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

by

Aditya Rajuladevi

Bachelor of Technology (B-Tech)
Jawahar Lal 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

Laxmi Gewali, Ph.D., Committee Member

Justin Zhan, 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 refer to the percentage of first-year students who return to the same institution for their sophomore year. The national average in the institutions at the U.S. for the year 2016 is at around 76% which indicates that most of the universities are performing poorly in terms of retaining the first-year students. 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. Hence it became important for universities to formulate strategies to identify students at risk 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 very challenging.

The retention rate at the University of Nevada, Las Vegas (UNLV) is close to 74% which indicate the need for specific intervention programme's 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 to whom the instructors can offer help. For this, we compared various classification algorithms of machine learning such as 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 UNLV's data warehouse that captured students' information such as pre-college academics, family background, financial situation and academic performance during their first-year at UNLV. The experimental results showed us 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 also like to extend my thanks to Dr. Laxmi Gewali, Dr. Justin Zhan, 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 Enterprises Application Services department at UNLV for their continuous support and valuable comments on this thesis. They answered my many questions about student enrollments and retention problems at UNLV and played a major role in helping me find the student data I was looking for from 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

Abstract iii

Acknowledgements iv

Table of Contents v

List of Tables viii

List of Figures ix

Chapter 1 Introduction 1

1.1 Objective . . . 3

1.2 Outline . . . 3

Chapter 2 Background and Preliminaries 4

2.1 Related Work . . . 4

2.2 Preliminaries . . . 6

2.2.1 Machine Learning Concepts . . . 6

2.2.2 Predictive Analytics . . . 6

2.2.3 Selected Models . . . 8

2.2.3.1 Logistic Regression Model . . . 8

2.2.3.2 Decision Trees . . . 9

2.2.3.3 Random Forest Classifiers . . . 11

2.2.3.4 Support Vector Machines . . . 12

2.2.4 Evaluation Methods . . . 13

2.2.4.1 Confusion Matrix . . . 14

2.2.4.2 Classification Accuracy . . . 15

2.2.4.3 Sensitivity . . . 16
2.2.4.4 Specificity . . . 16
2.2.4.5 Precision . . . 16
2.2.4.6 F_1 Score . . . 17
2.2.4.7 Area Under Curve (AUC) . . . 17
2.2.4.8 Cross Validation . . . 17

Chapter 3 Methodology **20**

3.1 Data Collection . . . 20
3.2 Data Preparation . . . 20
3.2.1 Data Description . . . 20
3.2.2 Feature Extraction . . . 22
3.2.2.1 Computation of SAT_ACT_Score Variable . . . 22
3.2.2.2 Computation of F2_Not_Retained Variable . . . 23
3.3 Data Preprocessing . . . 23
3.3.1 Handling Outliers in Data . . . 23
3.3.1.1 UnwHSGPA scores . . . 23
3.3.2 Handling Missing Values . . . 23
3.3.2.1 Imputing SAT_ACT_Score . . . 24
3.3.2.2 Imputing CoreHSGPA and UnwHSGPA . . . 25
3.3.2.3 Imputing F1_MidtermGPA . . . 26
3.3.2.4 Imputing F1_MathGradePass . . . 26
3.3.3 Data Transformations . . . 27
3.3.4 Feature Selection . . . 28
3.4 Exploratory Data Analysis . . . 29
3.4.1 Academic year Vs F2_Not_Retained . . . 29
3.4.2 IPEDS_Race Vs F2_Not_Retained . . . 30
3.4.3 Mom_Edu_Level Vs F2_Not_Retained . . . 31
3.4.4 Dad_Edu_Level Vs F2_Not_Retained . . . 33
3.4.5 College1 Vs F2_Not_Retained . . . 34

Chapter 4 Building Models **37**

4.1 Data Splitting . . . 37

4.2 Experiments on Models 38
4.2.1 Logistic Regression 38
4.2.2 Decision Trees 41
4.2.3 Random Forest 43
4.2.4 Support Vector Machines 45
4.2.5 Comparison of the models 48
4.2.6 Feature importance calculation based on selected models. 48
4.2.7 Probability estimates to risk scores 48

Chapter 5 Conclusion **49**
5.0.1 Future Work 49

Bibliography **50**

Curriculum Vitae **52**

List of Tables

2.1 Admissions data at a University 7

3.1 Description of data fields for first-year student data 21
3.2 Variables with missing values and their count 24
3.3 F2_Not_Retained outcome probabilities w.r.t availability of data points in SAT_ACT_Score 25
3.4 Tests based on type of input and output variables 28
3.5 Retention Rates of each IPEDS Race Category 31
3.6 Retention Rates of different categories of Mom_Edu_Level variable 32
3.7 Retention Rates of different categories of Dad_Edu_Level variable 34
3.8 Retention Rates of different categories of College1 variable 35

4.1 Count of students in each academic year 37
4.2 Computed metrics based on actual and predicted test data values using LR model . . . 39
4.3 Computed metrics based on actual and predicted unseen data values using LR model . 40
4.4 Computed metrics based on actual and predicted test data values using DTree model . 41
4.5 Computed metrics based on actual and predicted unseen data values using DTree model 43
4.6 Computed metrics based on actual and predicted test data values using RF model . . . 43
4.7 Computed metrics based on actual and predicted unseen data values using RF model . 45
4.8 Computed metrics based on actual and predicted test data values using SVM model . . 46
4.9 Computed metrics based on actual and predicted unseen data values using SVM model 47
4.10 Comparison of metrics from different models on the training data 48

List of Figures

1.1 Freshmen Retention Rates at UNLV 2

2.1 Machine Learning Approach 7
2.2 Classification tree of passenger survival in Titanic 10
2.3 Random forest trees 11
2.4 Example SVM Classifier 13
2.5 Example Confusion Matrix 14
2.6 Example ROC curve 18

3.1 Removing Outliers from UnwHSGPA 24
3.2 Boxplot of the SAT_ACT_Score vs F2_NotRetained 25
3.3 Analyzing difference between CoreHSGPA and UnwHSGPA 26
3.4 Distribution plot of the F1_MidtermGPA 27
3.5 Academic Year vs F2_NotRetained 29
3.6 Count plot of IPEDS_Race variable 30
3.7 Count plot of Mom_Edu_Level variable 32
3.8 Count plot of Dad_Edu_Level variable 33
3.9 Count plot of College1 variable 35

4.1 Confusion Matrix of Logistic Regression on test data 39
4.2 ROC curve for Logistic Regression on test data 40
4.3 Confusion Matrix of Logistic Regression on unseen data viz 2016 academic year data . . 41
4.4 Confusion Matrix of Decision Tree on test data 42
4.5 ROC curve for Decision Tree on test data 42
4.6 Confusion Matrix of Decision Tree on unseen data viz 2016 academic year data 43
4.7 Confusion Matrix of Random Forest on test data 44

4.8 ROC curve for Random Forest on test data 44
4.9 Confusion Matrix of Random Forest on unseen data viz 2016 academic year data 45
4.10 Confusion Matrix of Support Vector Machine model on test data 46
4.11 ROC curve for Decision Tree on test data 46
4.12 Confusion Matrix of SVM model on unseen data viz 2016 academic year data 47

Chapter 1

Introduction

The first-year or freshmen retention rate refers to the number of freshmen in a college or university who return for their sophomore year. Many universities are facing huge problems with low or decreasing first-year student retention rates. Low retention rates are 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 the 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 development 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 GPA's, absence rates, previous grades, SAT or ACT scores from enrollment data to generate rules that can be used to identify students at risk [BS16]. Although such rule-based systems served as good indicators of identifying at-risk students for some years, they had some downsides such as fewer accuracies, static, expensive to generate and maintain and most importantly they lacked a validation mechanism to verify the predictions. Alternatively, recent research has indicated the potential value of machine learning algorithms such as Logistic

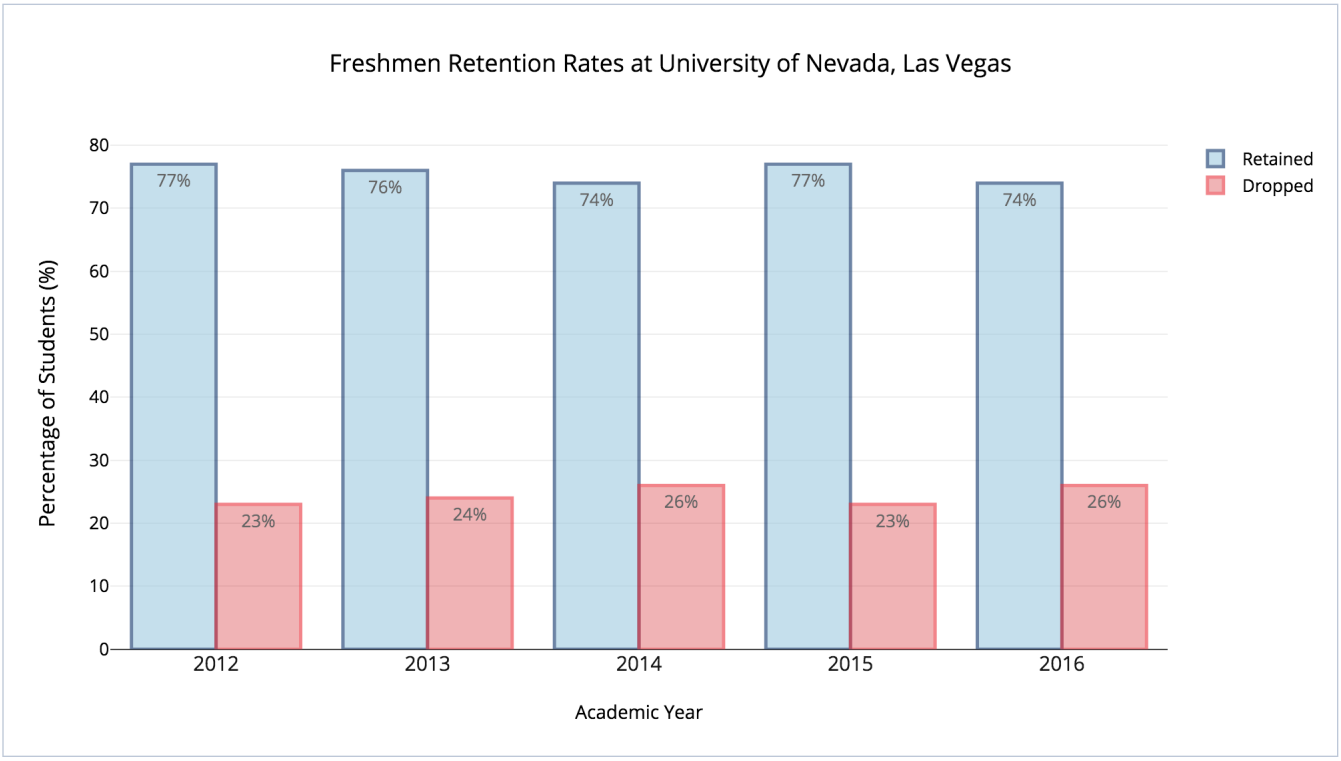


Figure 1.1: Freshmen Retention Rates at UNLV

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 implementation complexity and the availability of data. 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 2012 to 74% in 2014 and then increased to 77% in 2015, which later on fell to 74% in 2016 Figure 1.1. Such an unstable pattern of freshmen retention rates has drawn a lot of attention by the educators and administration at UNLV. Hence, a predictive approach to identifying 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 at-risk students. In this thesis, important machine learning algorithms such as Logistic Regression, Decision trees, Random Tree Classifiers and SVM's will be trained on real-time student data obtained from UNLV's enrollment census. The trained models will then be used to predict at-risk students from a test dataset which the model has not seen earlier. The models are evaluated and compared using metrics such as precision, recall, and area under the curve to determine which model provides the best results. The results of the analysis such as the evaluation metrics will be converted to risk scores which can be easily understood by the educators and administrators at UNLV. Another contribution of this thesis is to rank students based on risk scores which will be very helpful in an 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 was described.

In Chapter 2, we will discuss the existing research on improving first-year retention rates and the background information required to understand the proposed predictive approach using machine learning. It will also 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 datasets used for these methods will also be described.

In Chapter 4, we will present the experimental results. The characteristics of datasets used for these methods will also be described.

In Chapter 5, we will summarize the proposed methods and their results along with the possible extension of this thesis.

Chapter 2

Background and Preliminaries

2.1 Related Work

The prediction of first-year student retention rates and identification of students at risk of not being retained has been a well-researched problem in the area of higher education sector 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 Ernest Pascarella, Patrick Terenzini, and Alexander Astin, which focused more on the external factors such as the institution's administration and its policies when determining the reasons for student retention [A$^+$12]. Tinto in his 2006 study [Tin06] has stated that there has been a huge increase in the number of businesses and organizations to analyze and help institutions with the student retention problem. Later, in the same study, he revealed that there was only little change in the retention rates even with some huge businesses helping the universities. He also described the importance of external factors such as student-faculty relationships, extracurricular program, and orientation programs for first years. Moreover, he incorporated the role of academic factors into his model to make it more suitable to the college structure [Tin06]. Astin in his Input-Environment model [A$^+$12], suggests that researchers should consider pre-college factors such as gender, race/ethnicity, family background, high school GPA as important for student retention.

In addition to understanding the factors responsible for student dropout, the researchers were interested in identifying students at 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 student retention rates [LAS+15, MDDM16, AC17]. The results from these models showed that the learning algorithms were in fact better than many existing rule based models in 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, Jay Bainbridge, James Melitski, Anne Zahradnik, Eitel J. M. Laura, Sandeep Jayaprakash and Josh Baron used fall 2010 undergraduate students 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[BMZ+15]. The results showed that logistic regression and SVM trees provided higher classification accuracies compared to the decision trees to predict students at risk.

Serge Herzog a researcher from University of Nevada, Reno(UNR) campus has done some extensive research on student retention and graduate prediction. He used Decision trees and Neural networks to predict student retention of data from UNR[Her06]. FarshidMarbouti, Heidi A.Diefes-Dux, KrishnaMadhavan [MDDM16] have compared seven different prediction models for identifying at-risk students using in-semester performance factors (i.e., grades) and based on standards-based grading.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 being the lack of 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 the efficient allocation of the resources [LAS+15]. Though the above-mentioned framework was giving good results, it was restricted to school level data and thus cannot yield accurate results in the university level as both are very different environments with different set of factors. Hence, in this thesis we try to include such an analysis into college level studies of UNLV and compare our results to the existing approaches.

2.2 Preliminaries

2.2.1 Machine Learning Concepts

Tom 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 experience, which it uses 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 perform such a learning process in identifying patterns from complex data which can be very difficult for a human to understand.

2.2.2 Predictive Analytics

Predictive analytics primarily deals with extracting patterns using machine learning models from existing datasets. the extracted patterns are used to predict future events and behaviors in previously unseen data. Predictive analytics is being used in 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 properties 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.

An illustrative example is shown in Table 2.1 which shows student data captured during admissions at a university. The aim is to predict if the student will be accepted or not by looking at the

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

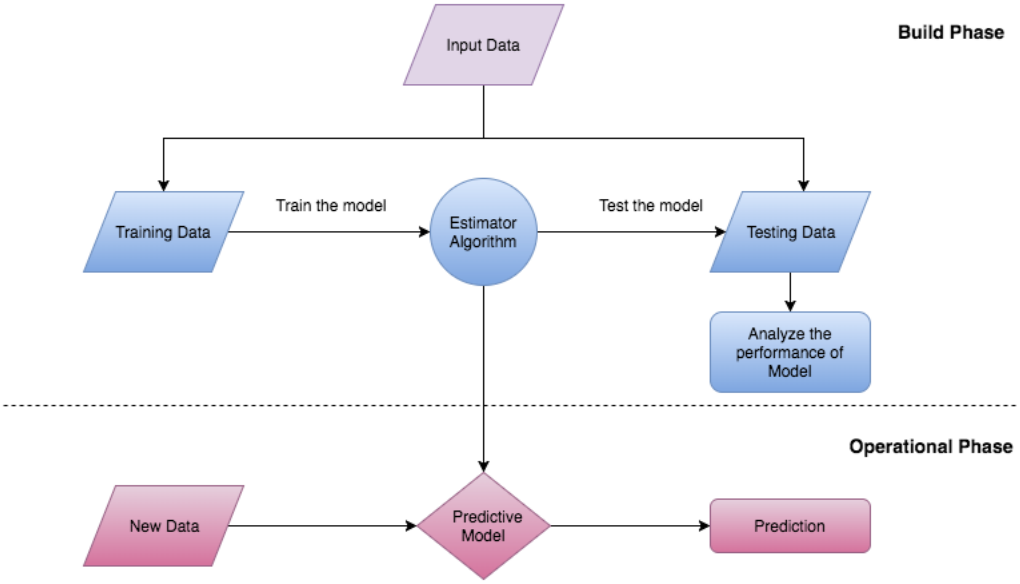


Figure 2.1: Machine Learning Approach

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 prediction model based on the relations between the input independent variables and the output dependent variable. The learned model is then able to predict the output variable by taking 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 a data can be divided into 2 phases as shown in figure 2.1. Build phase deals with the process of creating a prediction model and

testing its performance. The process of creating the prediction model is known as training and the data used is known as training data. To test the effectiveness of the created model, it is tested on another set of previously unseen data known as testing data. We use two different data sets to check the generalization capacity of the model in predicting 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 other unseen data. A common approach to handle the problem of overfitting is to divide the input data into training and testing sets. The model is then evaluated for performance using test data. The evaluated model is used in the operational phase to perform predictions given a new data. The metrics used to evaluate a model are explained in detail in coming chapters.

2.2.3 Selected Models

Based on the way machine learning algorithms "learn" patterns from data, they can be classified into two groups: supervised and unsupervised learning. In supervised learning, we train the machine learning model using data whose possible outcomes are already known, whereas unsupervised learning is the training of machine learning models using information that is neither classified nor labeled and allowing the algorithm to learn by itself.

In our case, since we have data about students along with the labels of the outcome variable, we will do supervised learning. Additionally, the outcome variable in our analysis is a categorial variable, indicating if the student was retained in their second year or not and thus our problem is of classification. In this section we discuss various classification models that were used in this thesis to analyze the student data from UNLV. The models selected for comparison were Logistic Regression, Decision Trees, Random Forest, and SVM's.

2.2.3.1 Logistic Regression Model

Logistic Regression is a supervised machine learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). It is a variation of linear regression, where 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 an output value (y), whereas logistic regression calculates the probability of getting the outcome dependent variable (y). The general equation for

this learning approach with one independent variable and a dependent output variable 'y' is as shown in the Equation2.1

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

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 the Equation2.2. 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 . Hence our logistic regression problem then represents the probability of the student passing an exam given the number of hours he studied before the exam. A rule of thumb in logistic regression is if the probability is > 0.5 then the decision is true. Hence you end up with a model that finds the probability of a student passing the exam as shown in Equation2.3

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

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 the tree representation to solve the problem, 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.

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 an

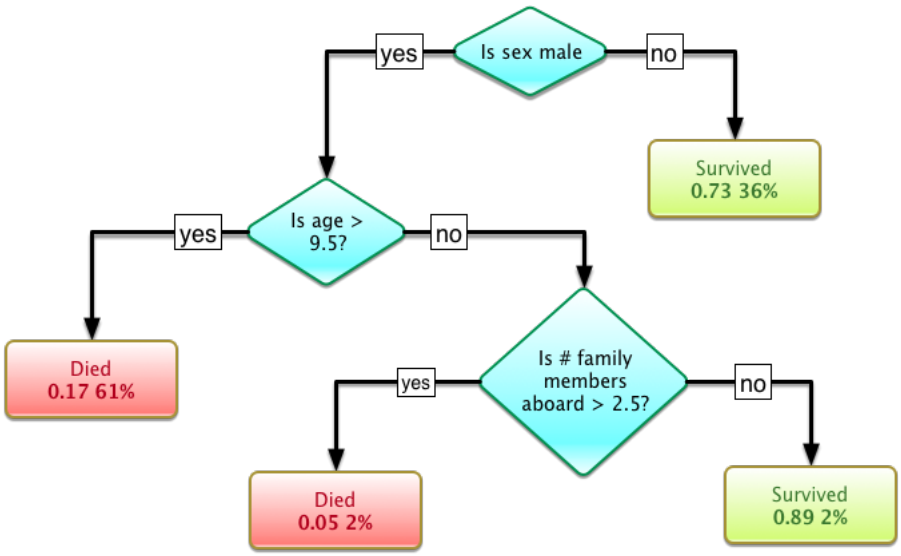


Figure 2.2: Classification tree of passenger survival in Titanic

attribute). The process of splitting based on decisions of different internal node attributes continues until a subset at a node has all the same value 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 so, in practice, different metrics are used for evaluating splits. Two main metrics used to evaluate the splits are :

Gini impurity : 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 yes or no, given that the example reached that node. The node with highest information gain value yields the best split.

For illustration purpose, consider the classification tree showing survival of passengers on the Titanic ship when it sinked in Figure 2.2. 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 goodness evaluation metric. From the tree we can make the following inferences:

1. If the passenger is a male and has an age greater than 9.5 then he did not survive the titanic sink.

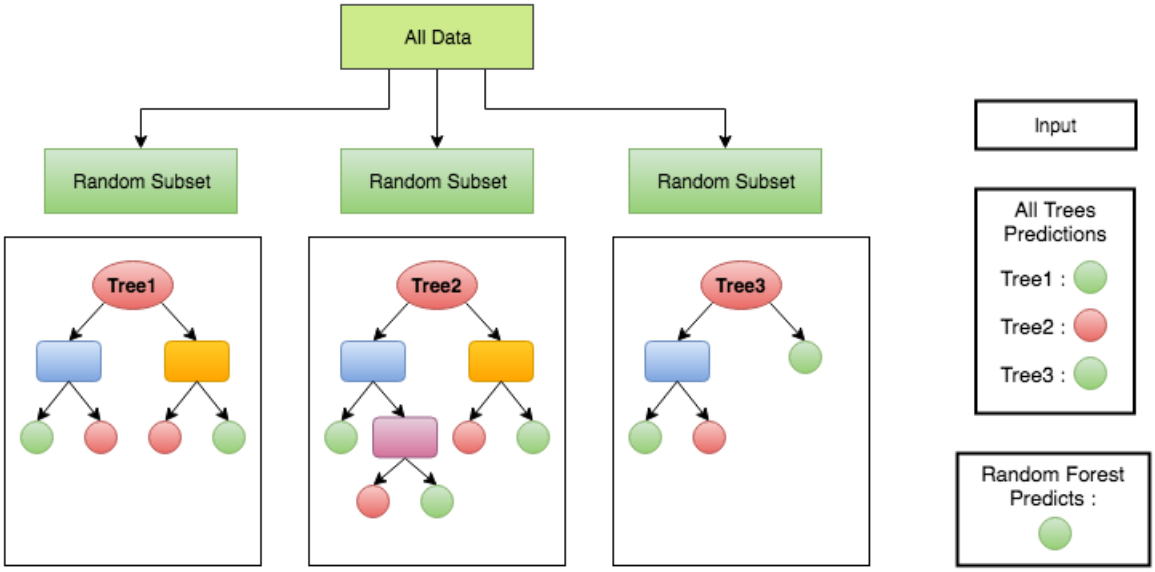


Figure 2.3: Random forest trees

2. All the female passengers had a survival probability of 0.73.

3. If the passenger was male, had an age less than 9.5 and had lesser than 2.5 family members aboard 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 method known as an Ensemble method 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 is the case of overfitting. This happens due to the greedy approach employed by decision tree algorithm along with information gain or gini index metrics, 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 as 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 are the predictions.

The 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 as shown in the figure.

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 mostly useful for classification problems. In this algorithm, each data item 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 very well. 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 that separates all of the input points (in this case a line). After the line that separates the input data well 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 its 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 the construction of the classifier and thus are called the support vectors. The hyperplane is learned from the training data using an optimization procedure that maximizes the margin.

In practice, real world data is messy and cannot be perfectly separated with a hyperplane. In

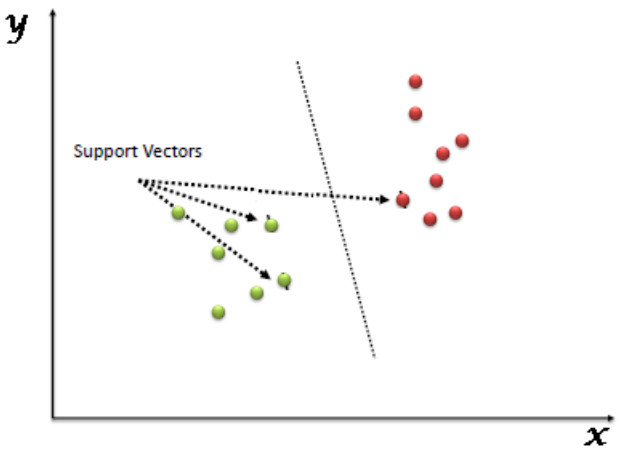


Figure 2.4: Example SVM Classifier

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 chose the one that is best for the data.

In SVM, it is easy to have a linear hyper-plane between the 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 takes 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 subsection, we will explain the various evaluation methods that are used in this thesis to assess and compare the performance of different 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 unseen data and ensure that it is not just memorizing patterns in the data. To make a better analysis of the generalizing power of the model, the data is generally split into training and testing sets, where training data is used to create the model while testing set to evaluate the model. To do this, we need to define the performance measure strategies we will use to evaluate our models. Having a performance measure defined for the machine learning 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 in this thesis, we will concentrate more on the metrics for a classification problem.

2.2.4.1 Confusion Matrix

Confusion matrix is a representation of the predictions made by the classification model. It is a very important and useful metric in classification problems as it can not only be used to represent the models performance by itself but can also 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 top left 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 top right 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 bottom left 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 bottom right 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', and '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 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 true 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 false 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 true. 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 true 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.

$$F_1 = \frac{True\,Positives}{True\,Positives\,+False\,Positives} \tag{2.8}$$

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 Curve (ROC) is the most commonly used way 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].

The 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.

2.2.4.8 Cross Validation

So far, we discussed the calculation of various metrics based on the predictions obtained from a single test data which was split from the original data. The approach of dividing the input data into training and testing data is known as holdout method. The typical split rate is about 70% of input data into training and remaining 30% into testing data. These splits are generally created by random sampling. One problem with the holdout strategy to evaluate the models performance

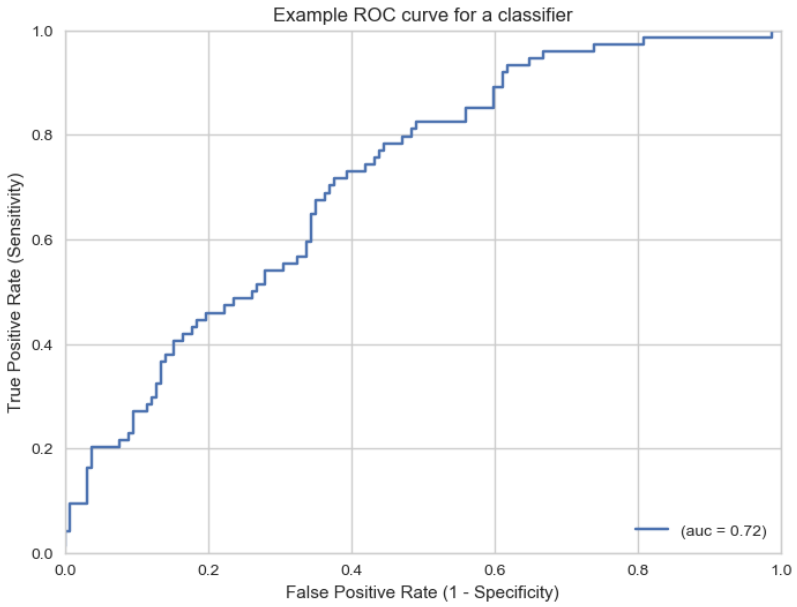


Figure 2.6: Example ROC curve

is that the random splits might not be fair and lets say a specific combination of input data might always end up in testing 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 each time, one of the k subsets is used as the test set/ 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 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 bias as we are using most of the data for fitting, and also significantly reduces variance as most of the data is also being used in validation set. Interchanging the training and test sets also adds to the effectiveness of this method. The typical value of K suggested by many researchers is 10.

Stratified K-Fold cross validation is actually a variation of the K-fold cross validation which

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 discussed about 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. Hence, in these cases F_1 score and AUC can be used as a single value comparators of the models 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 regarding student enrollment and performance from the UNLV Data Warehouse and reporting the census to Integrated Postsecondary Education Data System (IPEDS). The data used in this thesis were obtained from the census data submitted to IPEDS every year. The following subsections will describe more information about the data.

3.2 Data Preparation

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 consisted of a total of 17,708 freshmen student records captured cohort-wise from the academic year 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 finally if the student was retained in the sophomore year. On the whole, a total of 41 attributes were captured for each student. Table 3.1 describes the variables and their types. The following subsections describe in detail about the dataset.

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

No.	Feature	Type	Description
1	Cohort_Year	Multi Nominal	First Fall Academic year of the student
2	SummerAdmit	Binary Nominal	If Student was admitted in prior summer
3	AltAdmit	Binary Nominal	If Student was Transferred or admitted differently
4	F1_EnrollCredits	Numerical	Count of semester credits enrolled for admit term
5	F1_MidtermGPA	Numerical	F1 mid-term GPA
6	F1_GPA	Numerical	Term GPA for first fall term (admit 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 first fall (for GPA)
10	F1_Complete	Binary Nominal	Completed first fall term (final census)
11	S1_Retain	Binary Nominal	Enrolled in 1st spring (S1) - yes or no
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 kid 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 admit 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	ACT scores scaled to SAT scores
28	ACT_SAT	Binary Nominal	If the student has taken ACT or SAT

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	1st fall Millennium Scholarship recipient
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 data collection techniques at UNLV. The raw data was used to generate the following important features.

3.2.2.1 Computation of SAT_ACT_Score Variable

The admissions office at UNLV requires the students to report scores of standardized exams such as ACT and SAT during their admission process. From the raw dataset, we noticed that the ACT and SAT scores were missing for many students. Especially 56% of students did not have ACT scores while only 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 variable.

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 freshmen year. In the raw dataset the flag F2_Retain captures the aspect of student being retained and hence can serve as the outcome variable. We then 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 the 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 the 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 tiny clean dataset. This technique is performed before the execution of any predictive Analysis.

3.3.1 Handling Outliers in Data

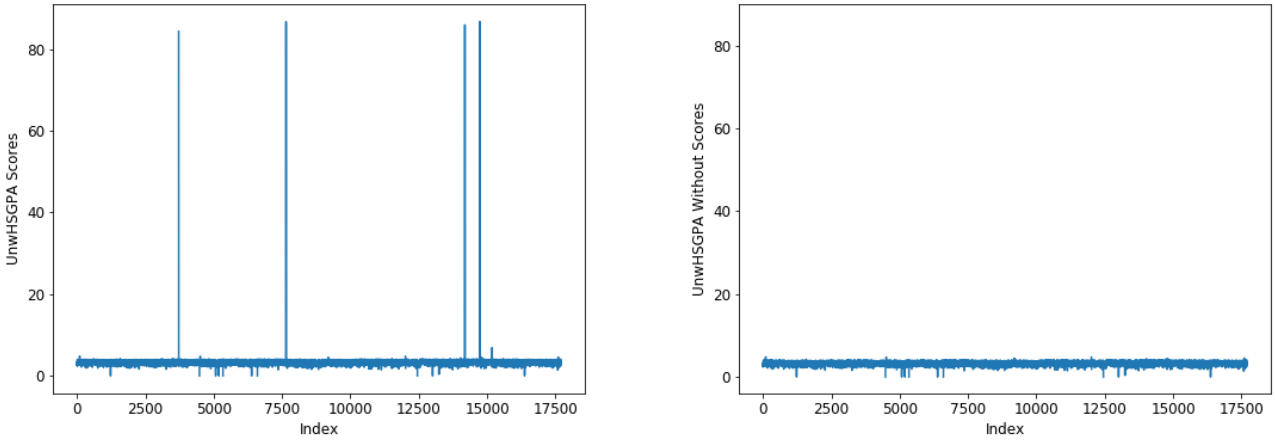
Generally the process of collecting data from different sources can introduce outliers into the dataset. This may be due to a faulty data extraction program or a human error. In predictive data analysis outliers can have significant effect on the predictive power of the models as they can introduce huge noise into the model. Hence it is almost always essential to check for outliers in the dataset before performing any analysis. Outliers were identified in our dataset. Data plots with index on x-axis and values on y-axis was used to demonstrate the presence of outlier in a feature.

3.3.1.1 UnwHSGPA scores

There were a few outliers identified for variable UnwHSGPA, as shown in figure 3.1a. The outliers of the UnwHSGPA were replaced by mean value of the said dataset excluding all the outliers. Data plot of the variable UnwHSGPA after removing outliers is shown in figure 3.1b

3.3.2 Handling Missing Values

In this subsection, we will discuss how we handled missing values in the dataset. The table 3.2 shows variables along with the count of missing values. Machine learning deals with the usage of



(a) UnwHSGPA with Outliers

(b) UnwHSGPA without Outliers

Figure 3.1: Removing Outliers from UnwHSGPA

Table 3.2: Variables with missing values and their count

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

mathematical models on the data and hence 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 given values to substitute in the missing field.

3.3.2.1 Imputing SAT_ACT_Score

The SAT_ACT_Score variable had 830 missing values. Firstly, to understand the impact of missing values, we calculated the probabilities of the outcomes of the F2_Not_Retained variable with respect to the availability of data points in SAT_ACT_Score variable. In one case we ignored the records that had missing values and in other case we included all the records as shown in Table 3.3. By looking at the table, it can be inferred that presence of missing values in SAT_ACT_Score variable had very little impact on the percentage of students not being retained. A total of 24.1% students

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%

were not retained when a value was present in the variable and 25.6% were not retained when the variable had a missing value.

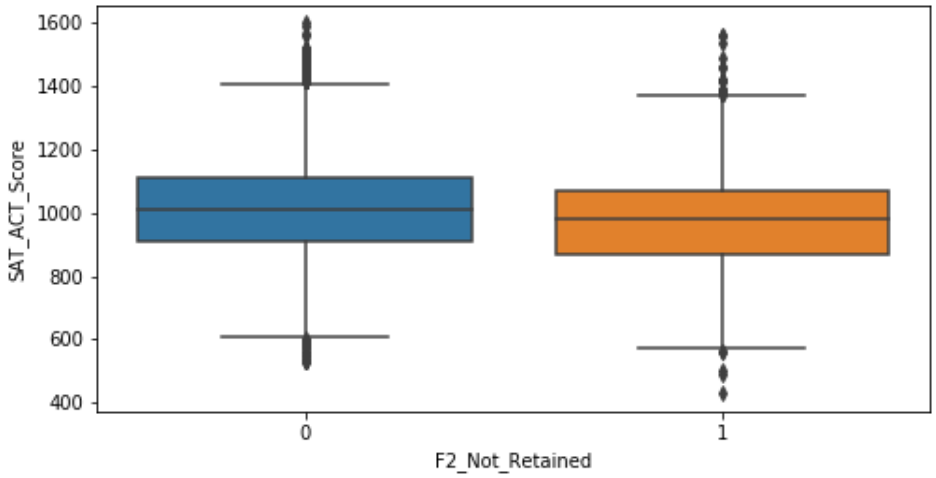


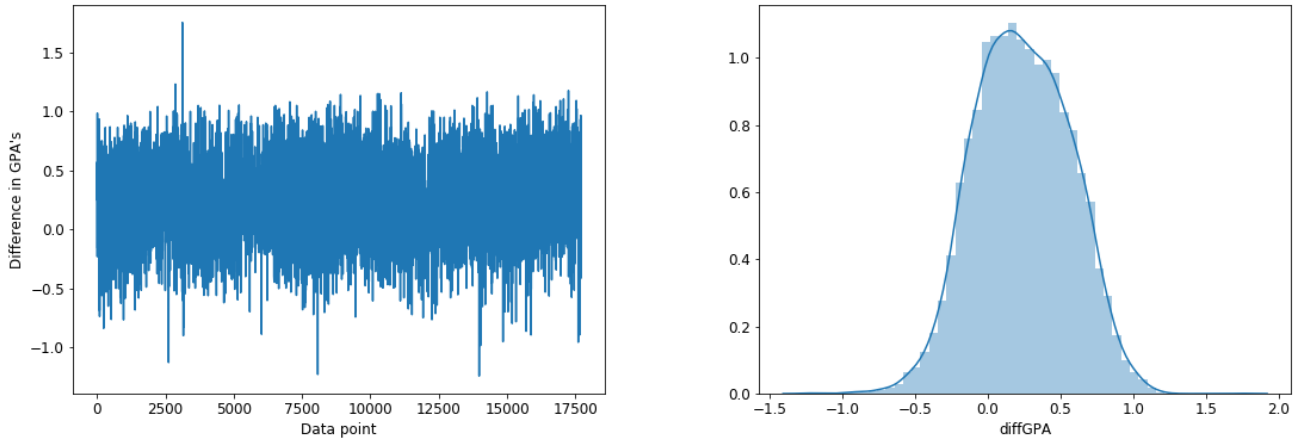
Figure 3.2: Boxplot of the SAT_ACT_Score vs F2_NotRetained

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 variable were replaced by the mean value.

3.3.2.2 Imputing CoreHSGPA and UnwHSGPA

The CoreHSGPA variable had 3463 missing values, whereas the UnwHSGPA had 1943 missing values. Before imputing, we tried to analyze the relation between both the variables (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 variables had very close relation. Similar pattern was revealed from the distribution plot of the variable differences as shown in figure(3.3b),

which revealed that the differences followed a gaussian distribution with a mean of 0.23. Based on the above facts, we replaced the missing values with the available variable scores. The remaining missing values were replaced by the mean value of the respective variable.



(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 variable 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 also 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 variable with its mean as it would not shift the centrality of the variable.

3.3.2.4 Imputing F1_MathGradePass

F1_MathGradePass is a categorical variable 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 he got a failing grade. It had a total of 4989 missing values in the variable. Such a large number of missing values for this variable is attributed to the fact that a student who did not take any math course will not have a value in this variable. Hence the missing value was replaced with a 'N'.

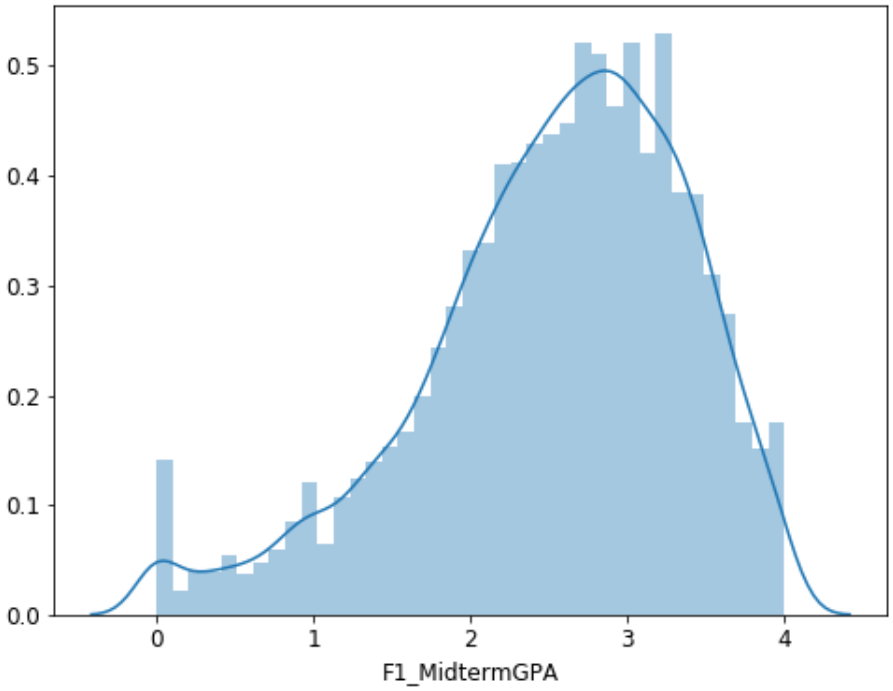


Figure 3.4: Distribution plot of the F1_MidtermGPA

3.3.3 Data Transformations

Now that we have a cleaned 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 variables 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 which a categorical variable is converted into a binary vector, where each possible value of the categorical variable is assigned to a dummy variable with a default value of '0' and the dummy variable which was the value of the categorical variable will have the value '1'. In simple terms, applying one-hot encoding to a categorical variable results in an one-hot vector, where only one element is non-zero, or hot. There were a total of 23 categorical variables in our raw dataset which were one-hot encoded to yield a total of 85 variables. We noticed that applying transformations on our raw data resulted in a huge increase in the number of variables available for analysis. Having more number of variables can result in poor performance

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

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

of the model, 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 raw 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 you are going to use on your data.

One of the feature selection approaches deals with the idea of identifying the relationship of features with the output variable to decide their importances. To find such relationships, we need to identify the data type of features and the output variable. The table 3.4 shows the types of tests used based on input feature types and output response 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 variables. The higher the magnitude of the correlation coefficient, the greater is the variables influence on predicting 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, also written as $\tilde{\chi}^2$ 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 response variable is a categorical with categories {'Y' , 'N'} and input features

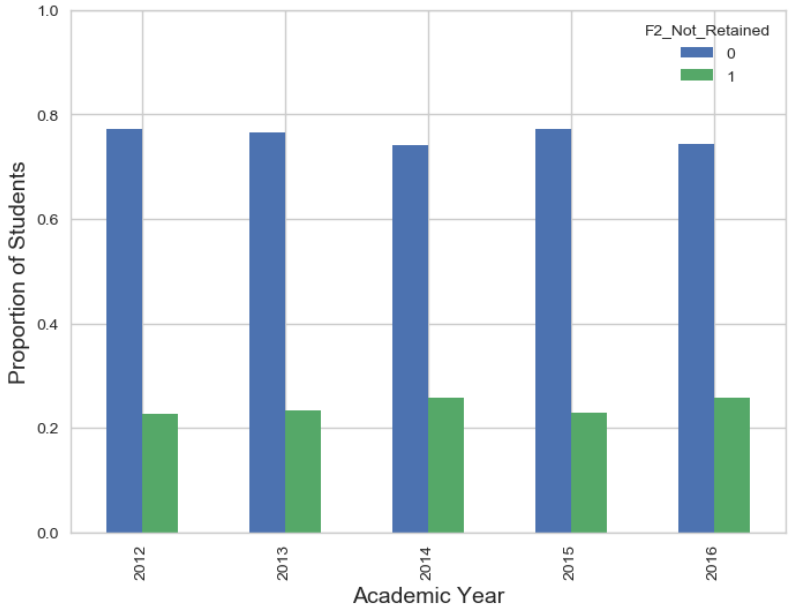


Figure 3.5: Academic Year vs F2_NotRetained

are both categorical and quantitative. Hence an appropriate choice of test to find relationships between the input features and output response is to use the Chi-square test. In machine learning the process of using such statistical tests on the dataset is known as Univariate selection. Hence 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 data sets and summarize their main characteristics, often with visual methods, without making any assumptions about its contents. It is an important step to take 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 Academic year Vs F2_Not_Retained

To understand the distribution of student dropouts at the end of each year, we plotted the proportions of the outcome variable F2_Not_Retained for each year as shown in Figure 3.5. Each academic year represents the cohort of students who were in their F1 semester in that specific 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 section, we tried to analyze 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 x-axis and the count of students belonging to that particular race on y-axis as shown in figure 3.6. We observed that the student population of different races was very different at UNLV with higher populations belonging to Asian, Hispanic and White races.

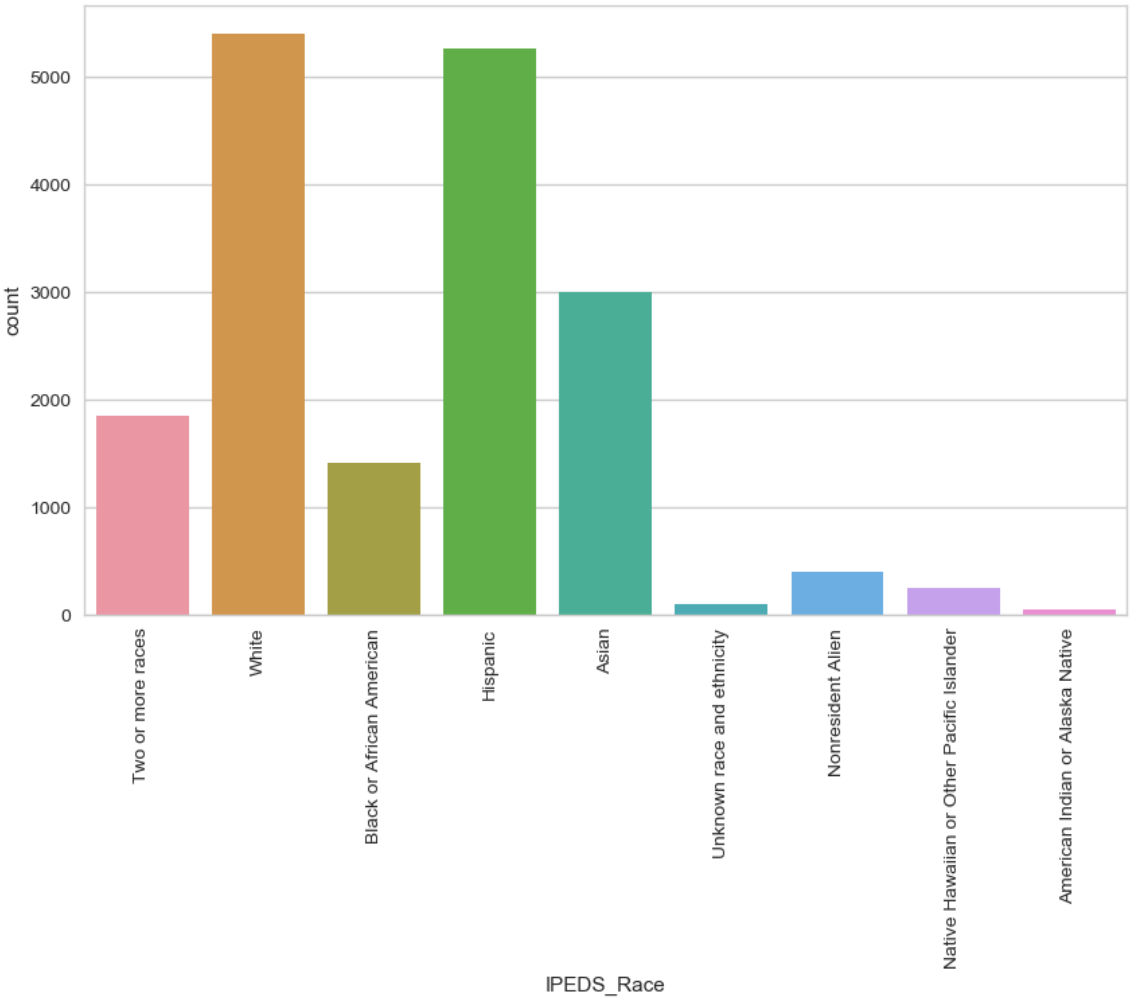


Figure 3.6: Count plot of IPEDS_Race variable

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%

Although fig 3.6 gives us a good idea about the type of students at UNLV, it is not sufficient to help us understand the retention patterns among the 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 table, we conclude that the Asian race category had the highest student retention followed by Hispanic and White categories. On the whole we observe that the student's race is useful in predicting if he/she will be retained after the first year.

3.4.3 Mom_Edu_Level Vs F2_Not_Retained

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

To get a better understanding of the effect of Mom's Education on students retention, we calculated the retention rates of students from each category of the variable and tabulated them in table 3.6. Considering the student distributions in each category and the retention rates of the variable from table, we notice some expected patterns such as more students were retained if

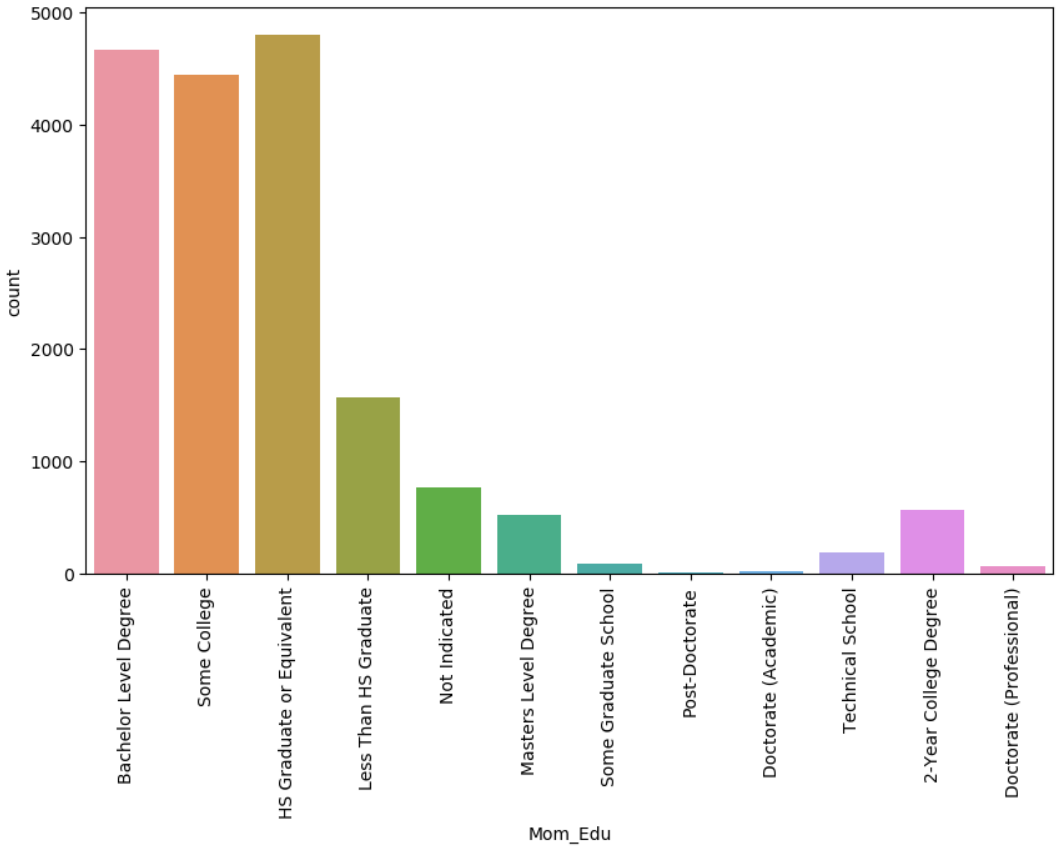


Figure 3.7: Count plot of Mom_Edu_Level variable

Table 3.6: Retention Rates of different categories of Mom_Edu_Level variable

Mom's Education 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%

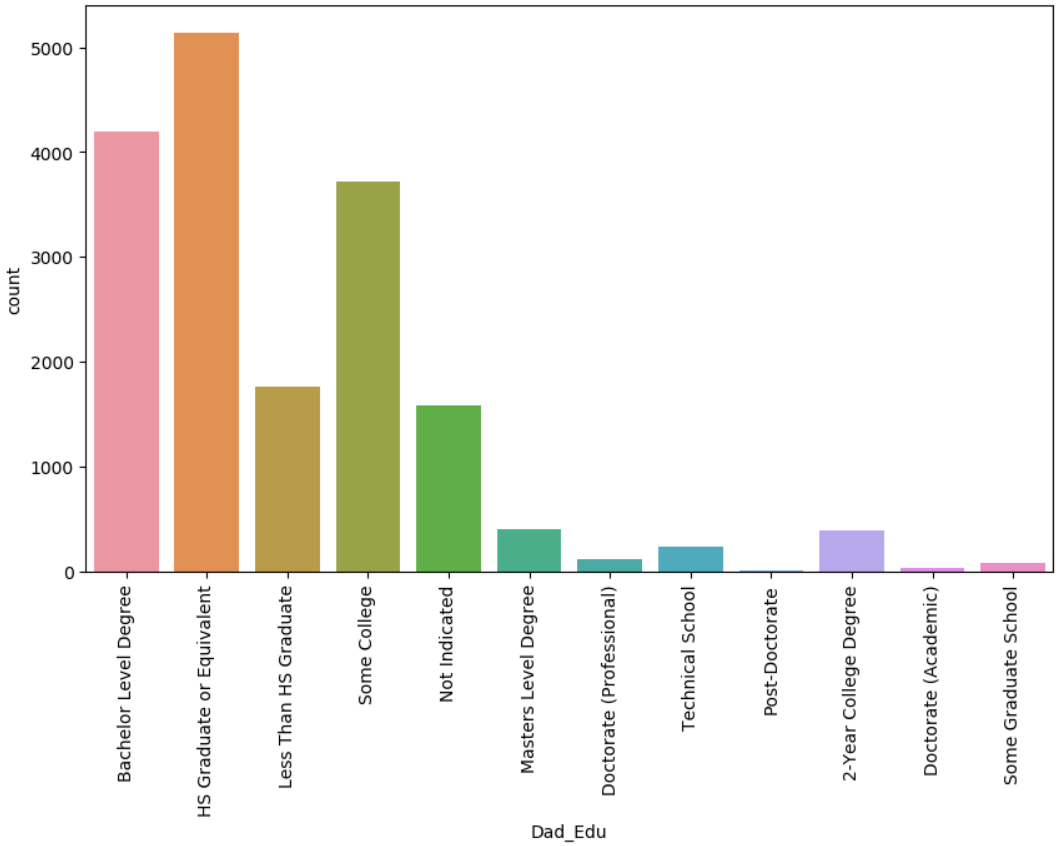


Figure 3.8: Count plot of Dad_Edu_Level variable

their Mom had an Education level higher than bachelor's. But strangely we also noticed that the retention rate of students with their Mom's education level as Less than HS Graduate was also good at UNLV. This gives us a hint of the variability of patterns inherent in the students data from UNLV.

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 section, we tried to analyze the patterns observed by different categories of the Dad_Edu_Level variable with respect to student retention. For this, we plotted the different categories of Dad_Education_Level variable on x-axis and the count of students belonging to that category on y-axis as shown in figure 3.8. We observed that the education level of the Dad's of students at UNLV was highly distributed. It especially gives us the information that most of the student's Dad's had a minimum education of High School or more.

To get a better understanding of the effect of Dad's Education on students retention, we calcu-

Table 3.7: Retention Rates of different categories of Dad_Edu_Level variable

Dad's Education 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%

lated the retention rates of students from each category of the variable and tabulated them in table 3.7. Considering the student distributions in each category and the retention rates of the variable from table, we notice many expected patterns such as more students were retained if their Dad had an Education level higher than bachelors'. But strangely, we also noticed that the retention rate of students with their Dad's education level as Less than HS Graduate was also good at UNLV. This gives us a hint of the variability of patterns inherent in the students data from UNLV.

3.4.5 College1 Vs F2_Not_Retained

The variable College1 represents the name of the college of the student during his first year. In this section, we tried to analyze the patterns observed by different categories of the student's college in his F1 term with respect to student retention. For this, we plotted the different categories of College1 variable on x-axis and the count of students belonging to that category on 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 followed by 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 notice few patterns such as more students were retained in College of Sciences and College of Engineering which had

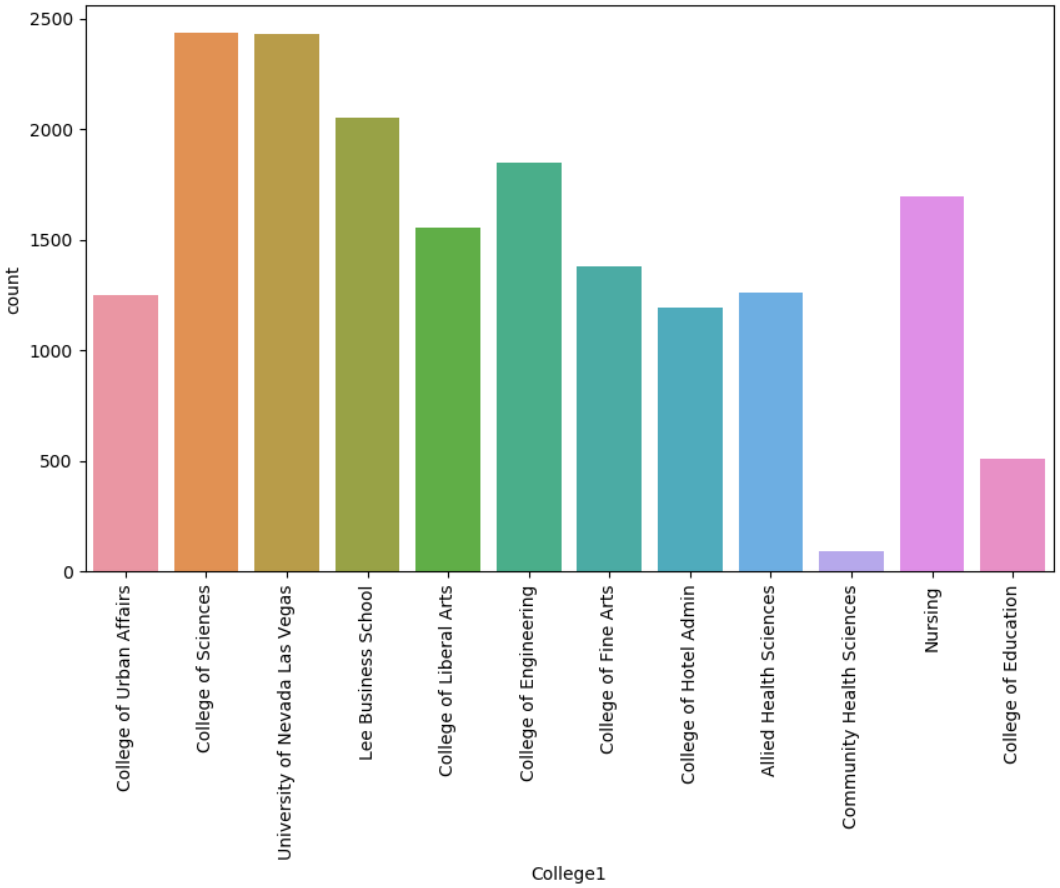


Figure 3.9: Count plot of College1 variable

Table 3.8: Retention Rates of different categories of College1 variable

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%

higher student counts. Even though other colleges had lesser student populations, their student retention rates were similar and averaged to about 75% . This gives us a hint of the variability of patterns inherent in the students data from UNLV.

Chapter 4

Building Models

4.1 Data Splitting

Once the dataset was cleaned and feature selection was performed on it, there were 51 variables excluding the F2_Not_Retained variable. Hence, we have a total of 51 independent variables which act as predictors and are inputted into the model to predict the value of F2_Not_Retained the output variable.

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

Table 4.1: Count of students in each academic year

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

From the entire dataset, the student records from the academic years 2012, 2013 ,2014 and 2015 were combined to form the train_test dataset. The remaining student records from the year 2016 were used to form test_unseen dataset. The idea was to use the 4 years of data to train, test and evaluate the model. Once the model is trained and tested, it will be used to make predictions on student records from the test_unseen dataset. This way, by not using any data from the 2016

academic year in training and testing processes, the model will 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 a total of 13,192 student records in the train_test dataset and 3756 student records in test_unseen dataset.

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

4.2 Experiments on Models

Once the training, testing and unseen data were created from the original dataset, each of the selected models were trained using the training data and evaluated on the test data. The models which were trained were Logistic Regression, Decision Tree, Random Forest and Support Vector Machines.

Once each model was trained, we used it to make predictions and generate a confusion matrix on the test data. The confusion matrix was used to calculate the classification accuracy, sensitivity, specificity, precision, F_1 scores and 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 get from the predictions on test data, it means that the model is actually learning from the training data. Furthermore, we use the trained model to make predictions on unseen data and generate confusion matrix to compute the performance metrics which give us a better idea of the generalizing power of the model on unseen data.

All the above-mentioned performance metrics were used to compare the models to find the one that suits best for the UNLV students 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 perform predictions on the test data. The confusion matrix generated from the predictions of the test data is as shown in Figure 4.1

The logistic regression model built on the training data had a classification accuracy of 0.843. The table 4.2 shows the other metrics calculated from the confusion matrix in Figure 4.1.

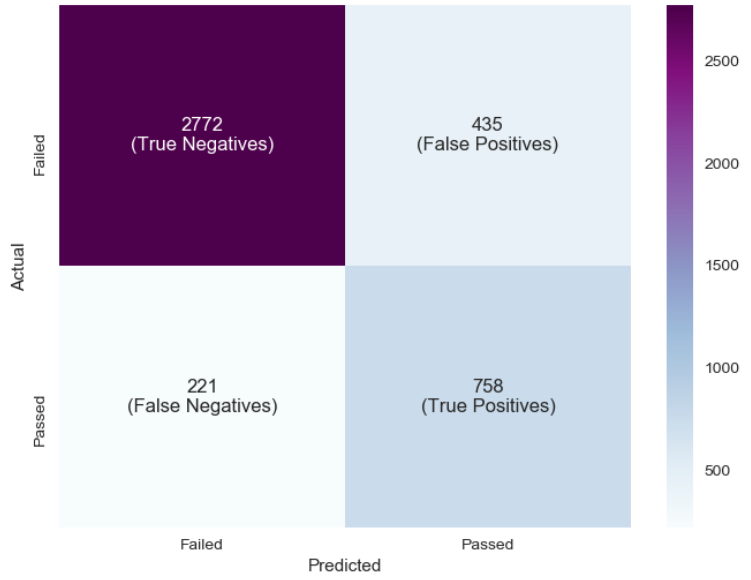


Figure 4.1: Confusion Matrix of Logistic Regression on test data

Table 4.2: Computed metrics based on actual and predicted test 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

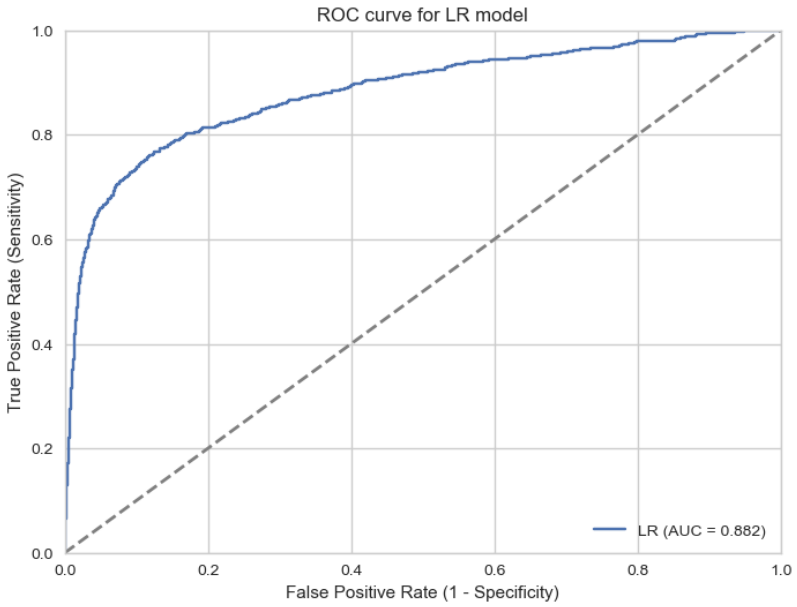


Figure 4.2: ROC curve for Logistic Regression on test data

ROC curve for the LR model on test data was plotted and is as shown in the 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. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the unseen data well.

The trained LR model was used to perform predictions on the unseen dataset that was created from the original dataset. The predictions on the unseen data define the generalizing power of the LR model at the individual student level as well as the academic year level. The confusion matrix generated from the predictions on unseen 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 the unseen data. The table 4.3 shows the other metrics calculated from the confusion matrix in Figure 4.3.

Table 4.3: Computed metrics based on actual and predicted unseen 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

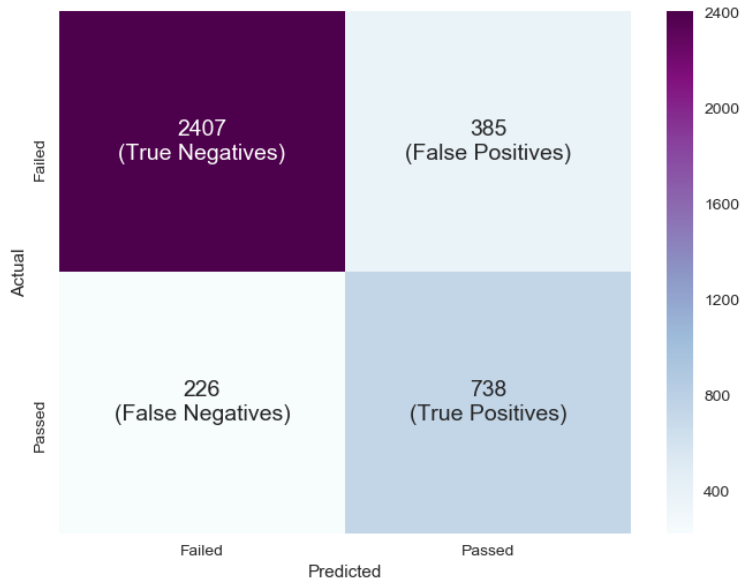


Figure 4.3: Confusion Matrix of Logistic Regression on unseen data viz 2016 academic year data

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 then used to perform predictions on the test data. The confusion matrix generated from the predictions of the test data is as shown in Figure 4.4

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

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

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
DTree model	0.871	0.699	0.924	0738	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. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the unseen data well.

The trained DTree model was used to perform predictions on the unseen dataset that was created from the original dataset. The predictions on the unseen data define the generalizing power of the Dtree model at the individual student level as well as the academic year level. The

Figure 4.4: Confusion Matrix of Decision Tree on test data

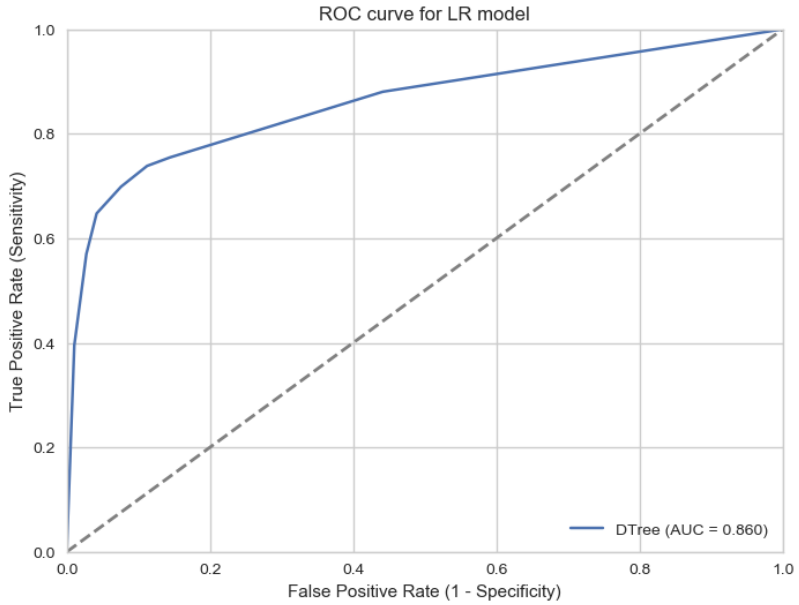


Figure 4.5: ROC curve for Decision Tree on test data

confusion matrix generated from the predictions on unseen data is shown in the figure 4.6.

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

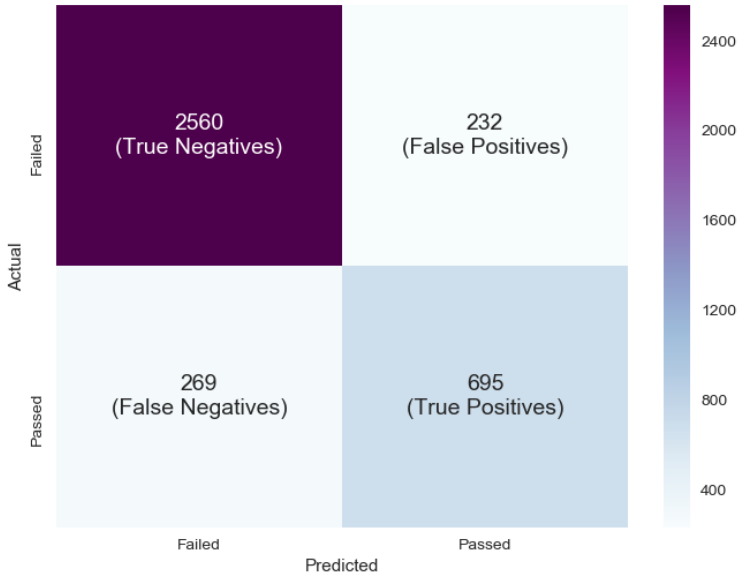


Figure 4.6: Confusion Matrix of Decision Tree on unseen data viz 2016 academic year data

Table 4.5: Computed metrics based on actual and predicted unseen 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

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

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

ROC curve for the RF model on test 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 validated AUC score of 0.876. This clearly confirms that the model is actually learning from the training

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

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
RF model	0.861	0.723	0.903	695	0.71	0.876

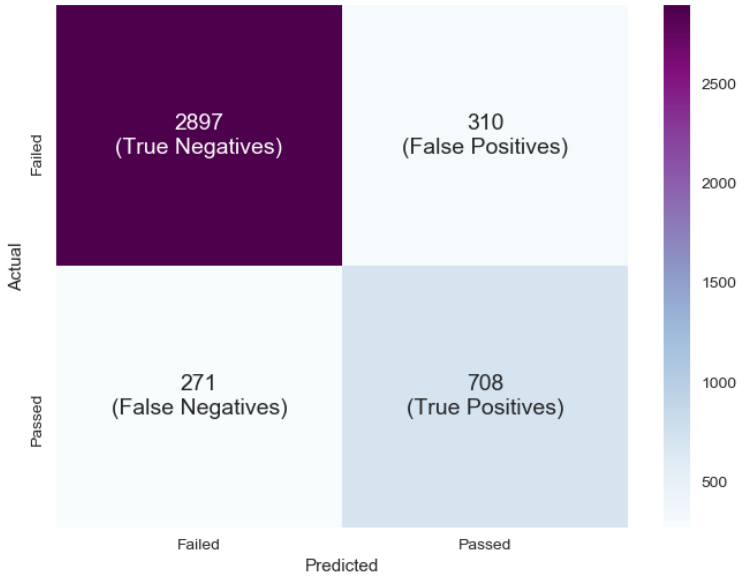


Figure 4.7: Confusion Matrix of Random Forest on test data

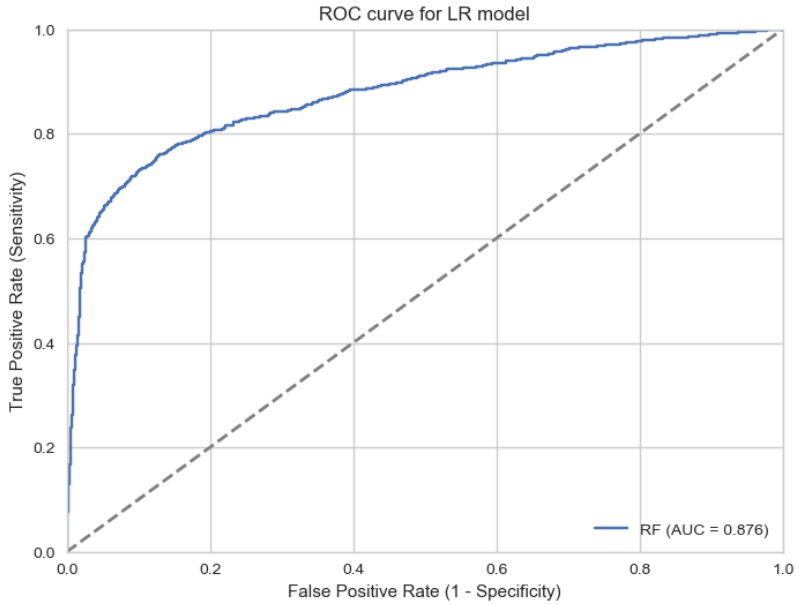


Figure 4.8: ROC curve for Random Forest on test data

data and is in fact generalizing the unseen data well.

The trained RF model was used to perform predictions on the unseen dataset that was created from the original dataset. The predictions on the unseen data define the generalizing power of the RF model at the individual student level as well as the academic year level.. The confusion matrix

generated from the predictions on unseen data is shown in the figure 4.9.

Figure 4.9: Confusion Matrix of Random Forest on unseen data viz 2016 academic year data

The decision tree model built on the training data had a classification accuracy of 0.867 on the unseen data. The Table 4.7 shows the other metrics calculated from the confusion matrix in Figure 4.9.

Table 4.7: Computed metrics based on actual and predicted unseen 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

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 then used to perform predictions on the test data. The confusion matrix generated from the predictions of the test data is as shown in Figure 4.10

The decision tree model built on the training data had a classification accuracy of 0.885. The table 4.8 shows the other metrics calculated from the confusion matrix in Figure 4.10.

ROC curve for the SVM model on test data was plotted and is as shown in the figure 4.11

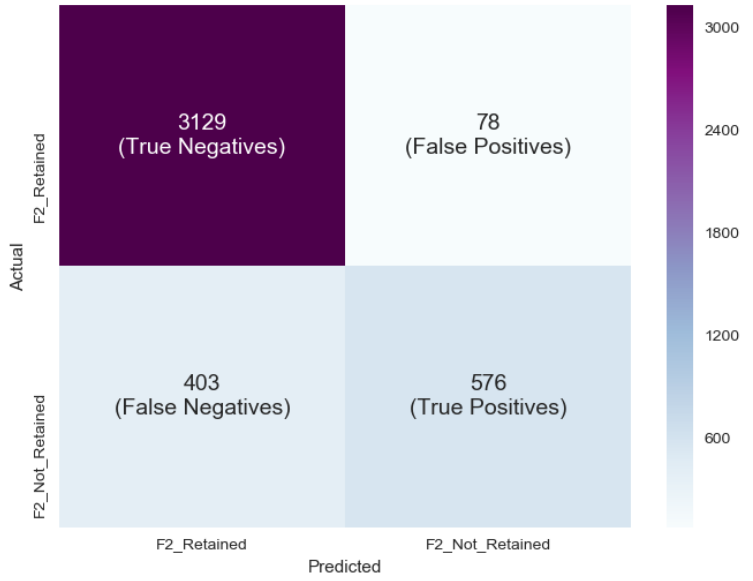


Figure 4.10: Confusion Matrix of Support Vector Machine model on test data

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

	Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F_1 Score**	**AUC**
SVM model	0.885	0.588	0.976	881	0.71	0.857

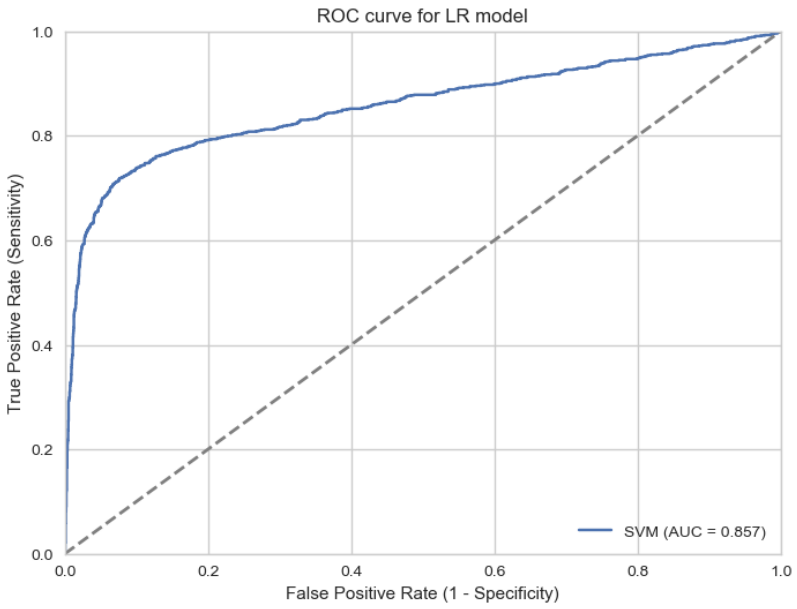


Figure 4.11: ROC curve for Decision Tree on test data

. 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. This clearly confirms that the model is actually learning from the training data and is in fact generalizing the unseen data well.

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

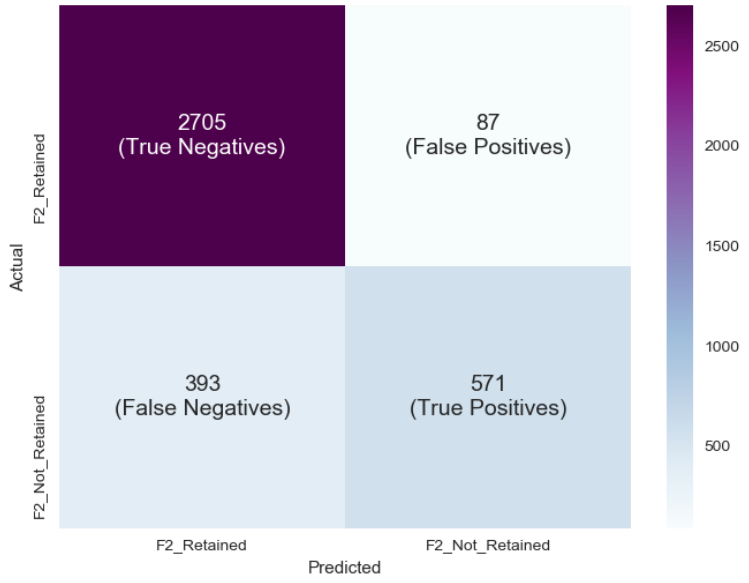


Figure 4.12: Confusion Matrix of SVM model on unseen data viz 2016 academic year data

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

Table 4.9: Computed metrics based on actual and predicted unseen 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

Table 4.10: Comparison of metrics from different models on the training 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	0738	0.72	0.860
RF model	0.861	0.723	0.903	695	0.71	0.876
SVM model	0.885	0.588	0.976	881	0.71	0.857

4.2.5 Comparison of the models

4.2.6 Feature importance calculation based on selected models.

4.2.7 Probability estimates to risk scores

Chapter 5

Conclusion

5.0.1 Future Work

Bibliography

[A+12] Alexander W Astin et al. *Assessment for excellence: The philosophy and practice of assessment and evaluation in higher education*. Rowman & Littlefield Publishers, 2012.

[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.

[BMZ+15] Jay Bainbridge, James Melitski, Anne Zahradnik, Eitel JM Lauría, Sandeep Jayaprakash, and Josh Baron. Using learning analytics to predict at-risk students in online graduate public affairs and administration education. *Journal of Public Affairs Education*, pages 247–262, 2015.

[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.

[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.

[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.

[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

Degrees:

Bachelor Degree in Computer Engineering 2014

Jawaharlal Nehru Technological University, Hyderabad, India

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. Laxmi Gewali, Ph.D.

Committee Member, Dr. Justin Zhan, Ph.D.

Graduate Faculty Representative, Dr. Magdalena Martinez, Ph.D.