A MACHINE LEARNING APPROACH TO PREDICT FIRST-YEAR STUDENT RETENTION RATES AT UNIVERSITY OF NEVADA, LAS VEGAS

by

Aditya Rajuladevi

Bachelor of Technology, Computer Science
Jawaharlal Nehru Technological University, Hyderabad, India
2014

A thesis submitted in partial fulfillment of
the requirements for the

Master of Science in Computer Science

Department of Computer Science
Howard R. Hughes College of Engineering
The Graduate College

University of Nevada, Las Vegas
May 2018

UNLV
UNIVERSITY OF NEVADA LAS VEGAS

The Graduate College

We recommend the thesis prepared under our supervision by

Aditya Rajuladevi

entitled

A Machine Learning Approach to Predict First-Year Student Retention Rates at University of Nevada, Las Vegas

be accepted in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

Department of Computer Science

Fatma Nasoz, Ph.D., Committee Chair

Justin Zhan, Ph.D., Committee Member

Yoohwan Kim, Ph.D., Committee Member

Magdalena Martinez, Ph.D., Graduate College Representative

Kathryn Hausbeck Korgan, Ph.D., Interim Graduate College Dean

May 2018

Abstract

First-year student retention rates for a four-year institution refers to the percentage of First-time Full-time students from the previous fall who return to the same institution for the following fall. First-year retention rates act as an important indicator of the student satisfaction as well as the performance of the university. Moreover, universities with low retention rates may face a decline in the admissions of talented students with a notable loss of tuition fees and contributions from alumni. Therefore, it is important for universities to formulate strategies to identify students who are at risk of not being retained and take necessary measures to retain them. Many universities have tried to develop successful intervention programs to help students increase their performance. However, identifying and prioritizing students who need early interventions still remains to be a very challenging task.

The retention rate at the University of Nevada, Las Vegas (UNLV) from Fall 2016 to Fall 2017 is 74.4% which indicates the need for specific intervention programs to retain the students who are at risk of dropping out after their first year. In this thesis, we propose the use of predictive modeling methods to identify such at-risk students at an early stage, so that the interventions can be offered in a timely manner. For this, we implemented and compared various classification algorithms of machine learning including Logistic Regression, Decision Trees, Random Forest Classifier, and Support Vector Machines in identifying at-risk students using classic machine learning metrics. The models were trained and tested using a set of features extracted from the datasets housed in UNLV's data warehouse that capture student information such as pre-college academics, family background, financial situation and academic performance during their first year at UNLV. The experimental results showed that Logistic Regression and Random Forest classifiers performed better in predicting at risk students at UNLV. Furthermore, students were ranked based on their risk of dropping out, which would enable the educators to focus on concentrating their intervention resources effectively.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Dr. Fatma Nasoz, for her motivation, guidance, and support throughout the research. She continuously steered me in the right direction in this research as well as my Master's program.

I would like to extend my thanks to Dr. Justin Zhan, Dr. Yoohwan Kim and Dr. Magdalena Martinez for their support and for being a part of my thesis committee. I am really grateful for all the support from Dr. Ajoy K Datta who was always available to me whenever I needed his guidance.

I am gratefully indebted to Kivanc Oner, Carrie Trentham and Becky Lorig from the Enterprise Application Services department at UNLV for their continuous support and valuable comments on this thesis. They have been of great help in answering my questions about student enrollment and retention problem at UNLV and helped me find the student data required for my analysis housed in the UNLV data warehouse.

My deep sense of gratitude to my parents Venkat Rao Rajuladevi, Mallika Rajuladevi and my sister Arthi Rajuladevi who are my moral strength and motivation. I would like to thank Sai Phani Krishna Parsa, Paritosh Parmar, and Ashish Tamarakar for their constant support and guidance throughout my Master's program.

Finally, I would like to thank all my friends, seniors and juniors who made my time here at UNLV very memorable.

Aditya Rajuladevi

University of Nevada, Las Vegas

May 2018

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Integrated Postsecondary Education Data System (IPEDS) is a national system of surveys conducted annually by the National Center for Education Statistics (NCES). Among other student information, it collects retention and graduation rates from the participating institutions and it requires the institutions to follow certain definitions and standards during data reporting. IPEDS defines first-year student retention rate (often referred to as "the retention rate") as the percentage of first-time full-time students from the previous fall semester who enroll in the same institution in the current fall. A first-time student is one who is attending college for the first time and is considered full-time if he enrolls for more than 12 credits in a semester. The first-time full-time students of a given academic year are considered as a cohort, where specific cohort year represents census collected on students who were retained from the previous fall. The retention rates at postsecondary institutions in the U.S. as surveyed by IPEDS for the cohort year 2016 was 75.1% [IPE], which indicates that many universities are performing poorly in terms of retaining first-year students. Low retention rates are also a bad indicator of the university's performance and can damage the reputation of the institution in the eyes of students and parents.

The reasons behind student dropout after first year in universities can range from high expectations of the college programs, transition into an interdisciplinary curriculum, economic problems, inability to mix well with other students, or struggling due to unfulfilled prerequisite requirements [Lau03]. Many researchers have formulated solutions such as building learning communities, providing additional resources [Tin99], highlighting student participation in campus life and providing academic support [Lau03]. Also, few studies have indicated that the risk of dropping out decreases with an increase in academic performance [AMBS99]. Thus, one way to increase retention is to increase academic success. In recent years, many universities have invested significantly in devel-

opment and implementation of intervention programs to help at-risk students and support them individually to improve their academic performance.

The success of such intervention programs depends on the university's ability to accurately identify students who need help. In a traditional approach, many universities have used academic performance indicators such as GPAs, truancy rates, high school grades, SAT and/or ACT scores of students to generate rules that can be used to identify at-risk students [BS16]. Although such rule-based systems served as good indicators of identifying at-risk students for some years, they had some downsides such as having fewer accuracies, being static, being expensive to generate and maintain and most importantly lacking a validation mechanism to verify the predictions. Alternatively, recent research has indicated the potential value of machine learning algorithms such as Logistic Regression, Random Forest Classifiers, Decision Trees, Support Vector Machines (SVM) and Neural Networks for the problem [Pla13, LAS+15, MDDM16]. These algorithms, when trained using traditional academic data can identify at-risk students more accurately. The performance of these algorithms can be evaluated using various metrics such as Precision, Recall, and Area Under Curve (AUC) thus giving us a good indicator to validate the results. However, the application of such predictive methods to identify at-risk students is still at its early stages owing to the limited availability of data and the implementation complexity. Currently, many universities have defined rules in the collection of data to use for such a research.

Over the recent years, the retention rates at UNLV have displayed a highly varying pattern. It dropped from 77% in 2011 cohort year to 74% in 2013 cohort year and then increased to 77% in 2015 cohort year, which later on fell to 74% in 2015 cohort year as shown in Figure 1.1. Such an unstable pattern of freshmen retention rates has drawn a lot of attention by the educators and administration at UNLV. Therefore, a predictive approach to identify at-risk students and supporting them with additional resources would be quite beneficial to increase the retention rates at UNLV.

1.1 Objective

The objective of this thesis is to create predictive models that can be used to identify students who are at the risk of not being retained. In this thesis, machine learning algorithms including Logistic Regression, Decision trees, Random Tree Classifiers, and SVMs will be trained on UNLV student data. Trained models will then be used to predict at-risk students on a test dataset which the model has not seen earlier. The models are evaluated and compared using metrics such as

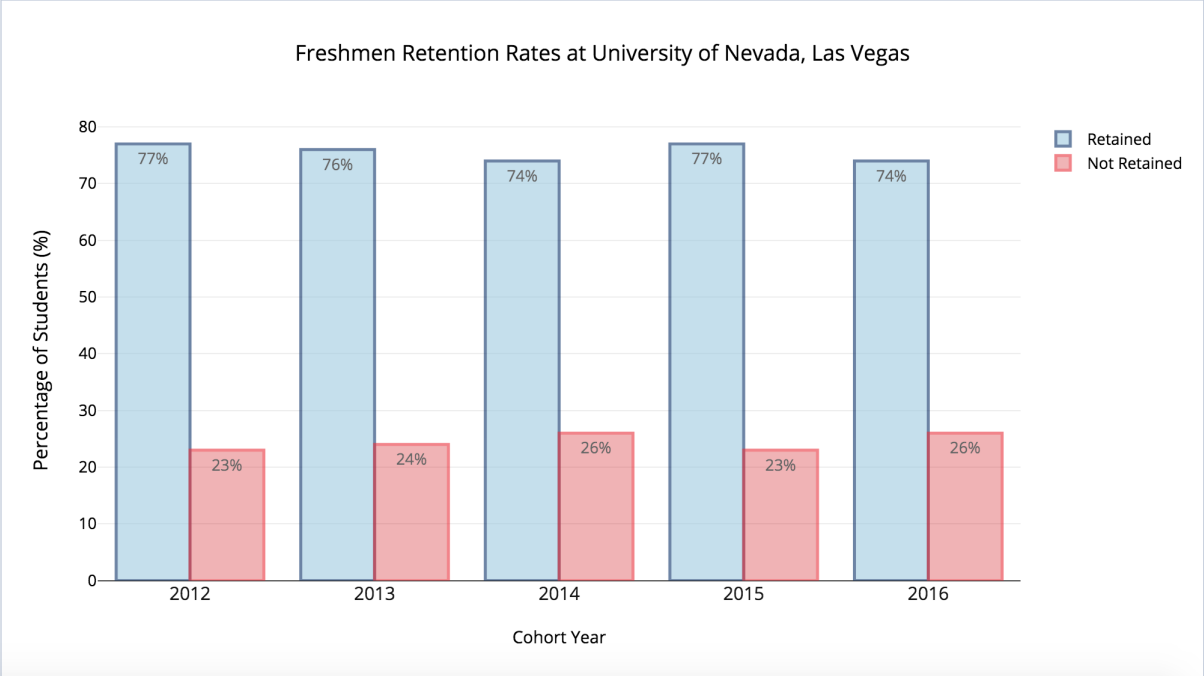


Figure 1.1: Freshmen Retention Rates at UNLV

precision, recall, and area under the curve to determine which model provides the best results. Another contribution of this thesis is to rank students based on predictions from models, which will be very helpful in efficient allocation of resources as part of the intervention programs.

1.2 Outline

In Chapter 1, the brief topic of First-Year student retention rates, its importance to universities and the proposed approach to increase the retention rates at UNLV were described.

In Chapter 2, we will discuss the existing research on improving first-year retention rates and provide the background information required to understand the proposed predictive approach using machine learning. It will cover the most important and popular algorithms of machine learning.

In Chapter 3, we will describe the methodology adapted for the analysis. The characteristics of the datasets will also be described.

In Chapter 4, we will perform experiments by building the models, report the results and compare their performances to identify the ones that are best suitable for the UNLV data.

In Chapter 5, we will summarize the findings of the analysis along with some insights about future work.

Chapter 2

Background and Preliminaries

2.1 Related Work

The prediction of first-year student retention rates and identification of students who are at risk of not being retained after their first year has been a well-researched problem in the area of higher education for decades. Early studies involved learning the important factors that lead to student dropout by developing a theoretical model. Tinto is one of the major and earliest researchers in this area. Tinto's student engagement model [Tin99] has served as the basis for a large number of theoretical studies [Bra02]. Similar research was carried out by Astin et al., which focused more on the external factors such as the institution's administration and its policies when determining the reasons for student retention [Ast12]. Tinto in his 2006 study [Tin06] has stated that there has been a huge increase in the number of businesses and organizations that analyze and help institutions with the student retention problem. Later in the same study, he revealed that there was only a little change in retention rates even with these businesses helping the universities. He also described the importance of external factors such as student-faculty relationships, extracurricular programs, and orientation programs for first-year students. Moreover, he incorporated the role of academic factors into his model to make it more suitable to the college structure [Tin06]. Astin et al. in their Input-Environment model [Ast12], suggest that researchers should consider pre-college factors such as gender, race and ethnicity, family background, and high school GPA to be important for student retention.

In addition to understanding the factors responsible for student dropout, researchers were interested in identifying students who are at the risk of not being retained in order to intervene and prevent them from dropping out. Early research included usage of statistical and analytical meth-

ods such as logistic regression and discriminant analysis for predicting if a student will be retained [LAS+15, MDDM16, AC17]. The results from these models showed that these learning algorithms were, in fact, better than many existing rule-based models in terms of learning patterns from the existing student data. Educational data mining has emerged into an important field of research in studying student retention, because of its high accuracy and robustness in working with missing data [AH14]. In another study, Lauria et al. used fall 2010 undergraduate student data from four different sources and applied classifiers such as logistic regression, support vector machines and C4.5 decision trees for prediction and comparison purposes[LBD+12]. The results showed that logistic regression and svm classifiers provided higher classification accuracies compared to the decision trees to predict students at risk.

Herzog, a researcher from University of Nevada, Reno(UNR) has done some extensive research on student retention and graduation prediction where he used decision trees and neural networks to predict student retention on data from UNR[Her06]. Marbouti et al. have compared seven different prediction models for identifying at-risk students using in-semester performance factors (e.g. grades) which were based on standards-based grading [MDDM16] . Another similar research focused on the problem of imbalanced output class distribution in the field of student retention, in which the researchers tested three balancing techniques such as over-sampling, under-sampling, and synthetic minority over-sampling (SMOTE) along with machine learning algorithms [TDMK14].

Although predictive analysis using machine learning models have proved to be very effective in identifying at-risk students, they were still not efficient and useful to educators who wanted to develop academic support programs and resources to support the identified students. The primary reason for this is the limited understanding of the metrics from the predictive models by the educators. A few researchers analyzed this issue at the high school level and came up with a framework to convert model accuracies to risk scores that could be used by the educators in efficient allocation of resources [LAS+15]. Although the above-mentioned framework was providing good results, it was restricted to school-level data and thus could not yield accurate results at the university-level as they are very different environments with different set of factors. Hence in this thesis, we perform such a predictive analysis on the retention problem at college-level to identify at-risk students, so interventions can be offered in a timely manner.

2.2 Preliminaries

2.2.1 Machine Learning Concepts

Mitchell defined Machine Learning as [Mit97]:

> "A computer is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Example: playing chess.
E = the experience of playing many games of chess with different people
T = the task of playing chess
P = the probability that the program will win the next game

In general, machine learning tries to learn patterns inherent in the underlying data and remembers it as an experience and uses it for predictions. It was conceptualized from the notion of learning process adopted by a human brain. Just as the human brain gets better at a specific task by repeated learning and previous experiences, a computer will also learn more patterns hidden in the data based on its previous experiences. Additionally, the huge processing power of computers enables them to learn patterns from highly complex data which can be very difficult for a human to understand.

2.2.2 Predictive Analytics

Predictive analytics primarily deals with extracting patterns from datasets using machine learning models. The extracted patterns are used to predict future events and behaviors of previously unseen data. Predictive analytics is being used in a wide range of fields such as education, finance, automobile, and healthcare. The analytics is performed by running different machine learning algorithms on previously collected data. The algorithms try to learn patterns between different features of the data and preserve the knowledge in model parameters. The resulting model is able to predict the unknown property of a future unseen data.

As an illustrative example, consider the student data captured during admissions at a university as shown in table 2.1. The aim is to predict if the student will be accepted or not by looking at

Table 2.1: Admissions data at a University

Age	Gender	HSGPA	ACT	Accepted
20	Male	3.89	25.6	1
21	Female	3.20	22.6	1
20	Female	2.50	18.3	0
22	Male	3.10	19.2	0
19	Female	3.60	23.5	1

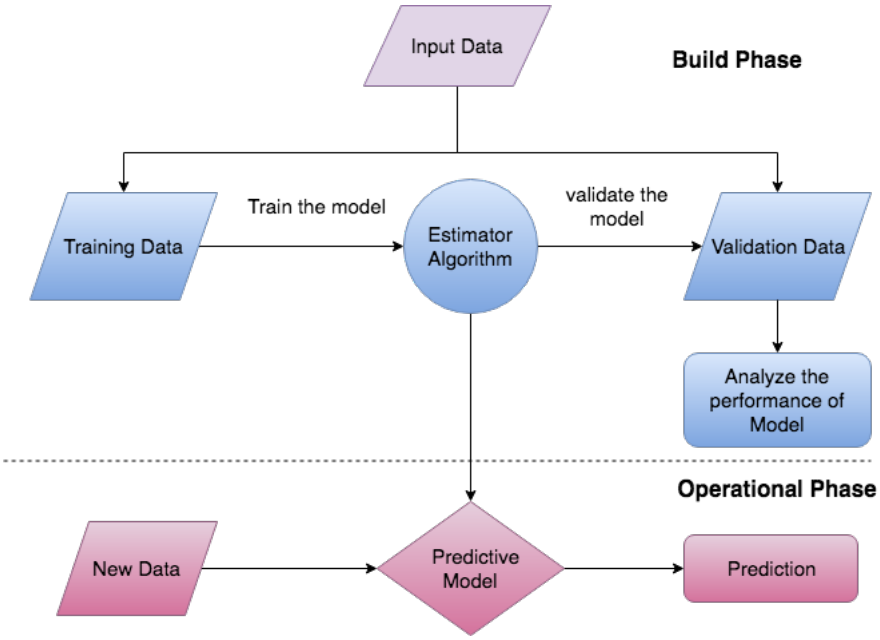


Figure 2.1: Machine Learning Approach

the other variables in the data. The column "Accepted" is said to be a dependent variable, and all the other variables are called independent variables. A "1" in the Accepted column represents the outcome of the student being accepted into the university and a "0" means he was rejected. To this data, we apply a machine learning algorithm that will learn a predictive model based on the relationships between the independent (input) variables and the dependent (output) variable. The learned model is then able to predict the output variable of an before unseen data sample by taking its other variables as input. The output variable can be of any data type. If it is a continuous numerical value, then the machine learning problem is known as regression and if it is categorical value, then the machine learning problem is known as classification.

The general approach for applying predictive analysis on datasets can be divided into 2 phases as shown in figure 2.1. Build phase deals with the process of creating a prediction model and

validating its performance. The process of creating the prediction model is known as training and the data used is known as the training data. To measure the effectiveness of the created model, it is tested on another set of previously unseen data known as validation data. We use two different datasets to check the generalization capacity of the model in making predictions on previously unseen data. If we use all the available data for training, the model may learn too much from the inputs giving rise to the problem of overfitting. Overfitting occurs when a model is performing well on its training data, but not on the previously unseen data. A common approach to handle the problem of overfitting is to divide the input data into training and validation sets. The models performance is then evaluated using metrics calculated on validation data. The metrics used to evaluate a model are explained in detail in the coming chapters. The model created in the build phase is used in the operational phase to perform predictions given a new data example.

2.2.3 Selected Models

Based on how machine learning algorithms "learn" patterns from data, they can be classified into two groups: supervised and unsupervised learning algorithms. In supervised learning, the machine learning model is trained using data whose outcomes are already known, whereas unsupervised learning is the training of machine learning models using information that is neither classified nor labeled.

In our case, since the data about students have the output labels that indicates if the student was retained in the next fall semester or not, we will apply supervised learning. Additionally, since the outcome variable in our analysis is a categorial variable, our problem is of classification. In this section, we discuss various classification models that were used in this thesis to analyze the student data of UNLV. The models selected for comparison were Logistic Regression, Decision Trees, Random Forest Classifiers, and SVMs.

2.2.3.1 Logistic Regression Model

Logistic Regression is a supervised machine learning classification algorithm that is used to predict the output of a categorical dependent variable, based on the probabilities of achieving the output categories. In logistic regression, the dependent variable is a categorical variable. It is similar to linear regression in which a model is constructed in supervision of the available data to calculate the unknown outcome variable. In linear regression, input values (x) are combined linearly using weights or coefficient values to predict a real-valued output (y), whereas logistic regression calculates

the probability of outcome dependent variable (y) belonging to each class. The general equation for this learning approach with one independent variable and a dependent output variable 'y' is as shown in equation 2.1

$$P(y = 1) = g(\beta 0 + \beta 1 * x) \tag{2.1}$$

where the coefficient '$\beta 0$' is the bias and '$\beta 1$'is the coefficient for the single input value (x) which are learned from the available input data during training and 'g' is a mathematical function known as Sigmoid function that maps the linear combination of inputs into the range of [0,1], thus giving us probabilities and is defined as shown in equation 2.2. For the case of multiple independent input variables, the input values (x) are represented as a vector x along with equal number of coefficients to be learned.

$$g(z) = \frac{1}{1 + e^{-}z} \tag{2.2}$$

As an illustrative example, consider the problem of identifying if a student will pass an exam based on the number of hours he studies before the exam. In this case, we can define the input independent variable as the number_of_hours_studied which takes a numerical value and the output variable as Student_Passed which takes values 0 or 1 . Then our logistic regression problem represents the probability of the student passing an exam given the number of hours he studied before the exam and is shown in equation 2.3. A rule of thumb in logistic regression is if the probability is > 0.5 then the decision is true (i.e., 1) and false (i.e., 0) otherwise. Therefore the logistic regression model predicts if the student passes the exam, by calculating the probability and thresholding it based on 0.5.

$$P(y = 1) = P(Student\_Passed = True) = 0.82 * (number\_of\_hours\_studied) - 0.32. \tag{2.3}$$

2.2.3.2 Decision Trees

A decision tree is a supervised machine learning algorithm, which is mostly useful for classification problems and works for both continuous and categorical input and output variables. In simple terms, a decision tree uses a tree representation to make predictions, in which each branch represents a choice between a number of alternatives of an attribute in the internal nodes leading to a final decision in the leaf node. Decision trees where the output variable is categorical are known as Classification trees.

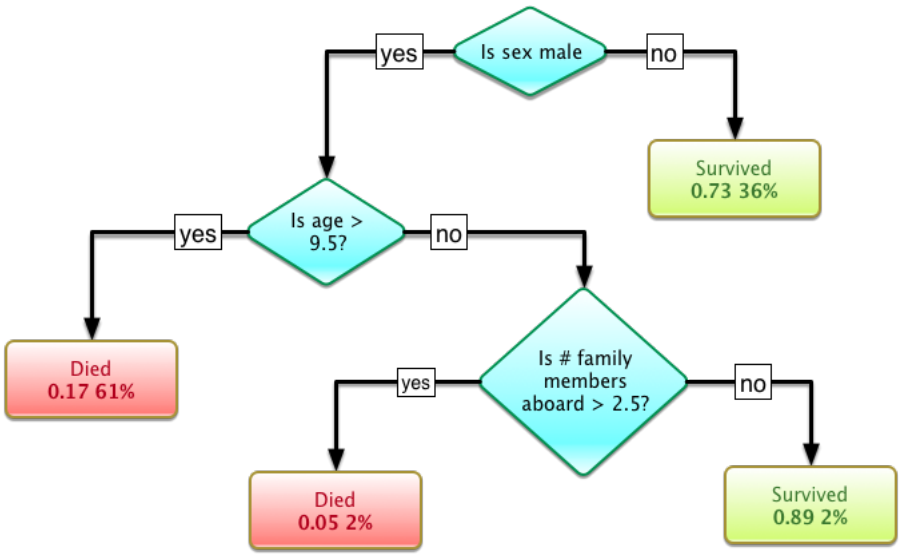


Figure 2.2: Classification tree of passenger survival in Titanic

The general approach of constructing a decision tree from the training data involves splitting the entire data at the root node into a subset based on a requirement (generally, a decision on a feature). The process of splitting based on decisions of different internal node features continues until a subset at a node has the same values of the target variable, or when splitting no longer adds value to the predictions. The main goal of decision trees is to find the best split of each node of the tree. But measuring the "goodness" of a given split is a subjective question. In practice, different metrics are used for evaluating splits. Two main metrics used to evaluate the splits are :

Gini impurity : It is used to compute the impurity of a data partition. It is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. It reaches its minimum (zero) when all cases in the node fall into a single target category.

Information gain : For each node of the tree, the information gain value represents the expected amount of information that would be needed to specify whether a new instance should be classified as yes or no, given that the example reached that node. The node with highest information gain value yields the best split.

For illustration purposes, consider the classification tree shown in figure 2.2 which describes the survival of passengers on the Titanic ship when it sinked. The figures under the leaf nodes show the probability of survival and the percentage of observations in the leaf. The tree was constructed using gini impurity as the split evaluation metric. From the tree we can make the following inferences:

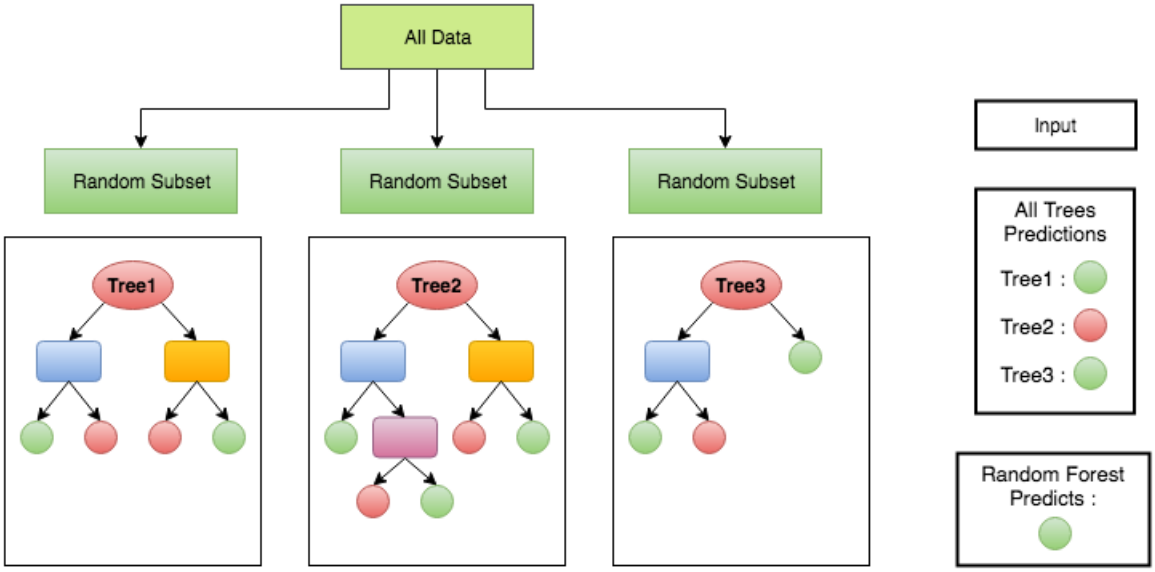


Figure 2.3: Example Random Forest Model

1. If the passenger was a male and had an age greater than 9.5 then he did not survive the Titanic sink.

2. Female passengers had a survival probability of 0.73.

3. If the passenger was male, had an age less than 9.5, and had less than 2.5 family members aboard the ship, he survived the sink with a probability of 0.89.

2.2.3.3 Random Forest Classifiers

Random Forest Classifier is another supervised machine learning algorithm that can be used for both classification and regression problems. It is generally classified into a special type of machine learning known as an Ensemble learning in which a combination of weak classifiers is used to form a strong classifier, which can be used to perform better predictive analysis. In case of the random forests, decision trees are the weak learners. Although decision trees perform really well on some datasets, they are generally called weak learners as they tend to have high variance when they are trained on different subsections of the same data. Here, variance refers to the spread of the predictions and occurs when a model is sensitive to small changes in the training data, which leads to overfitting. This happens due to the greedy approach of decision tree algorithm in selecting the best split to learn rules from the training data.

Random forests on the other hand create a number of decision trees during training by using different subsections of the training data. Moreover, the process of finding the root node and splitting the feature nodes occurs randomly in random forests. Once we have a forest of trees, decisions from different trees are combined to make a final decision regarding the data. This way the random forests will generalize the predictions better since the combination of decisions will not be sensitive to the trained data as each tree learns from different subsections of data. The more the number of trees in the random forest the better generalized the predictions.

Figure 2.3 explains the construction of random forests from the available input data. The initial training data is split into random subsets using which individual decision trees are constructed. The trees constructed from different subsets can have a different structure. Once we have the random forest, when a new input is given, it makes a prediction by using a voting of results from all the trees.

2.2.3.4 Support Vector Machines

Support Vector Machine (SVM) is a supervised machine learning algorithm, which can be used for both classification and regression problems. It is commonly known as a large margin classifier and is most useful for classification problems. In this algorithm, each data example with "n" features is plotted as a point in an n-dimensional space with the value of each feature being the value of a particular coordinate. Then, classification is done by finding the hyper-plane that separates the two classes best. A hyperplane is a line that splits the input variable space.

As an example, consider a binary classification problem with one input variable (x) and one output variable (y), which can be easily visualized in a two-dimensional space as shown in figure 2.4. The SVM classifier tries to find a hyperplane (in this case a line) that separates all of the input points. After the line that separates the input data is obtained, it can be used to make classifications by plugging in input values. If the equation returns a positive value, then the point is classified as being in the first class and if returns a negative value, then the point is classified as being in the second class. Alternatively, a point close to the line returns a value close to zero and it may be difficult to classify it. The perpendicular distance between the line and the closest data points is referred to as the margin. The best line that can separate the two classes is the line that has the largest margin, thus the name "Large-Margin classifier". These closest points play an important role in defining the line and in construction of the classifier, therefore they are called the support vectors. The hyperplane is learned from the training data using an optimization procedure

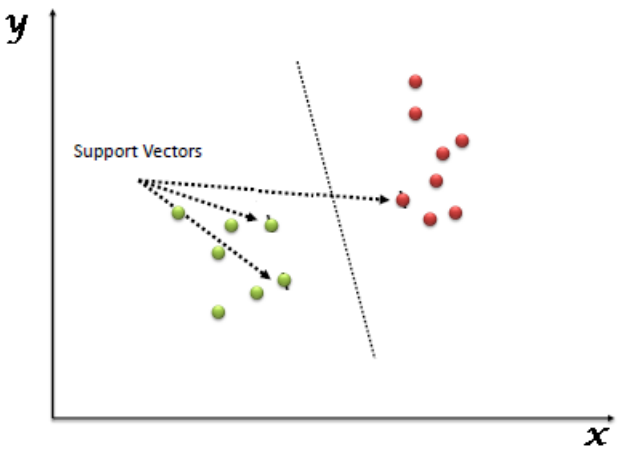


Figure 2.4: Example SVM Classifier

that maximizes the margin.

In practice, real world data is messy and cannot be perfectly separated with a hyperplane. In such cases, the SVM tries to relax the constraint of maximum-margin hyperplane by allowing some points in the training data to violate the separating hyperplane principle. This is generally done by varying a tuning parameter known as 'C' which defines the amount of the violation allowed by the classifier. The general approach is to try different values of C and choose the one that is the best fit for the data.

In SVM, it is easy to have a linear hyperplane between two classes. However, to create hyperplanes that separate non-linear data, SVM uses a technique called the kernel trick. In the kernel trick, SVM uses functions, which take the low dimensional non-separable input space and transforms it into a higher dimensional separable space. The functions used are known as kernels. The most commonly used kernels are linear kernels, polynomial kernels, and radial kernels.

2.2.4 Evaluation Methods

In this section, we will explain the various evaluation methods that are used in this thesis to assess and compare the performance of selected models.

In machine learning, the problem at hand is of making good predictions on unknown and unseen data by the learned models. In simple terms, we want to know for sure how well the model is able to generalize new previously unseen data and ensure that it is not just memorizing patterns from the training data. To make a better analysis of the generalizing power of the model, the data is generally split into training and test sets, where training data is used to create and evaluate the model while

test data is used to verify operational performance of the model. There are many performance measures in machine learning to evaluate our models. Moreover, having a performance measure defined for the problem also gives us the advantage of focusing our time effectively on improving our model predictions. We generally use the term 'metric' to refer to a performance measure in machine learning. Various metrics are available depending on whether the problem is of classification or regression. Since we are dealing with a classification problem, we will concentrate more on the metrics for classification problems.

2.2.4.1 Confusion Matrix

Confusion matrix is a representation of the predictions made by the classification model on a given dataset. It is a very important and useful metric in classification problems as it is not only useful to represent the models performance by itself, but can also be used to compute other metrics. A confusion matrix is composed of statistics such as true positives, true negatives, false positives, and false negatives, which are calculated using actual and predicted values. The idea of such a representation was to understand how many times the model was right in predicting a true or false value and how many times it failed to do so.

Figure 2.5: Example Confusion Matrix

To help understand the concepts of the confusion matrix representation, an example confusion

matrix is shown in figure 2.5. The matrix was generated from a total population of 231 students, where the predictions represent if the student passed a certain exam or not.

True Positive : The case where the actual value was true and the model also predicted it as true is known as a True Positive. The total number of such true positives from the population are represented on the bottom right corner of the confusion matrix. In Figure 2.5, the total number of true positives is 54 out of the total population of 231.

False Negative : The case where the actual value was true and the model predicted it as false is known as a False Negative. The total number of such false negatives from the population are represented on the bottom left corner of the confusion matrix. In Figure 2.5, the total number of false negatives is 20 out of the total population of 231.

False Positive : The case where the actual value was false and the model predicted it as true is known as a False Positive. The total number of such false positives from the population are represented on the top right corner of the confusion matrix. In Figure 2.5, the total number of false positives is 66 out of the total population of 231.

True Negative : The case where the actual value was false and the model also predicted it as false is known as a True Negative. The total number of such true negatives from the population are represented on the top left corner of the confusion matrix. In Figure 2.5, the total number of true negatives is 91 out of the total population of 231.

The confusion matrix in figure 2.5 is for a binary classification problem with 2 classes {'Passed', 'Failed'}. For a multi-class classification problem, each class has a row and a column in the confusion matrix and the problem is generally analyzed as a one-vs-all problem.

2.2.4.2 Classification Accuracy

Classification accuracy is the proportion of the number of correct predictions to all the predictions made by the model. This is the most common evaluation metric for classification problems. It can be calculated by using the values in the confusion matrix as the ratio of the sum of true positives and true negatives to the total size of the predicted population. The equation 2.4 represents the calculation of classification accuracy.

$$Classification\,Accuracy = \frac{True\,Positives + True\,Negatives}{size\,of\,predicted\,population} \tag{2.4}$$

The classification accuracy calculated from the confusion matrix shown in figure 2.5 would be (54+91)/231, which is 0.63.

2.2.4.3 Sensitivity

Sensitivity describes the probability of the prediction being true when the actual class is true. In simple terms, it describes how sensitive the model is when predicting positive instances. It is also known as "True Positive Rate" or "Recall" and is calculated as the ratio of true positives to the actual positive cases. It can be calculated by using the values in the confusion matrix as the ratio of true positives to the sum of true positives and false negatives. The equation 2.5 represents the calculation of sensitivity.

$$Sensitivity = \frac{True\,Positives}{True\,Positives\,+False\,Negatives} \tag{2.5}$$

The sensitivity calculated from the confusion matrix shown in figure 2.5 would be 54/(54+20), which is 0.73.

2.2.4.4 Specificity

Specificity describes the probability of the prediction being false when the actual class is false. In simple terms, it describes how specific the model is when predicting negative instances. It is calculated as the ratio of true negatives to the actual negative cases. It can be calculated by using the values in the confusion matrix as the ratio of true negatives to the sum of true negatives and false positives. The equation 2.6 represents the calculation of specificity.

$$Specificity = \frac{True\,Negatives}{True\,Negatives\,+False\,Positives} \tag{2.6}$$

The specificity calculated from the confusion matrix shown in figure 2.5 would be 91/(91+66), which is 0.58.

2.2.4.5 Precision

Precision defines the probability of the prediction being correct when the model identified it as positive. In simple terms, it describes how precise the model is when predicting positive instances. It is calculated as the ratio of true positives to the predicted positive cases. It can be calculated by

using the values in the confusion matrix as the ratio of true positives to the sum of true positives and false positives. The equation 2.8 represents the calculation of precision.

$$Precision = \frac{True\,Positives}{True\,Positives\,+ False\,Positives} \tag{2.7}$$

The precision calculated from the confusion matrix shown in figure 2.5 would be 54/(54+66), which is 0.45.

2.2.4.6 F_1 Score

F_1 score is calculated by taking a harmonic mean of the precision and recall of the model's predictions. It is also known as F-measure or balanced F-score. The equation 2.8 represents the calculation of precision.

$$F_1 = \frac{2 * (Precision * Recall)}{Precision + Recall} \tag{2.8}$$

The F_1 score calculated from the confusion matrix shown in figure 2.5 would be 0.56.

2.2.4.7 Area Under Curve (AUC)

In binary classification problems, the general rule of thumb is to use a probability threshold of 0.5 to make classification predictions. But for few scenarios, this threshold might not hold good and using a different threshold would be more appropriate. An Receiver Operating Characteristic (ROC) curve is the most commonly used method to visualize the performance of a binary classifier for different thresholds. It is obtained by plotting the True Positive Rate against the False Positive Rate. False positive rate is calculated as (1 - Specificity). From the ROC plot, we can calculate the Area Under the Curve (often referred to as simply the AUC) which is the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one [Faw06].

Figure 2.6 shows the ROC curve constructed from our example data captured by confusion matrix in the figure 2.5. In the plot, the area under the blue line refers to the AUC of the classifier, which in this case is 0.72. The dashed line in the diagonal represents the ROC curve of a random predictor whose AUC is 0.5.

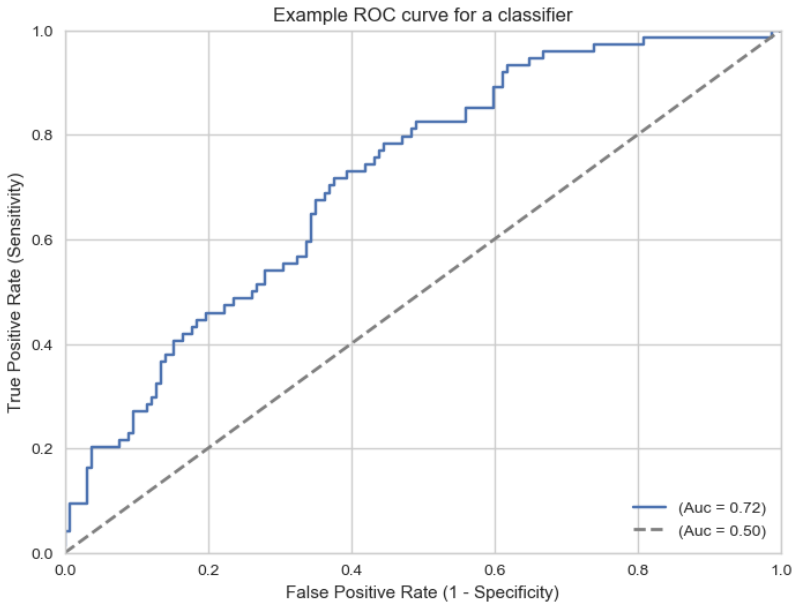


Figure 2.6: Example ROC curve

2.2.4.8 Cross Validation

So far, we discussed the calculation of various metrics based on the predictions obtained from single test data examples which was split from the original data. The approach of dividing the input data into training and validation data is also known as holdout method. The typical split rate is about 70% of input data into training and remaining 30% into validation data. These splits are generally created by random sampling. One problem with the holdout strategy to evaluate the models performance is that the random splits might not be fair and sometimes a specific combination of input data might always end up in validation data about which the model will have little information since it did not see them during training. As a result, the model cannot make good predictions on them. To tackle with such scenarios, we use a method known as cross validation.

In cross validation, the machine learning models are trained on several subsets of the available input data and evaluated on the complementary subset of the data. There are several available strategies based on the way the subsets are split. K-Fold cross validation and Stratified K-Fold cross validation are among the most popular and frequently used strategies.

In K-Fold cross validation, the data is divided into k subsets. In simple terms, the holdout method is repeated k times, such that at each time, one of the k subsets is used as the validation set and the other k-1 subsets are put together to form a training set. The error estimation is

averaged over all k trials to get the total effectiveness of our model. As can be seen, every data point gets to be in a validation set exactly once, and gets to be in a training set k-1 times. This significantly reduces the error in predictions as we are using most of the data for fitting, and also significantly reduces the spread of predictions as most of the data is also being used in validation set. Interchanging the training and validation sets also adds to the effectiveness of this method. The typical value of k suggested by many researchers is 10, which means that a total of 10 subsets are created from the data and during each run 9 subsets (i.e. 90% of the data) is used for training and the remaining 1 subset (i.e. 10% of data) is used for validation.

Stratified K-Fold cross validation is actually a variation of the K-fold cross validation that works better when applied on data that has huge imbalance in the response variable. In this approach, each fold contains approximately the same percentage of samples of each target class as the complete set.

In this subsection, we explained classification accuracy, sensitivity, specificity, recall, AUC and cross validation. Among the evaluation metrics, sometimes the classification accuracy can be a bad indicator of the model performance as it requires the output classes distribution in the original set to be balanced which is rarely true for real world datasets. As for other metrics, it becomes hard to compare three to four values to identify the model that performs best. Therefore, in these cases F_1 score and AUC can be used as a single value comparators of the model performances. Moreover, applying cross validation strategies to calculate the metrics would give evaluation measures that describe the model performance with higher confidence.

Chapter 3

Methodology

3.1 Data Collection

The Operations Support and Reporting Office at UNLV is responsible for extracting information on student enrollment and performance from the student data housed in the UNLV Data Warehouse and reporting the census to Integrated Postsecondary Education Data System (IPEDS). The data used in this thesis was obtained from the retention data submitted to IPEDS every year. The following subsections will describe more information about the data.

3.2 Data Preparation

In this section, we will discuss the steps taken to clean the dataset to obtain the input and output features from the original raw data that can be used by the machine learning models.

3.2.1 Data Description

The census data from the Operations Support and Reporting Office was exported into a .csv file to be used by the analysis. The raw data consists of 17,708 freshmen student records from five cohort years starting from 2012 to 2016. The data consisted of a variety of information about each student, such as pre-college academics, standardized test scores, the social and economic status of the student, freshmen-year academics, and whether the student was retained the following fall. On the whole, 41 attributes were captured for each student including the output variable. Table 3.1 describes the features and their types. The following subsections will describe, in detail, the steps taken to preprocess and transform the data to be used for the analysis.

Table 3.1: Description of data fields for first-year student data

No.	Feature	Type	Description
1	Cohort_Year	Multi Nominal	Returning Fall year of the student
2	Summer_Admit	Binary Nominal	If Student was admitted in prior summer
3	Alt_Admit	Binary Nominal	If Student was admitted differently
4	F1_EnrollCredits	Numerical	Count of credits enrolled in Fall 1(F1) term
5	F1_MidtermGPA	Numerical	F1 term mid-term GPA
6	F1_GPA	Numerical	Term GPA of student in F1 term
7	F1_Math_Type	Multi Nominal	Type of Math Course Taken in F1 term
8	F1_MathGradePass	Binary Nominal	F1 Math Grade Passed or Failed
9	F1_GPA_units	Numerical	Units completed in F1 term (for GPA)
10	F1_Complete	Binary Nominal	If the student completed F1 term
11	S1_Retain	Binary Nominal	If the student Enrolled in 1st spring term (S1)
12	S1_EnrollCredits	Numerical	Count of semester credits enrolled for S1 term
13	S1_GPA_units	Numerical	Units completed in S1 (for GPA)
14	S1_MathTaken	Binary Nominal	If Math taken in S1 term
15	S1_GPA	Numerical	Term GPA for S1 term
16	S1_Complete	Binary Nominal	If student completed S1 term
17	Yr1_MathTakenOverall	Multi Nominal	Type of Math taken in First year
18	Mom_Edu_Level	Multi Nominal	Highest education level of Mother
19	Dad_Edu_Level	Multi Nominal	Highest education level of Father
20	First_Gen	Binary Nominal	If the student is the first to attend college
21	Gender	Binary Nominal	Student Gender
22	Housing	Multi Nominal	on or off-campus housing
23	IPEDS_Race	Multi Nominal	IPEDS race/ethnicity data
24	Marital_Status	Multi Nominal	Marital status of the student in F1 term
25	CoreHSGPA	Multi Nominal	Core high school GPA of the student
26	UnwHSGPA	Multi Nominal	Unweighted high school GPA of the student
27	SAT_ACT_Score	Numerical	Combined ACT and SAT scores
28	ACT_SAT	Binary Nominal	If the student has taken ACT or SAT exam

Continued on next page

Table 3.1 – *Continued from previous page*

No.	Feature	Type	Description
29	F1_Residency	Binary Nominal	In state or Out of State in F1 term
30	F1_TuitionRes	Multi Nominal	Tuition status of the student in F1 term
31	F1_MillScholar	Binary Nominal	If Millennium Scholarship recipient in F1 term
32	WUE	Binary Nominal	Western Undergrad Exchange student
33	Age_Admit_Pcensus	Numerical	Age at time of preliminary census of first fall term
34	College1	Multi Nominal	First fall college
35	FA_INFO_RECD	Binary Nominal	FAFSA info recorded
36	PELL_Eligibility	Binary Nominal	If student is eligible for PELL grant
37	PELL_DISB_AMT	Numerical	PELL Grant Disbursed amount
38	MILL_DISB_AMT	Numerical	Millennium Scholarship disbursed amount
39	Other_SCHOL_AMT	Numerical	Other Scholarships disbursed amount
40	LOAN_DISB_AMT	Numerical	Student Loan disbursed amount
41	F2_Retained	Binary Nominal	If the student was retained in Fall 2 term(F2)

3.2.2 Feature Extraction

Feature extraction involves creating important useful features from the available raw data. Based on the problem statement and the available attributes from the collected raw data, the following features were generated.

3.2.2.1 Computation of SAT_ACT_Score Feature

The admissions office at UNLV requires all students to report scores of standardized exams such as ACT and SAT during their applications. However, few students who are admitted alternatively are exempted from submitting these scores. As a result of this, the ACT and SAT scores were missing for many students in the raw data. More specifically 56% of students did not have ACT scores and 28% did not have SAT scores. Hence to capture the importance of the standardized scores into our analysis, we converted the available ACT composite scores to equivalent SAT composite scores based on the conversion table from [Hal]. The term SAT_ACT_Score was used to refer to this converted scores feature.

3.2.2.2 Computation of F2_Not_Retained Variable

The main focus of this thesis is to identify the students who are at the risk of not being retained after their first year. In the raw dataset the flag F2_Retain captures the aspect of a student being retained and hence can be used to generate the outcome feature. We compute the complement of this flag to obtain the feature F2_Not_Retained which takes "0" if the student was retained and "1" if the student was not retained. The problem of identifying students who are at risk of not being retained after first year is then a binary classification task with F2_Not_Retained as the outcome variable.

3.3 Data Preprocessing

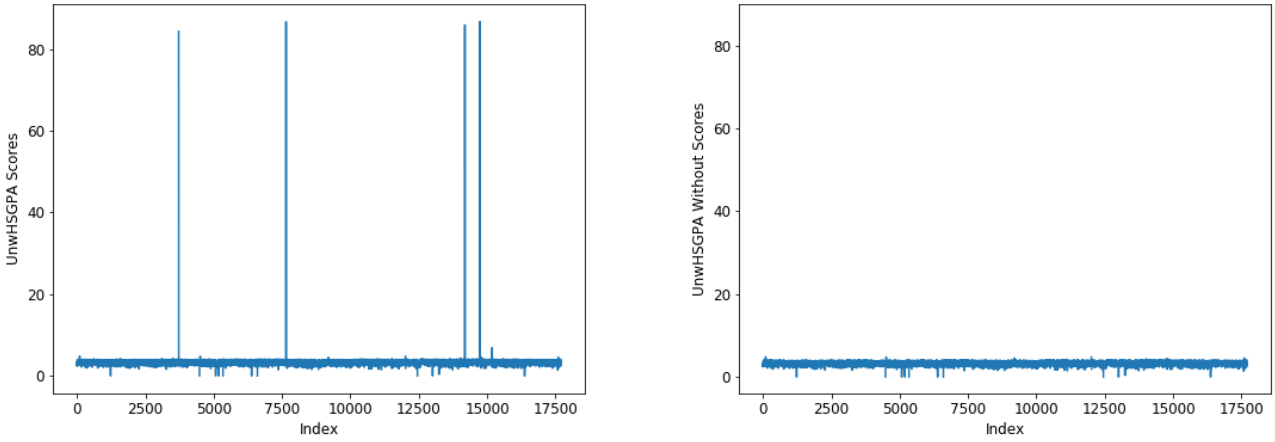
Data preprocessing is a technique that is used to convert raw data into a clean data set. In other words, whenever the data is gathered from different sources, it is collected in raw format which is not feasible for the analysis. Therefore, certain steps are executed to convert the data into a clean dataset. This technique is performed before the execution of any predictive analysis.

3.3.1 Handling Outliers in Data

The process of collecting data from different sources can introduce outliers into the dataset. This may be due to a faulty data extraction program or human error. In predictive data analysis, outliers can have a significant effect on the predictive power of the models as they can introduce huge noise into the model. Hence, it is always essential to remove outliers in the dataset before performing any analysis. Outliers were identified in our dataset and data distribution plots with index on x-axis and values on y-axis were used to demonstrate the presence of outlier in a feature. The general approach to handle outliers is to replace them with the mean of the attribute.

3.3.1.1 UnwHSGPA scores

There were a few outliers identified for the feature UnwHSGPA. As shown in figure 3.1a, we notice that few UnwHSGPA scores have values greater than 80 which is highly unlikely to be the UnwHSGPA score of a student. The outliers of the UnwHSGPA were replaced by mean value of the said dataset excluding all the outliers. Data distribution plot of the variable UnwHSGPA after removing outliers is shown in figure 3.1b.



(a) UnwHSGPA with Outliers

(b) UnwHSGPA without Outliers

Figure 3.1: Removing Outliers from UnwHSGPA

3.3.2 Handling Missing Values

In this subsection, we will discuss how we handled missing values in the dataset. Table 3.2 shows features along with the count of missing values. Machine learning deals with the usage of mathematical models on the data, therefore it cannot be applied to datasets that have missing values. The general approach is to fill the missing value with a suitable value. The process of filling missing values in the data is known as Imputation. Researchers frequently use the mean, mode, or median of the observed values to substitute for the missing field.

Table 3.2: Features with missing values and their count

Features	**Number of missing values**
SAT_ACT_Score	830
UnwHSGPA	1933
CoreHSGPA	3463
F1_MidtermGPA	286
F1_MathGradePass	4989

3.3.2.1 Imputing SAT_ACT_Score

The SAT_ACT_Score feature had 830 missing values. First, to understand the impact of missing values, we calculated the probabilities of the outcomes of the F2_Not_Retained feature with respect

Table 3.3: F2_Not_Retained outcome probabilities w.r.t availability of data points in SAT_ACT_Score

		F2_Not_Retained	
		0	**1**
SAT_ACT_Score	**Has value**	75.89%	24.11%
	Missing value	74.33%	25.67%

to the availability of data points in SAT_ACT_Score feature. In one case we ignored the records that had missing values and in the other case we included all the records. The probabilities are as shown in table 3.3. By looking at the table, it can be inferred that presence of missing values in SAT_ACT_Score feature had very little impact on the percentage of students not being retained. A total of 24.1% students were not retained when a value was present in the variable and 25.6% were not retained when the feature had a missing value.

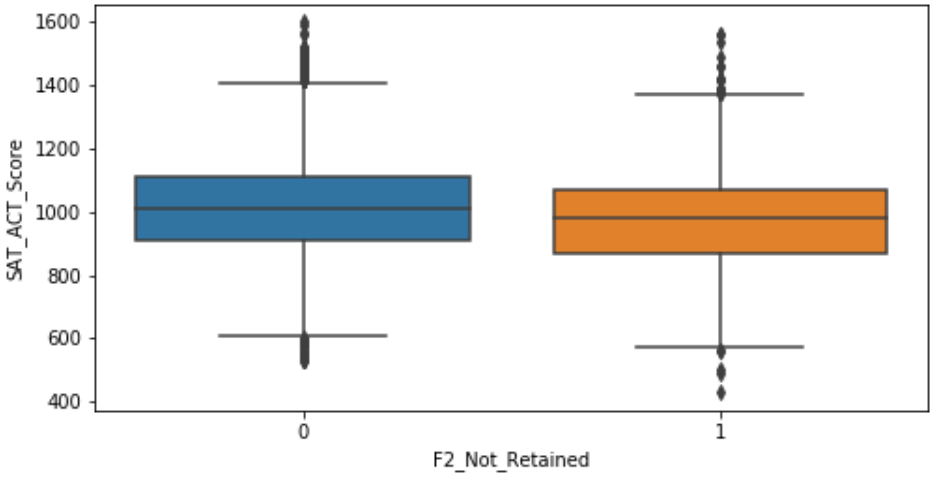


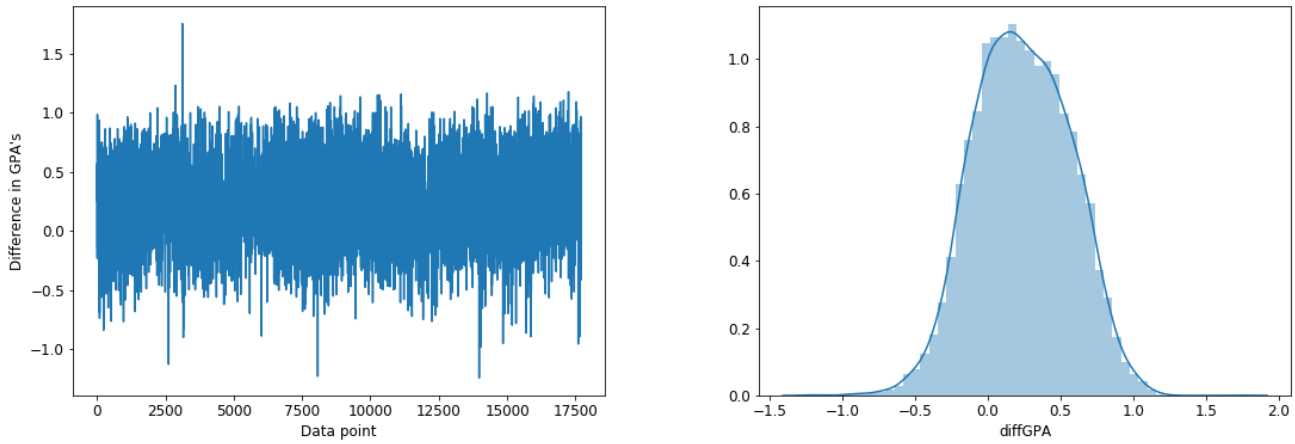
Figure 3.2: Boxplot of the SAT_ACT_Score vs F2_NotRetained

Moreover, the boxplot of the SAT_ACT_Score vs F2_NotRetained (figure 3.2) also shows that SAT_ACT_Score has very low impact on the student dropout. Based on the above facts, the missing values in this feature were replaced by the mean value.

3.3.2.2 Imputing CoreHSGPA and UnwHSGPA

The CoreHSGPA feature had 3463 missing values, whereas the UnwHSGPA had 1943 missing values. Before imputing, we analyzed the relation between these two features (figure 3.3). For this,

we calculated the difference between CoreHSGPA and UnwHSGPA of available records and plotted them as shown in figure 3.3a. We can clearly observe that majority of the differences were lying in the range of [-1,1], which indicates that the two features were very closely related. Similar pattern was revealed from the distribution plot of the variable differences as shown in figure(3.3b), which shows that the differences followed a Gaussian distribution with a mean of 0.23. Based on the above facts, we replaced the missing values in the feature with the available scores. The remaining missing values were replaced by the mean value of the respective feature.



(a) Difference between CoreHSGPA and UnwHSGPA

(b) Distribution

Figure 3.3: Analyzing difference between CoreHSGPA and UnwHSGPA

3.3.2.3 Imputing F1_MidtermGPA

The F1_MidtermGPA feature had 286 missing values, which adds up to a very small percentage in the overall data. We analyzed the distribution plot of F1_MidtermGPA (figure 3.4) and noticed that it follows a Gaussian distribution with mean value of 2.5272 and a standard deviation of 0.851. Considering the above information we imputed the missing value of the feature with its mean as it would not shift the distribution of the variable.

3.3.2.4 Imputing F1_MathGradePass

F1_MathGradePass is a categorical feature with values 'Y' or 'N', where a 'Y' represents if a student got a passing grade in math taken during his first fall term and 'N' if they got a failing grade. It had 4989 missing values in the variable. Such a large number of missing values for this feature is

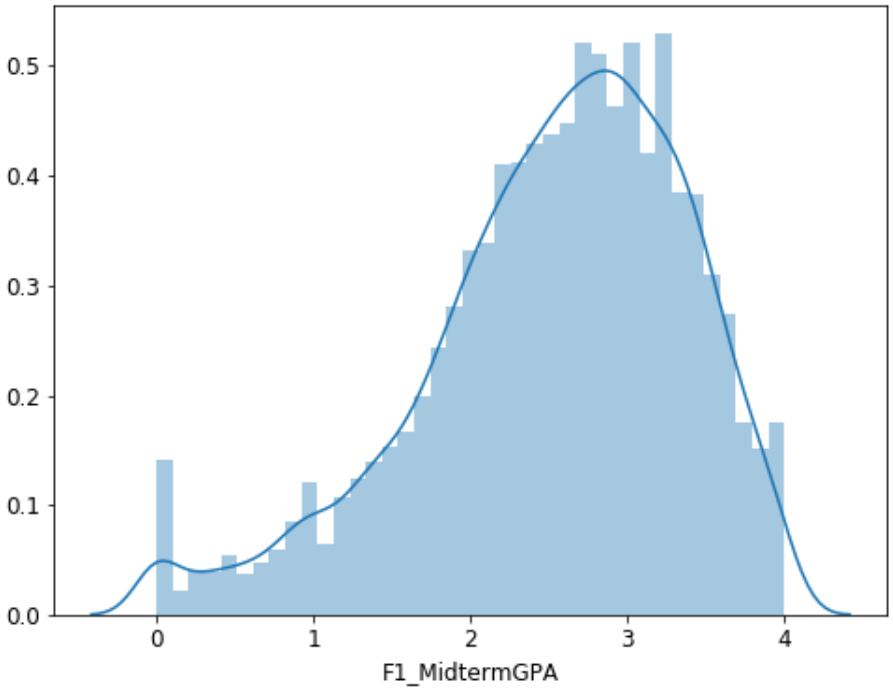


Figure 3.4: Distribution plot of the F1_MidtermGPA

attributed to the fact that a student who did not take any math course will not have a value in this feature. Hence the missing value was replaced with a 'N'.

3.3.3 Data Transformations

Now that we have a clean dataset, we need to apply transformations on the data before they can be inputted to a machine learning algorithm. This step is essential as our dataset has a lot of categorical features, which need to be converted to numerical values before we perform any further analysis.

The general approach is to encode the categories with numerical values. One-hot encoding is a popular encoding technique used in machine learning. In this technique a categorical feature is converted into a binary vector, where each possible value of the categorical feature is assigned to a dummy feature with a default value of '0' and the dummy feature that was the value of the categorical feature will have the value '1'. In simple terms, applying one-hot encoding to a categorical variable results in a one-hot vector, where only one element is non-zero, or hot. There were 23 categorical features in our dataset which were one-hot encoded to yield 85 features. We

Table 3.4: Tests based on type of input and output features

		Response Type	
		Quantitative	**Categorical**
Feature type	**Quantitive**	Correlation	Chi-Square Test
	Categorical	ANOVA	Chi-Square Test

noticed that applying transformations on our clean data resulted in a huge increase in the number of features available for analysis. Having more number of features can result in poor performance of the models as there might be some variables that are redundant and irrelevant to our prediction problem. To handle such scenarios, feature selection is useful as it automates the process of selecting features that are important to the prediction model. The next subsection describes in detail about the feature selection process used in this thesis.

3.3.4 Feature Selection

Feature selection is the process of identifying and selecting features from our data that contribute most to the prediction of our output variable. Feature selection methods are useful to identify and remove unneeded, irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model. There are various feature selection methods available in machine learning that can be applied to the available dataset. One key factor to consider before using a particular method is to have an idea of the models that will be used on the data.

One of the feature selection approaches deals with the idea of identifying the relationship of features with the output variable to decide on their importance. To find such relationships, we need to identify the data type of the features and the output variable. Table 3.4 shows the types of tests used based on input feature types and output types. The tests are as follows:

Correlation: In general, we use the Pearson correlation coefficient to measure the strength of a linear association between two numerical features. The higher the magnitude of the correlation coefficient, the greater the feature's influence on predicting the output variable.

ANOVA: ANOVA refers to Analysis of Variance, which is a collection of statistical models and their associated procedures (such as "variation" among and between groups) used to analyze the differences among group means.

Chi-Square: A chi-square test ($\tilde{\chi}^2$ test) is used to determine whether there is a significant difference between the expected frequencies and the observed frequencies in one or more categories.

In our dataset, the output response is categorical with classes {'Y', 'N'} and input features are both categorical and quantitative. Therefore, an appropriate choice of test to find relationships between the input features and output response is the Chi-square test. In machine learning the process of using such statistical tests on the dataset is known as univariate selection. Univariate selection using Chi-Square test was applied on the transformed dataset to yield 51 important features as opposed to the 85 features obtained after data transformation.

3.4 Exploratory Data Analysis

Exploratory data analysis (EDA) is an approach employed to analyze datasets and summarize their main characteristics, often with visual methods, without making any assumptions about its contents. It is an important step to perform before diving into statistical modeling or predictive analysis, because it provides the important information and context needed to develop an appropriate model for the problem. In this section, we try to uncover some important patterns inherent in our data with appropriate plots.

3.4.1 Cohort year vs. F2_Not_Retained

To understand the distribution of student dropouts in each cohort year, we plotted the proportions of the outcome variable F2_Not_Retained for each cohort year as shown in Figure 3.5. Each cohort year represents the students who re enrolled in Fall semester of that specific year after their first year. From the plot, we can infer that the number of students who drop out after their first-year varies very slightly over the recent years. More importantly, we see a pattern of increasing student dropout rates as opposed to an expected decrease even though efforts are being made by educators at UNLV to improve the student retention rates.

3.4.2 IPEDS_Race vs. F2_Not_Retained

In this subsection, we analyzed the patterns observed by different categories of the IPEDS_Race with respect to student retention. For this, we first plotted the different IPEDS_Race categories on the x-axis and the count of students belonging to that particular race on the y-axis as shown in figure 3.6. We observe that UNLV has a diverse student population with more students belonging to 'Asian', 'Hispanic' and 'White' races.

Although figure 3.6 gives us a good idea about the type of students at UNLV, it is not sufficient

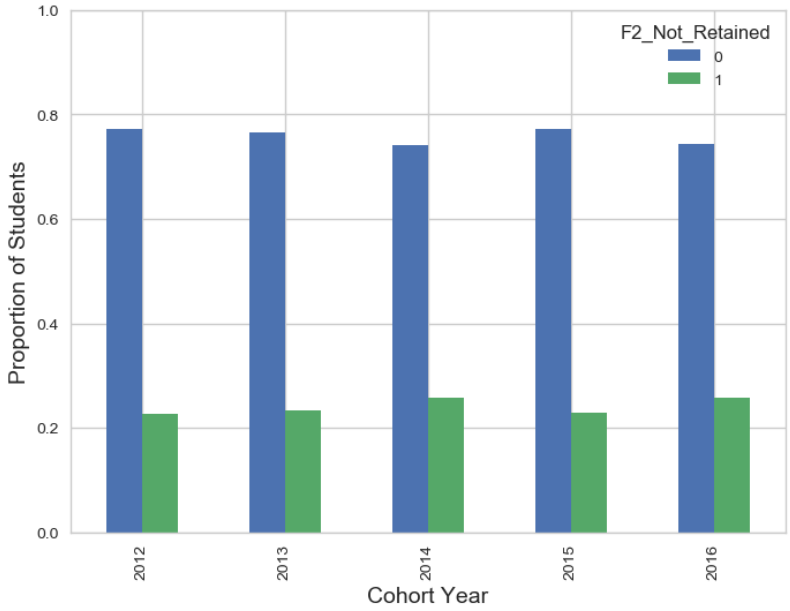


Figure 3.5: Academic Year vs F2_NotRetained

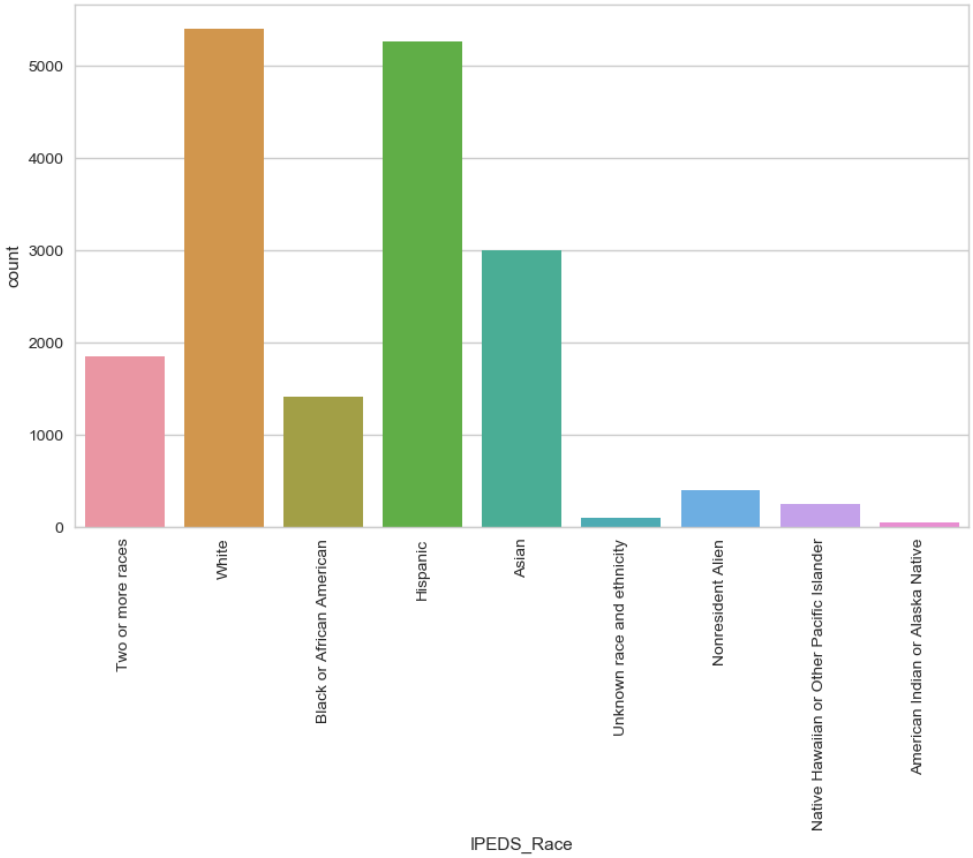


Figure 3.6: Count plot of IPEDS_Race feature

to help us understand the retention patterns among different categories. For this, we calculated the retention rates of each race category and tabulated them in table 3.5. Considering the student distributions in each race category and their retention rates from the table, we conclude that the 'Asian' race category had the highest student retention followed by 'Hispanic' and 'White' categories. On the whole, we observed that a student's race is useful in predicting if they will be retained after the first year.

Table 3.5: Retention Rates of each IPEDS_Race Category

IPEDS_Race	**F2_Retained**	**F2_Not_Retained**
Asian	85.75%	14.25%
American Indian or Alaska Native	53.48%	46.51%
Black or African American	66.83%	33.17%
Hispanic	74.34%	25.66%
Native Hawaiian or Other Pacific Islander	68.65%	31.35%
Nonresident Alien	86.14%	13.86%
Two or more races	74.23%	25.76%
Unknown race and ethnicity	75.26 %	24.74%
White	74.38%	25.61%

3.4.3 Mom_Edu_Level Vs F2_Not_Retained

The feature Mom_Edu_Level represents the highest education level of the student's mother. In this section, we analyzed the patterns observed by different categories of this feature with respect to student retention. For this, we plotted the different categories of Mom_Edu_Level feature on the x-axis and the count of students whose Mom's education level belonged to that category on the y-axis as shown in figure 3.7. We observed that the values of Mom_Education_Level of the students was highly distributed. It gives us information that most of the students Mom_Edu_Level had a minimum education level of 'High School' or more. However, this does not explain much about the effect of the mom's education level on the student's retention.

To get a better understanding of the effect of Mom_Edu_Level on the students' retention, we calculated the retention rates of students from each category of the feature and tabulated them in table 3.6. Considering the student distributions in each category and the retention rates of the feature from table, we notice some expected patterns such as more students were retained if Mom_Edu_Level was higher than 'Bachelor Level Degree'. But interestingly, we also noticed that

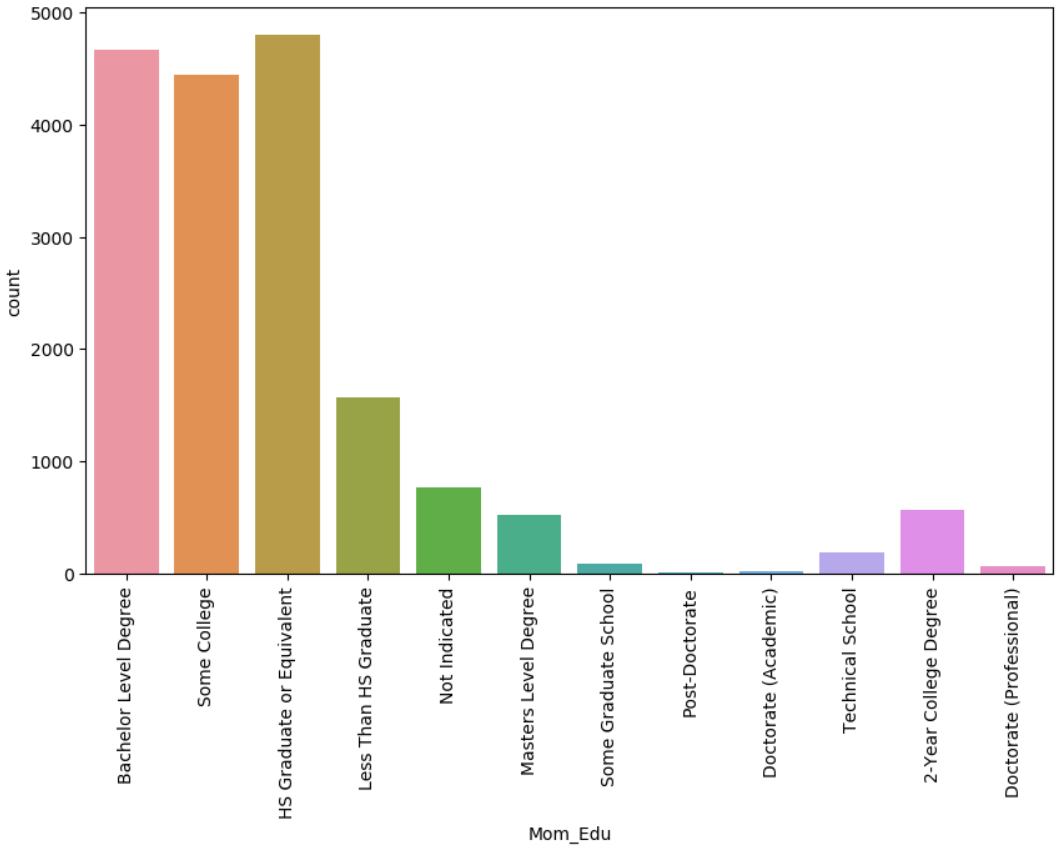


Figure 3.7: Count plot of Mom_Edu_Level feature

Table 3.6: Retention Rates of different categories of Mom_Edu_Level feature

Mom_Edu_Level	**F2_Retained**	**F2_Not_Retained**
Post-Doctorate	100.00%	00.00%
Doctorate (Professional)	84.37%	15.62%
Doctorate (Academic)	77.78%	22.22%
Masters Level Degree	76.14%	23.86%
Bachelor Level Degree	78.76%	21.24%
Some College	74.76%	25.24%
Some Graduate School	78.82%	21.17%
2-Year College Degree	74.82%	25.18%
HS Graduate or Equivalent	74.20%	25.79%
Less Than HS Graduate	74.52%	25.47%
Technical School	69.64%	30.36%
Not Indicated	77.46%	22.53%

the retention rate of students whose Mom_Edu_Level is less than 'HS Graduate' was high. This gives us a hint of the variability of patterns inherent in the data.

3.4.4 Dad_Edu_Level Vs F2_Not_Retained

The variable Dad_Edu_Level represents the highest education level of a student's father. In this subsection, we analyzed the patterns observed by different categories of the Dad_Edu_Level feature with respect to student retention. For this, we plotted the different categories of Dad_Education_Level feature on the x-axis and the count of students belonging to that category on the y-axis as shown in figure 3.8. We observed that the values of Dad_Education_Level of the students was highly distributed. It gives us information that most of the student's Dad_Edu_Level had a minimum education of High School or more. However, this does not explain much about the effect of the Dad_Edu_Level on the student's retention.

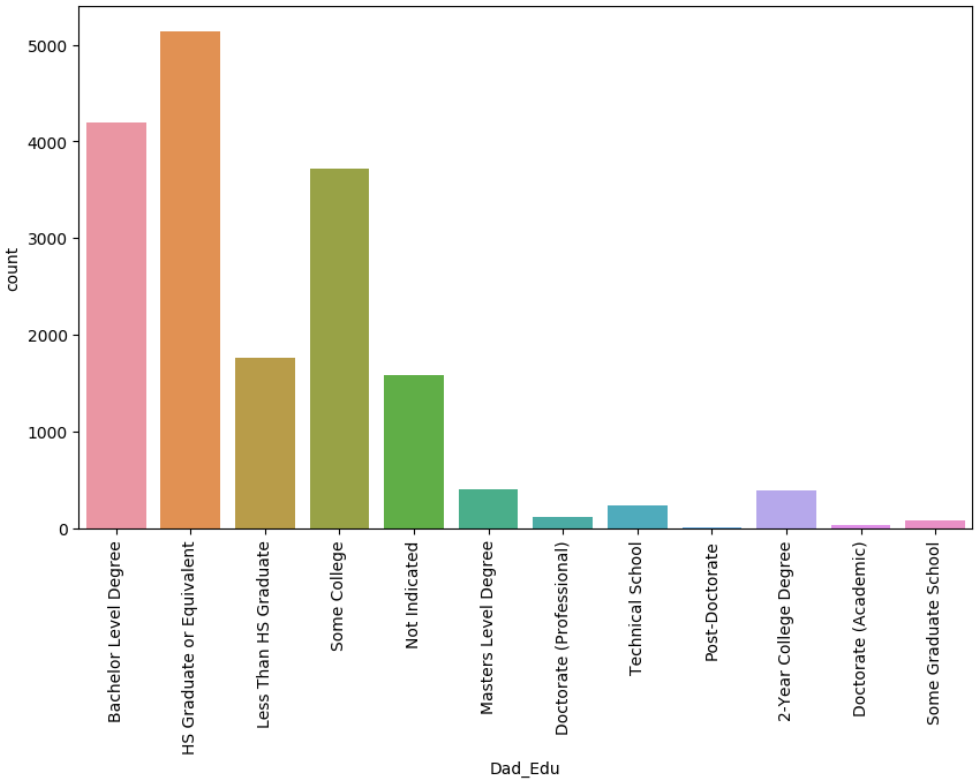


Figure 3.8: Count plot of Dad_Edu_Level feature

To get a better understanding of the effect of Dad_Edu_Level on students' retention, we calculated the retention rates of students from each category of the feature and tabulated them in table 3.7. Considering the student distributions in each category and the retention rates of the

Table 3.7: Retention Rates of different categories of Dad_Edu_Level feature

Dad_Edu_Level	**F2_Retained**	**F2_Not_Retained**
Post-Doctorate	86.66%	13.34%
Doctorate (Professional)	77.86%	22.14%
Doctorate (Academic)	86.84%	13.16%
Masters Level Degree	75.80%	24.20%
Bachelor Level Degree	80.34%	19.66%
Some College	76.30%	23.70%
Some Graduate School	76.62%	23.38%
2-Year College Degree	81.15%	19.75%
HS Graduate or Equivalent	73.36%	26.64%
Less Than HS Graduate	73.86%	26.14%
Technical School	69.50%	30.50%
Not Indicated	71.78%	28.22%

feature from table, we notice some expected patterns such as more students were retained if their Dad_Edu_Level was higher than 'Bachelor Level Degree'. But interestingly, we also noticed that the retention rate of students whose Dad_Edu_Level is less than 'HS Graduate' was high. This gives us a hint of the variability of patterns inherent in the data.

3.4.5 College1 Vs F2_Not_Retained

The feature College1 represents the name of the student's college during their first fall term. In this section, we analyzed the patterns observed by different categories of the College1 with respect to student retention. For this, we plotted the different categories of College1 feature on the x-axis and the count of students belonging to that category on the y-axis as shown in figure 3.9. We observed that the count of students belonging to a college was quite distributed with more students from 'College of Sciences' and 'Lee Business school'.

To get a better understanding of the effect of student's college on retention, we calculated the retention rates of each college and tabulated them in table 3.8. Considering the student distributions in each college and the retention rates of the colleges from the table 3.8, we noticed a few patterns such as more students were retained in 'College of Sciences' and 'College of Engineering', which had higher student counts. Even though other colleges had lesser student populations, their student retention rates were similar and averaged at about 75%.

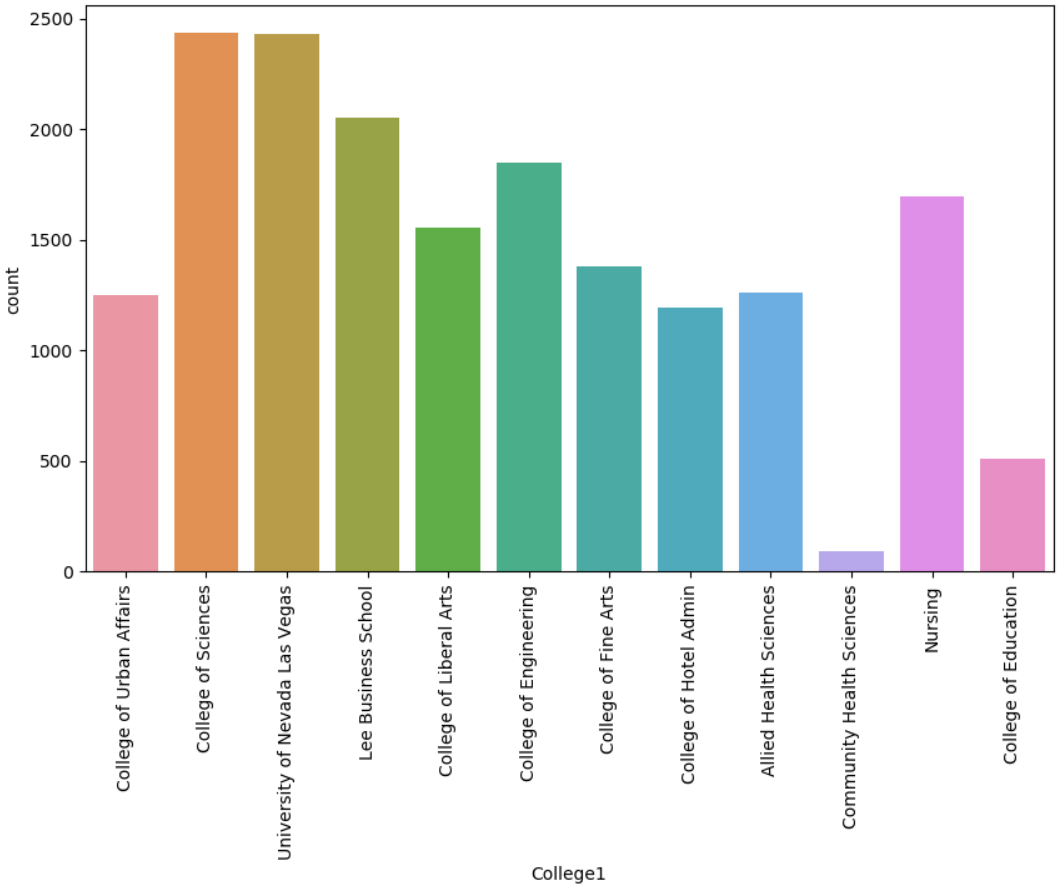


Figure 3.9: Count plot of College1 feature

Table 3.8: Retention Rates of different categories of College1 feature

College1	**F2_Retained**	**F2_Not_Retained**
College of Hotel Admin	80.98%	19.02%
College of Sciences	78.62%	21.38%
College of Engineering	77.63%	23.37%
College of Education	77.73%	23.27%
College of Fine Arts	76.61%	23.39%
College of Liberal Arts	76.06%	23.94%
Lee Business School	75.31%	24.69%
Nursing	75.14%	24.86%
Allied Health Sciences	73.53%	26.57%
University of Nevada Las Vegas	73.07%	26.93%
College of Urban Affairs	70.97%	29.03%
Community Health Sciences	64.51%	35.59%

Chapter 4

Experiments and Results

4.1 Data Splitting

Once the dataset was cleaned and feature selection was performed on it, there were 51 features excluding the F2_Not_Retained variable, which means we had 51 independent features that acted as predictors and were inputted into the model to predict the value of F2_Not_Retained (the output variable).

The data was split based on the cohort year, so it would be easy to analyze and test the retention rates on individual students as well as the whole cohort year. The count of students in each cohort year is shown in table 4.1.

Table 4.1: Count of students in each Cohort year

Cohort Year	**Count of Students**
2012	2996
2013	3585
2014	3656
2015	3715
2016	3756

From the entire dataset, the student records from the cohort years 2012, 2013 ,2014 and 2015 were combined to form the train_validate dataset. The remaining student records from the cohort year 2016 were used to form test_data dataset. The idea was to use the 4 years of data to train, validate, and evaluate the model. Once the model is trained and validated, it was used to make predictions on student records from the test_data dataset. This way, by not using any data from

the 2016 cohort year in training and validation processes, the model would treat it as totally unseen data and makes predictions which will not be based on any assumptions on the data of that year. There were 13,192 student records in the train_validate dataset and 3756 student records in test_data dataset.

Seventy percent (70%) of the train_test dataset, which was 9766 student records were used to train the selected models. The remaining thirty percent (30%), which was 4186 student records were used to validate, evaluate, and compare the performances of the trained models.

4.2 Experiments on Models

Once the training, validation and test data were created from the original dataset, each of the selected models were trained using the training data and evaluated on the validation data. The models which were trained are logistic regression, decision tree, random forest classifier and support vector machines.

Once each model was trained, we used it to make predictions and generate a confusion matrix on the validation data. The confusion matrix was used to calculate the classification accuracy, sensitivity, specificity, precision, F_1 scores, and to plot the ROC curve for the model. We also applied K-fold cross-validation on the training data to get the cross-validated AUC value, that was used to verify if the model results were not biased. If the cross-validated AUC score was similar to the one we got from the predictions on validation data, it means that the model was actually learning from the training data. Furthermore, we used the trained model to make predictions on test data and generate confusion matrix to compute the performance metrics which give us a better idea of the generalizing power of the model on previously unseen data.

All the above-mentioned performance metrics were used to compare the models to find the one that suits best for the UNLV student data.

4.2.1 Logistic Regression

A Logistic Regression (LR) model was created using the sklearn package in python and it was fitted on the training data. The trained model was then used to make predictions on the validation data. The confusion matrix generated from predictions on validation data is as shown in figure 4.1.

The logistic regression model built on the training data had a classification accuracy of 0.843 when used to perform predictions on the validation data. Table 4.2 shows other metrics calculated

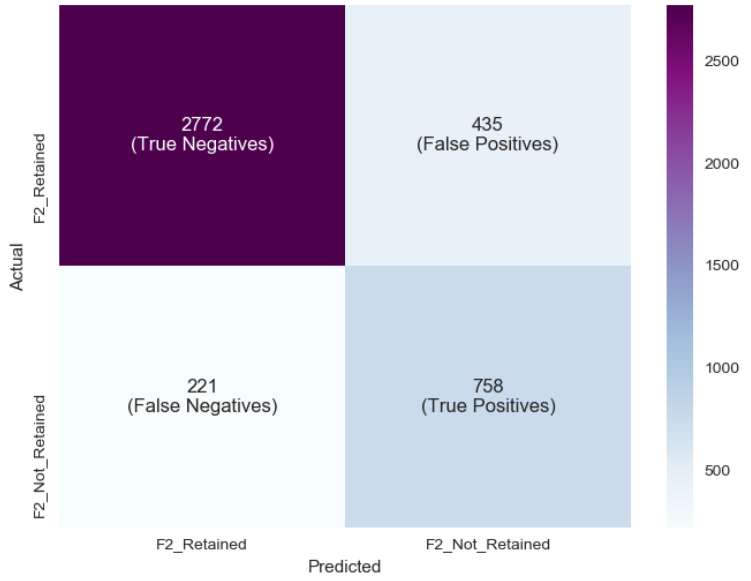


Figure 4.1: Confusion Matrix of LR model on validation data

from the confusion matrix in Figure 4.1.

Table 4.2: Computed metrics based on actual and predicted validation data values using LR model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
LR model	0.843	0.774	0.864	0.635	0.70	0.882

ROC curve for the LR model on validation data was plotted and is as shown in figure 4.2. K-Fold cross validation with the 'K' value of 10 was applied on the training data to get a cross validated AUC score of 0.88 which is similar to what we achieved on the validation data. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the validation data well.

The trained LR model was used to perform predictions on the test dataset that was created from the original dataset. The predictions on test data define the generalizing power of the LR model at the individual student level, as well as the cohort year level. The confusion matrix generated from predictions on test data is as shown in figure 4.3.

The logistic regression model built on the training data had a classification accuracy of 0.843 on test data. The table 4.3 shows other metrics calculated from the confusion matrix in figure 4.3.

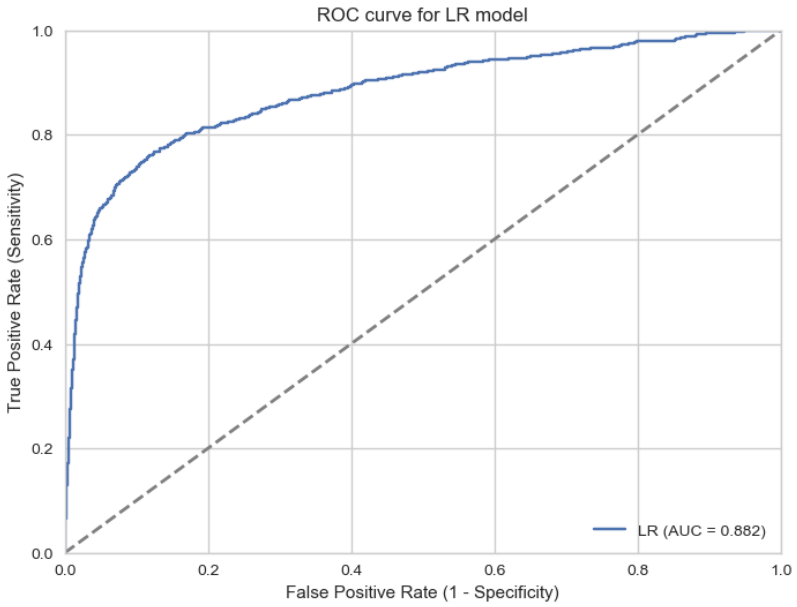


Figure 4.2: ROC curve for LR model on validation data

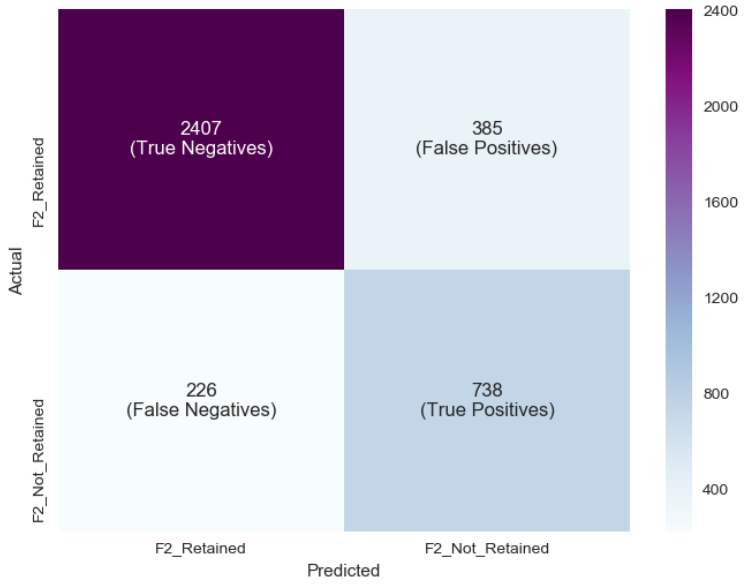


Figure 4.3: Confusion Matrix of LR model on test data (i.e., 2016 cohort year data)

Table 4.3: Computed metrics based on actual and predicted test data values using LR model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
LR model	0.837	0.766	0.862	0.657	0.71	0.883

4.2.2 Decision Trees

A Decision tree (DTree) model was created using the sklearn package in python and it was fitted on the training data. The trained model was used to perform predictions on the validation data. The confusion matrix generated from predictions on validation data is as shown in Figure 4.4.

Figure 4.4: Confusion Matrix of DTree model on validation data

The decision tree model built on the training data had a classification accuracy of 0.871 on the validation data. Table 4.4 shows other metrics calculated from the confusion matrix in figure 4.4.

Table 4.4: Computed metrics based on actual and predicted validation data values using DTree model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
DTree model	0.871	0.699	0.924	0.738	0.72	0.860

ROC curve for the DTree model on test data was plotted and is as shown in the figure 4.5 . K-Fold cross validation with the 'K' value of 10 was applied on the training data to get a cross validated AUC score of 0.86 which is similar to what we achieved on the validation data. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the validation data well.

The trained DTree model was used to perform predictions on test dataset that was created

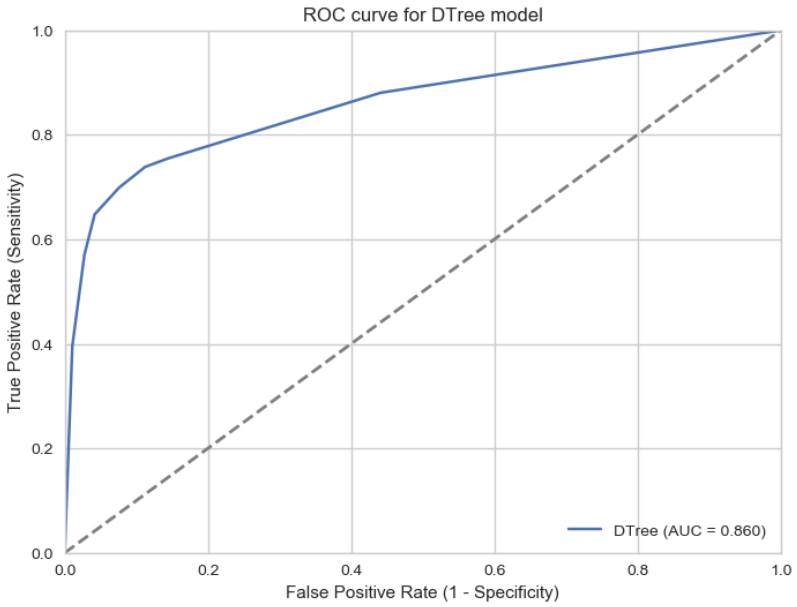


Figure 4.5: ROC curve for DTree model on validation data

from the original dataset. The predictions on the test data define the generalizing power of the Dtree model at the individual student level, as well as the cohort year level. The confusion matrix generated from predictions on test data is shown in figure 4.6.

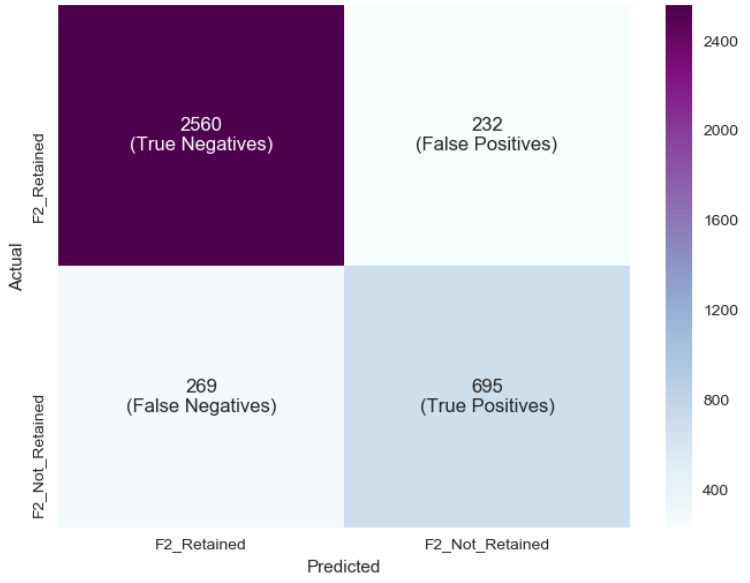


Figure 4.6: Confusion Matrix of DTree model on test data (i.e., 2016 cohort year data)

The decision tree model built on the training data had a classification accuracy of 0.867 on the test data. Table 4.5 shows other metrics calculated from the confusion matrix in figure 4.6.

Table 4.5: Computed metrics based on actual and predicted test data values using DTree model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
DTree model	0.867	0.721	0.917	0.750	0.74	0.87

4.2.3 Random Forest Classifier

A Random Forest classifier (RF) model was created using the sklearn package in python and it was fitted on the training data. The trained model was used to perform predictions on the validation data. The confusion matrix generated from predictions on validation data is as shown in Figure 4.7.

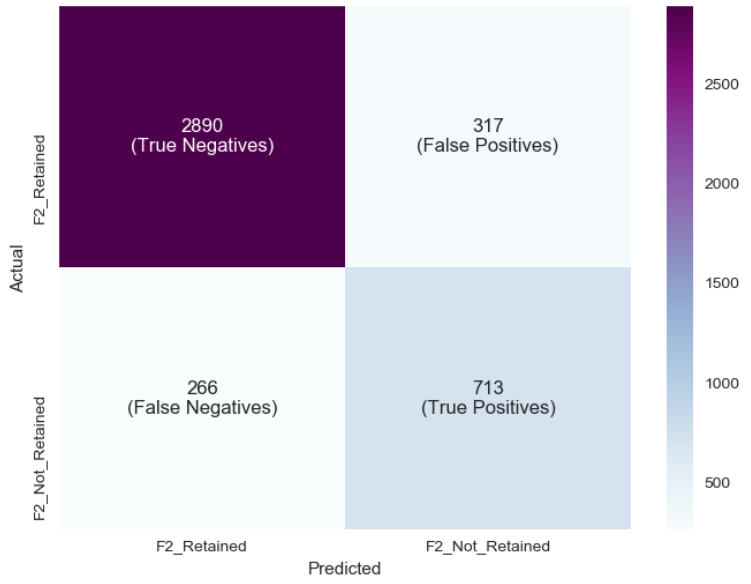


Figure 4.7: Confusion Matrix of RF model on validation data

The Random Forest model built on the training data had a classification accuracy of 0.861 when applied on validation data. Table 4.6 shows other metrics calculated from the confusion matrix in figure 4.7.

ROC curve for the RF model on the validation data was plotted and is as shown in the figure 4.8. K-Fold cross validation with the 'K' value of 10 was applied on the training data to get a cross

Table 4.6: Computed metrics based on actual and predicted validation data values using RF model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
RF model	0.861	0.723	0.903	0.695	0.71	0.876

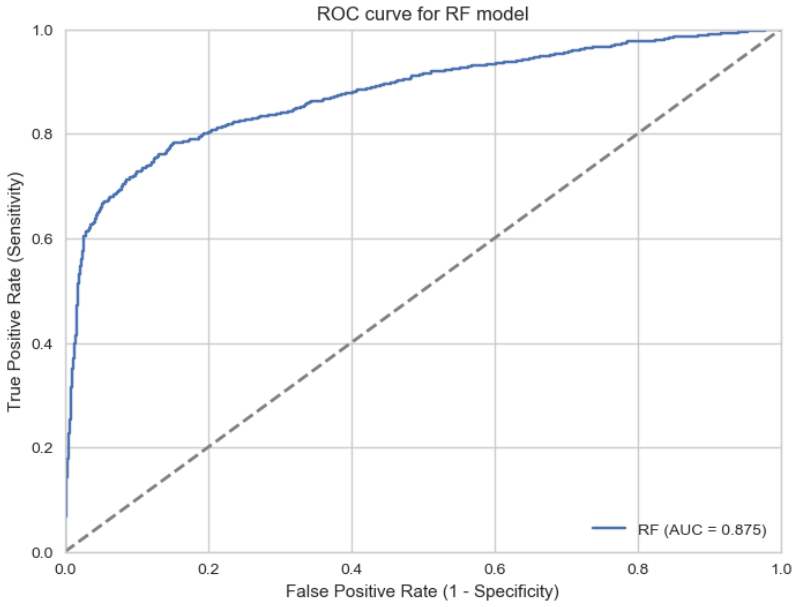


Figure 4.8: ROC curve for RF model on validation data

validated AUC score of 0.876 which is similar to what we achieved on the validation data. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the validation data well.

The trained RF model was used to perform predictions on test dataset that was created from the original dataset. The predictions on the test data define the generalizing power of the RF model at the individual student level, as well as the cohort year level. The confusion matrix generated from predictions on the test data is shown in the figure 4.9.

The random forest model built on the training data had a classification accuracy of 0.867 on the test data. Table 4.7 shows other metrics calculated from the confusion matrix in figure 4.9.

Table 4.7: Computed metrics based on actual and predicted test data values using RF model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
RF model	0.867	0.721	0.917	0.750	0.74	0.87

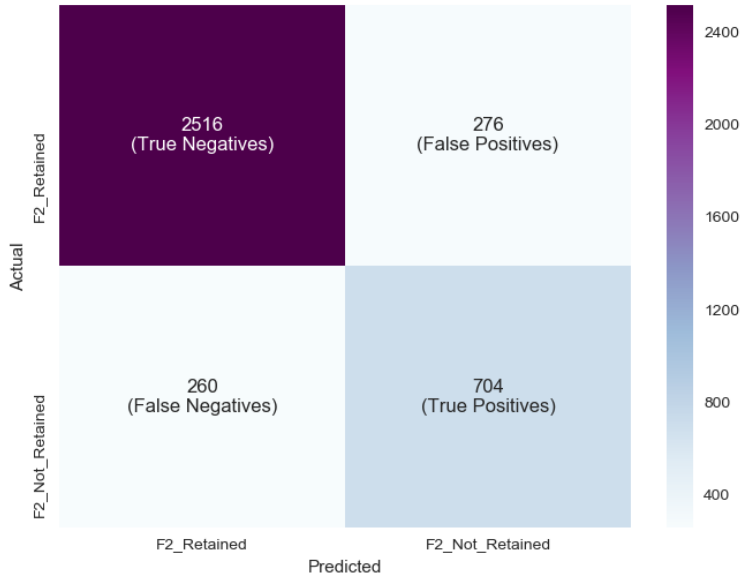


Figure 4.9: Confusion Matrix of RF model on test data (i.e., 2016 cohort year data)

4.2.4 Support Vector Machines

A Support Vector Machine (SVM) model was created using the sklearn package in python and it was fitted on the training data. The trained model was used to perform predictions on the validation data. The confusion matrix generated from predictions of the validation data is as shown in figure 4.10.

The SVM model built on the training data had a classification accuracy of 0.885 on validation data. Table 4.8 shows other metrics calculated from the confusion matrix in figure 4.10.

Table 4.8: Computed metrics based on actual and predicted validation data values using SVM model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
SVM model	0.885	0.588	0.976	0.881	0.71	0.857

ROC curve for the SVM model on validation data was plotted and is as shown in figure 4.11. K-Fold cross validation with the 'K' value of 10 was applied on the training data to get a cross validated AUC score of 0.86 which is similar to what we achieved on validation data. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the validation data well.

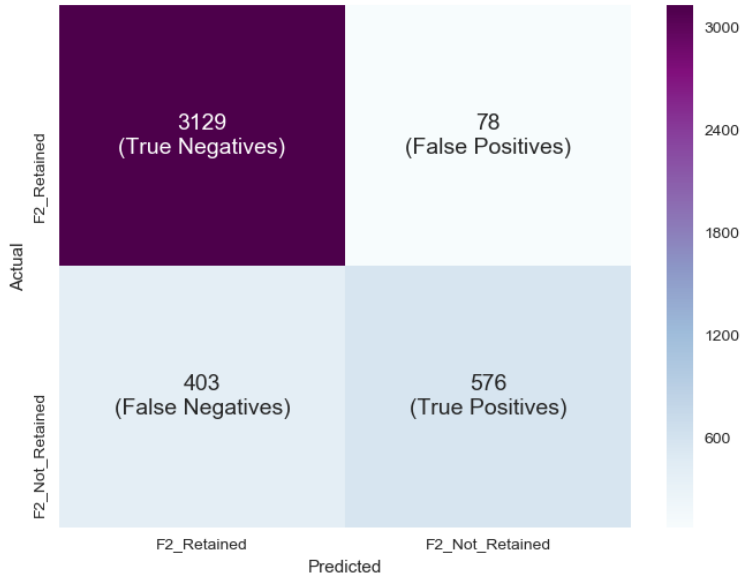


Figure 4.10: Confusion Matrix of SVM model on validation data

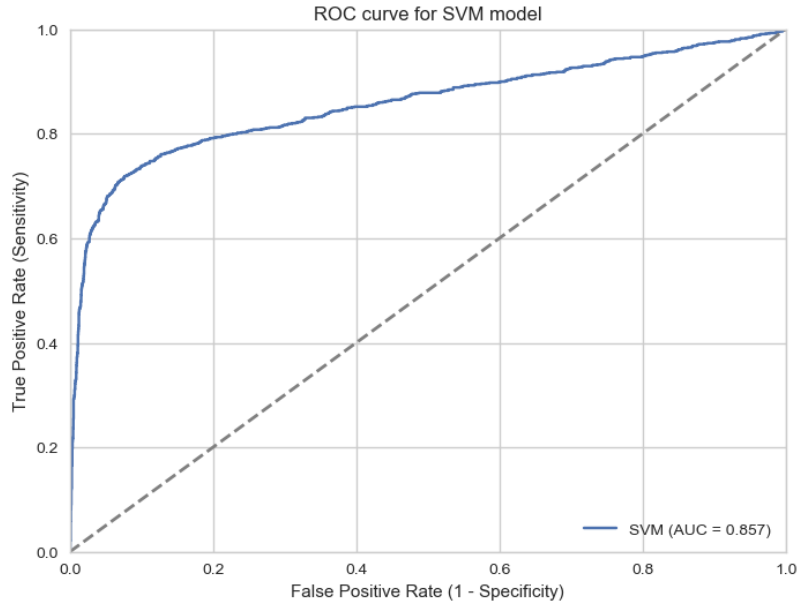


Figure 4.11: ROC curve for SVM model on validation data

The trained SVM model was used to perform predictions on test dataset that was created from the original dataset. The predictions on the test data define the generalizing power of the SVM model at the individual student level, as well as the cohort year level. The confusion matrix generated from predictions on test data is shown in the figure 4.12.

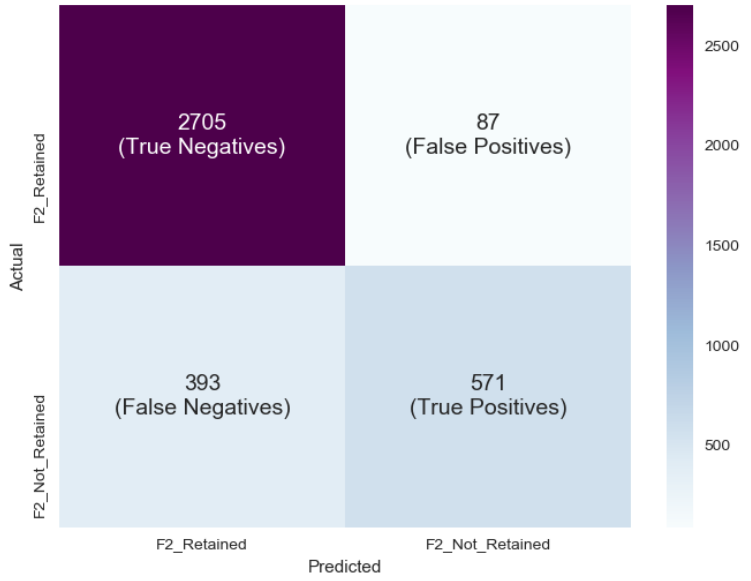


Figure 4.12: Confusion Matrix of SVM model on test data (i.e., 2016 cohort year data)

The support vector machine model built on the training data had a classification accuracy of 0.872 on the test data. Table 4.9 shows other metrics calculated from the confusion matrix in figure 4.12.

Table 4.9: Computed metrics based on actual and predicted test data values using SVM model

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
SVM model	0.872	0.592	0.969	0.868	0.70	0.860

4.2.5 Comparison of the Models

The results from the experiments conducted in the previous subsection will be compared to find the best model that is suitable for the UNLV student data.

4.2.5.1 Measures used to Compare Models

In our Experiments, we used the trained models to make predictions on the validation data and test data. From the predictions, we calculated various metrics that explain the performance of the model. However, comparing models' performance by combining results of all the metrics is a difficult task. Moreover, in classification problems using machine learning, it is important to

identify the metrics that better represent the problem being modeled. In our case, we were more focused on identifying the students who will not be retained in the next fall semester (i.e., class '1' of the F2_Not_Retained variable), which was the true positive case of our models' prediction. The metrics accuracy and specificity capture the effect of output class populations on learning (i.e., the model would learn more about the output class that has higher population). Since our data has more population of students who were retained, it was expected to get high classification accuracies and specificities as the model will predict more number of retained students correctly compared to the not retained students. On the other hand, sensitivity represents the true positive rate and precision represents the confidence of the model in identifying the true positives. F_1 score and AUC are metrics based on sensitivity and specificity and thus are also good representations of the model performance. Therefore sensitivity, precision, F_1 score and AUC were used to compare the models.

The metrics calculated from predictions on validation data for different models are summarized in table 4.10. The ROC curve plots of all the models on the validation data are as shown in figure 4.13.

Table 4.10: Comparison of metrics from different models on the validation data

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
LR model	0.843	0.774	0.864	0.635	0.70	0.882
DTree model	0.871	0.699	0.924	0.738	0.72	0.860
RF model	0.861	0.723	0.903	0.695	0.71	0.876
SVM model	0.885	0.588	0.976	0.881	0.71	0.857

Logistic regression had the highest sensitivity rate and AUC scores compared to all the other models. It had a very low precision of 63.5%, which indicates that the model had falsely predicted more students as not retained who actually were retained. However, our main focus is to identify at-risk students who will not be retained, so we can provide assistance to retain them. Therefore, providing assistance to the few falsely identified not retained students is still a positive approach as it may at the least help in increasing the academic performance of those students.

In the case of the decision tree model, we observed good accuracy and specificity values. In fact, it had the highest classification accuracy on the validation data compared to all the other models. Moreover, its sensitivity and AUC scores were lower compared to the logistic regression and random forest models, which indicates that it was performing poorly in terms of identifying not retained students. The random forests on the other hand had the second best sensitivity and

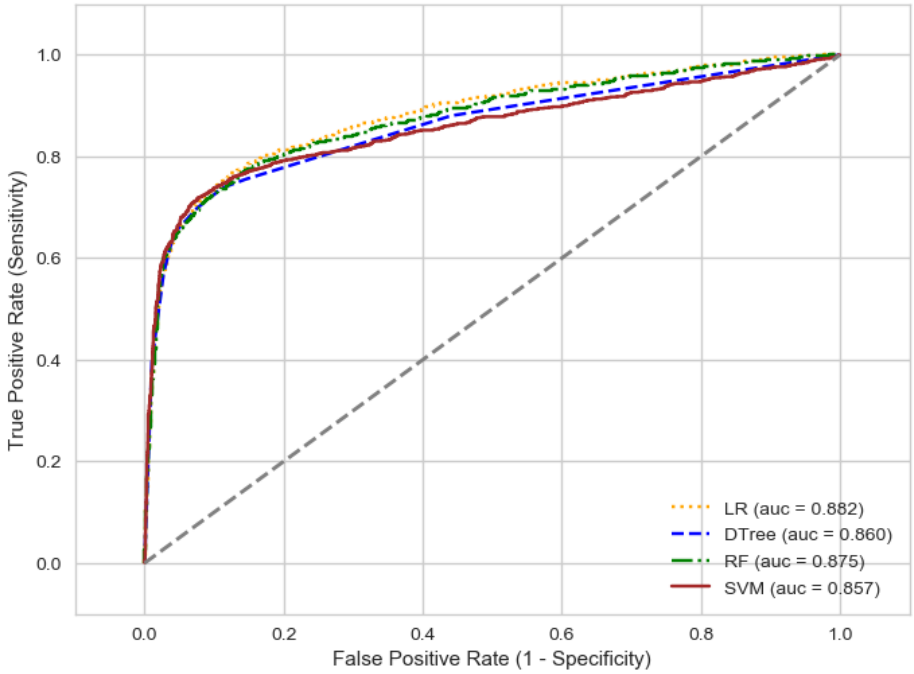


Figure 4.13: ROC curve's of all models on validation data

AUC scores compared to all other models. Moreover, it had a little better precision and F_1 score compared to logistic regression.

The SVM model had a very high classification accuracy and specificity compared to other models. However, its sensitivity and AUC scores were the lowest which indicates that it was performing poorly in terms of identifying not retained students.

The metrics calculated from predictions on test data for different models are summarized in table 4.11. The ROC curve plots of all the models on the test data are as shown in figure 4.14.

Table 4.11: Comparison of metrics from different models on the test data

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
LR model	0.837	0.766	0.862	0.657	0.71	0.883
DTree model	0.867	0.721	0.917	0.750	0.74	0.87
RF model	0.867	0.721	0.917	0.750	0.74	0.87
SVM model	0.872	0.592	0.969	0.868	0.70	0.860

The models performance on test data show us that the metrics are close enough to the ones we achieved on validation data. This gives us a hint of the reliability and uniformity of the model predictions on the combined cohort year data, as well as individual student level data, which is our

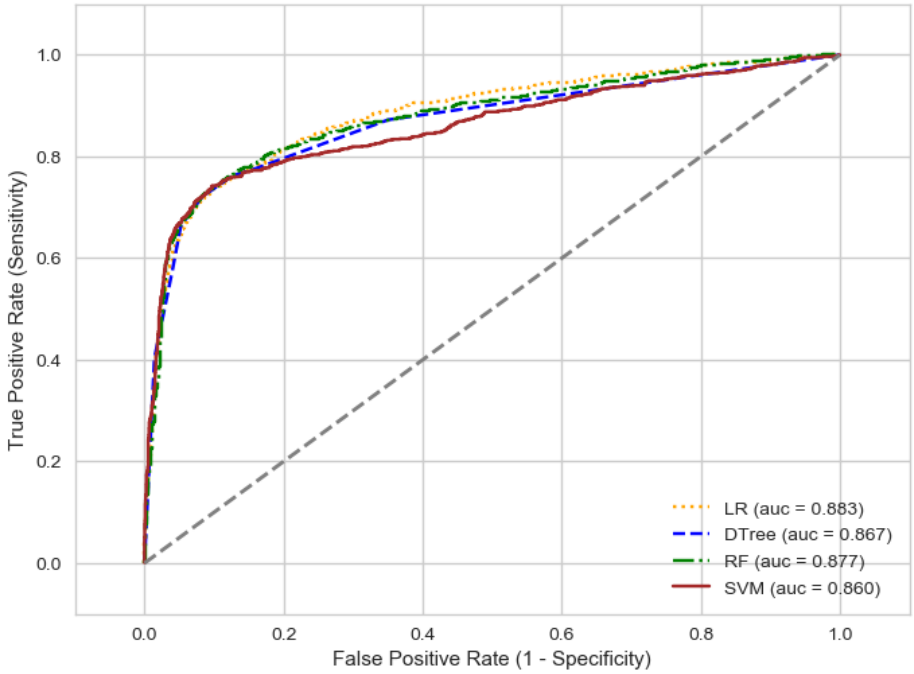


Figure 4.14: ROC curve's of all models on test data

primary focus.

Based on the above observations, we conclude that the logistic regression and random forest classifiers are good prediction models for the UNLV dataset.

4.2.6 Feature Importance calculation based on Selected Models

After comparing the models, the two models that were selected were logistic regression and random forest classifier. These models were used to identify the most significant factors that contributed to the prediction of the not retained students. To calculate the significant factors, the two models were retrained using the train and test data combined.

4.2.6.1 Top Fifteen Predictors for Logistic Regression Model

As explained earlier in section 2.2.3.1, the process of fitting a logistic regression model on data involves finding a linear combination of input factors along with coefficients that are calculated during training. Once we have the coefficients, the input features of a record are linearly combined with the coefficients and the result is fed into a sigmoid function to obtain the probabilities. Here, the coefficients actually represent the contribution of the feature towards the output prediction. The LR model which was fit on the train and validation data was used to obtained the coefficients

of each feature. The features were then ordered based on the magnitudes of their coefficients. The top fifteen features from the ordered set were selected and plotted in the order of significance as shown in figure 4.15.

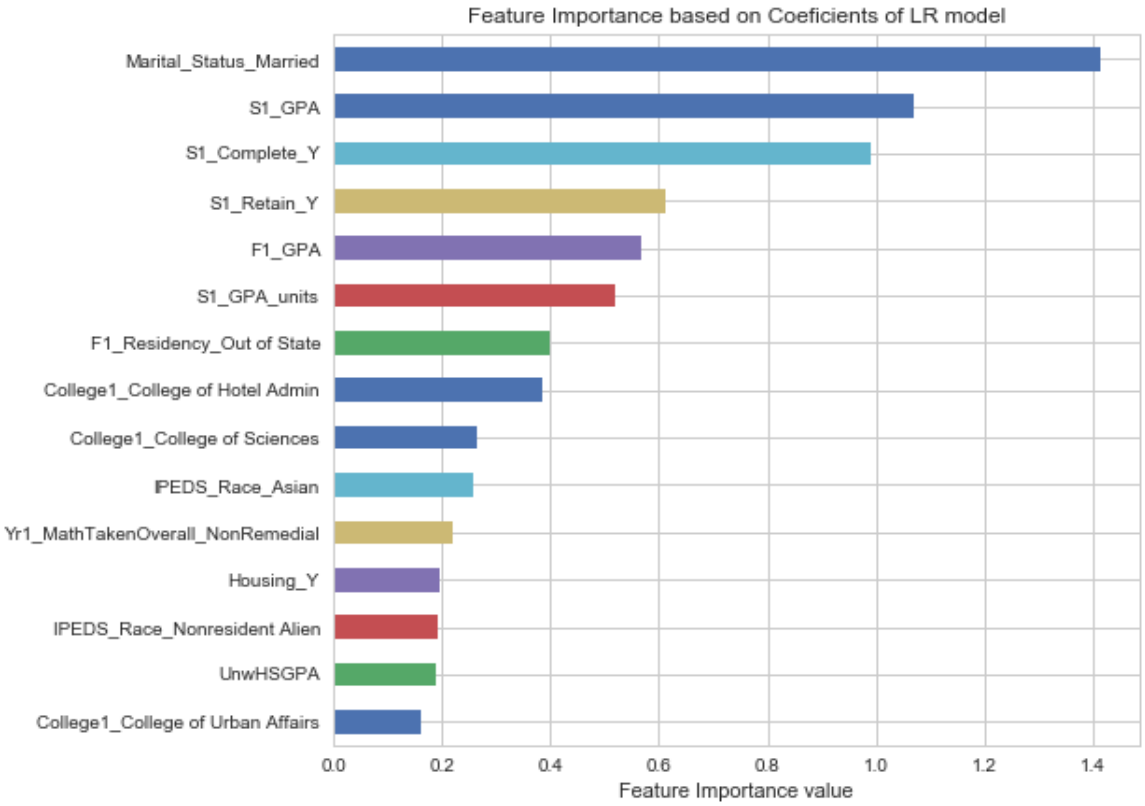


Figure 4.15: Top 15 Predictors of Logistic Regression Model

We noticed that the student's academic performance during their high school and the spring semester of first year, as well as some factors that represent a student's background were contributing more towards the output prediction than the other factors used in the training.

4.2.6.2 Top Fifteen Predictors for Random Forest Model

As explained earlier in section 2.2.3.3, the process of fitting a random forest model on data involves creating various trees using random splits on different features. Random forests can use gini impurity or information gain as the metrics to measure the goodness of each split. In most cases gini impurity is favored due its less computational complexity. During training, random splits are made

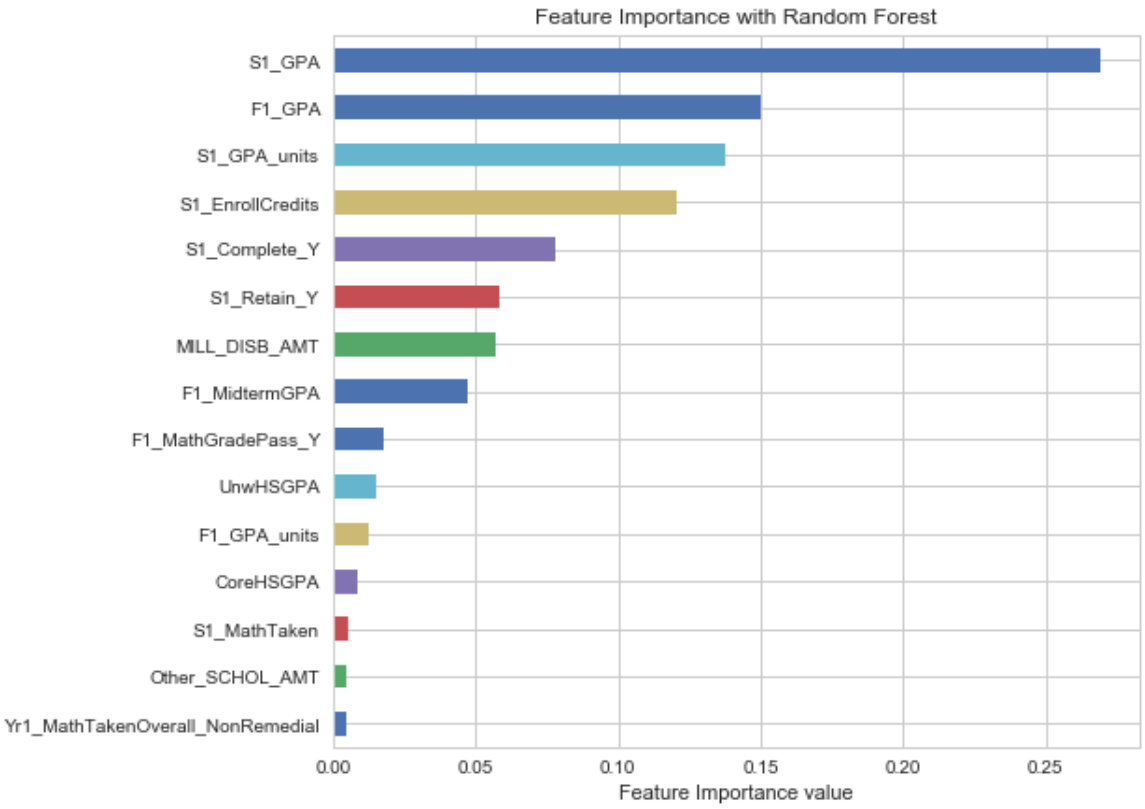


Figure 4.16: Top 15 Predictors of Random Forest Model

in different trees by analyzing the decrease in the gini impurity due to the split by a certain feature. This decrease in the gini impurity for a split on each feature can be used to identify the feature importances. The random forest model, which was fit on the train and validation data was used to identify and order the decrease in the mean gini impurity of the features. The top fifteen features from the ordered set were selected and plotted in the order of significance as shown in figure 4.16.

We noticed that a student's academic performance during high school and his first year at UNLV, as well as some factors that represent the students financial status were identified as contributing more towards the output prediction. This clearly points that improving a student's academic performance can boost their retention.

4.2.7 Ranking Students based on the Risk of Not Being Retained

As explained earlier in section 2.2.3.1, the logistic regression model calculates the probability of getting the positive class of the output variable based on which it makes the final class predictions. If the probability is greater than or equal to 0.5 then it predicts the outcome to be the positive class and if the probability is less than 0.5, then it predicts the outcome to be the negative class. In our case, the logistic regression model calculates the probability of the student not being retained in the next fall semester. These probabilities can be interpreted as a student's risk of dropping out. We can then rank students based on their risk scores, which can be used by the educators to identify students who need the most support as part of intervention programs.

We used the trained logistic regression model to obtain prediction probabilities on test data. We chose test data since we would like to find the students who are not retained for a cohort year that the model has not seen during training. These probabilities were then ordered to obtain the ranking of students based on their risk of dropping out. Figure 4.17 shows the top ten students who are at the risk of not being retained. The index in the figure represents the record id which identifies the student in the original dataset.

Index	RiskScore
17403	0.993864
14876	0.992666
15804	0.992618
14993	0.991109
15949	0.991067
15263	0.990655
17636	0.990589
15053	0.990435
15890	0.990113
16519	0.99002

Figure 4.17: Top 10 Students at the risk of not being retained in F2 term

Using this approach, we can rank the students of a newly available cohort year data of UNLV based on their risk scores. The instructors can then identify the top 'k' students to whom they can provide additional academic help to improve their retention chances. The value of 'k' can be chosen based on the availability of intervention resources.

Chapter 5

Conclusion And Future work

In this thesis, we used machine learning to predict the first-year student retention rates at UNLV. For this, we identified and collected important student data such as pre-college academics, financial information, and academic performance of students during their first-year, from the data housed in UNLV data warehouse. We performed data pre-processing to clean the data and applied feature selection to select the most significant features for the analysis.

The pre-processed data was used to train the machine learning models logistic regression, decision trees, random forest classifiers and support vector machines. The trained models were evaluated and compared for performances using metrics of classification problems calculated from predictions on test data. An analysis of the results revealed that the logistic regression model and random forest model performed better on the UNLV student data. We used these two models to identify top fifteen features that contributed the most towards student retention prediction. The top predictors from both the models showed us that the students' academic performance had a major influence on their retention. Therefore, we can formulate retention programs that would improve the academic performance of at-risk students, thus improving their chances of retention.

The operational performance of the models was also calculated and evaluated at the cohort level using test data (in this case 2016 cohort year data) which gives us a brief understanding of the models' generalization capacity on new data. In the future, the models can be used to make predictions on the the latest available census by replacing the test data with the newly available cohort year data. Additionally, we proposed the approach of ranking students of the latest census based on their risk scores, which can be used by the educators to concentrate their intervention resources effectively.

This research, helped us identify some important patterns in first-year student retention data

at UNLV. For future work, we would like to apply more advanced machine learning algorithms for predictions by including more financial information of the student, as well as to perform the identification of the not retained students at an earlier time that would give the educators more time to intervene and improve the at-risk students' retention chances.

Bibliography

[AC17] Olugbenga Adejo and Thomas Connolly. An integrated system framework for predicting students? academic performance in higher educational institutions. *International Journal of Computer Science and Information Technology (IJCSIT)*, 9(3):149–157, 2017.

[AH14] Ruba Alkhasawneh and Rosalyn Hobson Hargraves. Developing a hybrid model to predict student first year retention in stem disciplines using machine learning techniques. *Journal of STEM Education: Innovations and Research*, 15(3):35, 2014.

[AMBS99] Paul A. Murtaugh, Leslie Burns, and Jill Schuster. Predicting the retention of university students. 40:355–371, 06 1999.

[Ast12] Alexander W Astin. *Assessment for excellence: The philosophy and practice of assessment and evaluation in higher education.* Rowman & Littlefield Publishers, 2012.

[Bra02] John M Braxton. Introduction to special issue: Using theory and research to improve college student retention. *Journal of College Student Retention: Research, Theory & Practice*, 3(1):1–2, 2002.

[BS16] Melissa A Bingham and Natalie Walleser Solverson. Using enrollment data to predict retention rate. *Journal of Student Affairs Research and Practice*, 53(1):51–64, 2016.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[Hal] Edwards Halle. SAT / ACT Prep Online Guides and Tips. `https://blog.prepscholar.com/act-to-sat-conversion`.

[Her06] Serge Herzog. Estimating student retention and degree-completion time: Decision trees and neural networks vis-à-vis regression. *New directions for institutional research*, 2006(131):17–33, 2006.

[IPE] IPEDS. Retention Rates calculated by IPEDS. `https://nces.ed.gov/ipeds/trendgenerator/tganswer.aspx?sid=7&qid=32`.

[LAS+15] Himabindu Lakkaraju, Everaldo Aguiar, Carl Shan, David Miller, Nasir Bhanpuri, Rayid Ghani, and Kecia L Addison. A machine learning framework to identify students at risk of adverse academic outcomes. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1909–1918. ACM, 2015.

[Lau03] Linda K Lau. Institutional factors affecting student retention. *Education-Indianapolis then Chula Vista-*, 124(1):126–136, 2003.

[LBD+12] Eitel JM Lauría, Joshua D Baron, Mallika Devireddy, Venniraiselvi Sundararaju, and Sandeep M Jayaprakash. Mining academic data to improve college student retention: An open source perspective. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, pages 139–142. ACM, 2012.

[MDDM16] Farshid Marbouti, Heidi A Diefes-Dux, and Krishna Madhavan. Models for early prediction of at-risk students in a course using standards-based grading. *Computers & Education*, 103:1–15, 2016.

[Mit97] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[Pla13] Mark Plagge. Using artificial neural networks to predict first-year traditional students second year retention rates. In *Proceedings of the 51st ACM Southeast Conference*, page 17. ACM, 2013.

[TDMK14] Dech Thammasiri, Dursun Delen, Phayung Meesad, and Nihat Kasap. A critical assessment of imbalanced class distribution problem: The case of predicting freshmen student attrition. *Expert Systems with Applications*, 41(2):321–330, 2014.

[Tin99] Vincent Tinto. Taking retention seriously: Rethinking the first year of college. *NACADA journal*, 19(2):5–9, 1999.

[Tin06] Vincent Tinto. Research and practice of student retention: What next? *Journal of College Student Retention: Research, Theory & Practice*, 8(1):1–19, 2006.

Curriculum Vitae

Graduate College
University of Nevada, Las Vegas

Aditya Rajuladevi
Email: rajuladevi.aditya@gmail.com

Degrees:

Bachelor of Technology in Computer Science, 2014
Jawaharlal Nehru Technological University, Hyderabad, India

Master of Science in Computer Science, 2018
University of Nevada, Las Vegas, USA

Thesis Title: A Machine Learning Approach to Predict First-Year Student Retention Rates at University of Nevada, Las Vegas

Thesis Examination Committee:

Chairperson, Dr. Fatma Nasoz, Ph.D.
Committee Member, Dr. Justin Zhan, Ph.D.
Committee Member, Dr. Yoohwan Kim, Ph.D.
Graduate Faculty Representative, Dr. Magdalena Martinez, Ph.D.