and enhance their performance in a course. To achieve this goal, we designed a framework called *Faculty Support System (FSS)*. FSS uses classification models that are learned from past teachings of courses to identify the low-scoring students. Next, the identified student at risk receives additional support by Center of Excellence team by providing different types of interventions. This research has demonstrated that, by using classification models that are learned from past data, we can efficiently generate a reliable Faculty Support System (FSS) classification model, i.e. FSS provides data mining techniques for the evidence-based identification of struggling students. Moreover, FSS provides a framework for an academic environment that trains, engages, and motivates students to accomplish better course retention rates. Therefore, FSS is an important tool which can be used to help low-performing students to succeed, also contributing to reduce course retention rate. Therefore, when using FSS, no student will be left behind.

In this research, four classification models Naïve Bayes, Decision Trees, Random Forests, and Multiple Layer Neural Networks were used, compared, and evaluated for student course performance prediction. A benchmark of real-world datasets were collected from two different computer science courses at the University of Houston to assess and

compare the classification models. The performance of these data mining algorithms was evaluated based on different performance measures like accuracy, precision, recall, sensitivity, specificity, and ROC curves. The results show that Multiple Layer Neural Networks performed better when compared to other algorithms.

For future work, we aim to carry out more experiments using more course data and also more courses to further evaluate and enhance the FSS classification model. Also, we are planning to collect new types of data. A more comprehensive study of more than five hundred records of students' data from different courses in computer science department is being prepared to attempt broadening the scope of this research by studying different attributes and targeting new learning-related goals such as identifying the factors that affect students' failure. Also, approaches will be investigated to predict the low scoring students as early as possible in the semester because the earlier students at risk are identified, the better intervention can be provided. Additionally, the achieved accuracy of the implemented classification model in FSS can be improved. We are trying to explore different approaches like boosting, bagging to increase the accuracy of the FSS classification model. Moreover, identifying the factors that determine student success and developing models by which individual students would benefit from a particular intervention are important themes of the future work. Finally, we are planning to automate the total process of data collection, data standardization, grade prediction, and performance evaluation in FSS.

References

[1] Martinez, Gomez. Contributions from Data Mining to Study Academic Performance of Students of a Tertiary Institute. ACM Trans. Journal of Educational Research 2.9 (2014): 713-726.

[2] Kabakchieva, Dorina. Predicting student performance by using Data Mining methods for classification. Cybernetics and information technologies 13.1 (2013): 61-72.

[3] Rokach, Lior, and Oded Maimon. Data Mining and knowledge discovery handbook. Vol. 2. New York: Springer, 9.1 (2005): 149-157.

[4] Breiman Leo. Random forests. Machine learning 45.1 (2001): 5-32.

[5] Chan, Jonathan Cheung-Wai, and Desiré Paelinckx. Evaluation of Random Forest and Adaboost tree-based ensemble classification and spectral band selection for ecotype mapping using airborne hyperspectral imagery. Remote Sensing of Environment 112.6 (2008): 2999-3011.

[6] Jain, Anil, Jianchang Mao, and Mohiuddin M. Artificial neural networks: A tutorial Computer 3 (1996): 31-44.

[7] Han and Kamber. Data Mining: concepts and techniques. Elsevier 7.1(2011): 283-291.

[8] Shanmuga Priya, Senthil Kumar. Improving the Student's Performance Using Educational Data Mining. International Journal of Advanced Networking and Applications 4.4 (2013): 1806.

[9] Ben-Zadok, Galit. Examining online learning processes based on log files analysis: A Case Study. 5th International Conference on Multimedia and ICT in Education (2007).

[10] Goyal, Vohra R. Applications of Data Mining in Higher Education, International Journal of Computer Science Issues 9.2 (2012): 187-192.

[11] Surjeet Kumar, Brijesh Bharadwaj, and Saurabh Pal. Data mining applications: A comparative study for predicting student's performance 1202.4815 (2012).

[12] Ashraful Alam Pathan, Mehedi Hasan, Ferdous Ahmed, and Dewan Farid. A Mining Model for Developing Students' Programming Skills. 8th International Conference on IEEE (2014): 1-5.

[13] Minaei, Bidgoli B. Predicting student performance: an application of Data Mining methods with an educational web-based system. In 33rd ASEE/IEEE Frontiers in Education Conference (2003): 1-6.

[14] Pittman K. Comparison of Data Mining techniques used to predict student retention. Ph.D. Thesis, Nova Southeastern University (2008).

[15] Romero C. Data Mining algorithms to classify students. In 1st International Educational Data Mining Conference (2008): 8-17.

[16] Nguyen, Paul, and Peter H. A Comparative Analysis of Techniques for Predicting Academic Performance. In Proceedings of the 37th ASEE/IEEE Frontiers in Education Conference (2007): 7-12.

[17] Al-Radaideh, Al-Shawakfa, and AI-Najjar M. Mining Student Data using Decision Trees. In Proceedings of the International Arab Conference on Information Technology (ACIT'2006), Yarmouk University, Jordan (2006).

[18] Muslihah, Yuhanim, Norshahriah, Mohd Rizal, Fatimah, and Hoo Y. S. Predicting NDUM Student's Academic Performance Using Data Mining Techniques. In Proceedings of the Second International Conference on Computer and Electrical Engineering, IEEE Computer Society (2009).

[19] Ramaswami, Bhaskaran. CHAID Based Performance Prediction Model in Educational Data Mining. IJCSI International Journal of Computer Science Issues 7.1 (2010): 212-217.

[20] Mishra, Kumar, and Gupta S. Mining Students Data for Prediction Performance. In Advanced Computing & Communication Technologies (ACCT), Fourth International Conference on IEEE (2014): 255-262.