A MACHINE LEARNING APPROACH TO PREDICT RETENTION OF COMPUTER SCIENCE STUDENTS AT UNIVERSITY OF NEVADA, LAS VEGAS

By

Sudhir Deshmukh

Bachelor of Engineering - Computer Engineering
University of Mumbai
2015

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science - Computer Science

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

University of Nevada, Las Vegas
May 2020

UNLV GRADUATE COLLEGE

Thesis Approval

The Graduate College
The University of Nevada, Las Vegas

April 21st, 2020

This thesis prepared by

Sudhir Deshmukh

entitled

A Machine Learning Approach to Predict Retention of Computer Science Students at University of Nevada, Las Vegas

is approved in partial fulfillment of the requirements for the degree of

Master of Science - Computer Science
Department of Computer Science

Fatma Nasoz, Ph.D.
Examination Committee Chair

Kazem Taghva, Ph.D.
Examination Committee Member

Laxmi Gewali, Ph.D.
Examination Committee Member

Magdalena Martinez, Ph.D.
Graduate College Faculty Representative

Kathryn Hausbeck Korgan, Ph.D.
Graduate College Dean

Abstract

Student retention is an important measure when in determining student success. Retention refers to the first-time full-time student from previous fall term who returned to the same university for the following fall term. Decline in retention rate have adverse effect on stakeholders, parents, and students view about the institution, revenue generated from tuition cost and obtaining outside funds. In an effort to increase retention rates, universities have started analyzing the factors that correlate with students dropping out. Many universities have identified some of these factors and are working on developing intervention programs to help students to elevate their academic performance and eventually retain them at the university. However, identifying the students who require this intervention is a very challenging task.

In this thesis, we propose the use of machine learning models to identify students who are at-risk of not being retained so that university administration can successfully deploy intervention strategies at an early stage and prevent the students from dropping out. We implemented classification algorithms including feed forward neural networks, logistic regression, and support vector machine to determine at-risk students. The data to train these models was gathered from University of Nevada Las Vegas (UNLV) enterprise data warehouse: UNLV Analytics. The models were evaluated on various metrics and the results showed that logistic regression model performed best in predicting at-risk students for first-year retention and feedforward neural networks performed best in predicting at-risk students in second-year retention at UNLV's Department of Computer Science.

Acknowledgements

It is a pleasure to thank all those individuals who contributed and extended their valuable assistance in the preparation and completion of this study, and without whose guidance and help this thesis would not have been possible.

Firstly, I owe my deepest gratitude to my advisor, Dr. Fatma Nasoz, for her encouragement, guidance and feedback throughout this thesis. She was always willing and enthusiastic to assist me in any way she could whenever I ran into a trouble spot or had any questions.

I extend my thanks to Dr. Kazem Taghva, Dr. Laxmi Gewali, and Dr. Magdalena Martinez for their support and for being part of my thesis committee.

I am gratefully indebted to Becky Lorig, Daisy Duarte, Carrie Trentham, and Kivanc Oner from the Enterprise Application Services department at UNLV for their assistance in understanding the essential details of student enrollment and retention problems at UNLV. They also helped me tremendously in finding the student data required for my research.

To conclude, I cannot forget to thank my parents Ashok Deshmukh and Kamal Deshmukh, my sister Komal Deshmukh and my brother Bhushan Deshmukh for being my strength during this intensive academic year. I would also like to thank Swati Vishwakarma, Partha Chudal, Aashi Maharjan, Ashish Tamrakar, and Shalini Ravilla for their unconditional and continuous support throughout my master's program.

Sudhir Deshmukh

University of Nevada, Las Vegas

May 2020

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

From the early 21st century, most American universities have been facing difficulties in retaining students [Lau03]. A student is called first-time full-time when the student is admitted for the first time in the university and has met the registered credit requirements defined by the university. In a given academic year, the first-time full-time students are entitled as cohorts. Cohorts are used in census data to report university retention and graduation rates. The Integrated Post-Secondary Education Data System (IPEDS) defines retention rate as the percentage of cohort who enrolled in the previous fall and continue to enroll in the current fall at the same university [Unia]. Universities often spend more money on admitting new students than focusing on retaining their current students [Lau03].

Dropout students are those who do not enroll in their sophomore year after finishing their freshman year in the same university, or students who transfer to another university. If students drop out of the university, it affects the university's graduation rate, as well as its reputation [YL04] [Lau03]. This impacts parents' and students' opinions about the university and can impact the admission rate. Retention has also become a crucial factor for a university to receive outside funding [Nas96]. There are various reasons why freshman students do not retain to their sophomore year [TRL$^+$94]. Some reasons for dropouts are beyond the limits of a university, such as students' financial situations, changing career goals, or unrelated personal affairs. Many students drop out or change their college due to an unhealthy environment that holds back students' learning and educational needs. Some students find it difficult to comprehend basic course work, and hence, are unable to cope with the normal course requirements. Adding to this, students in their freshman year might feel overwhelmed due to the transition from high school to college life [Lau03]. Apart from these limitations, universities must focus on the factors affecting student retention that are dependent on students' academic achievements and success. Academic success is the most important component in student retention and can predict a student's perseverance in their studies [DAM02] [PT05]. Academic factors include, but are not limited to, amount of credits taken per term, hours invested in extracurricular activities, time spent at the library, enrollment in difficult courses before completing pre-requisites, grades in homework/assignments, final grades achieved in each course, part time job workload, and so on [Lau03].

Retention is necessarily an important aspect in accord with institutional growth and success. Retention of students is necessary to maintain institutional financial goals and continue current academic programs, by maintaining and improving them accordingly. Furthermore, universities want their students to have a positive educational experience while they are studying in their institutions [FF08]. Retention can also be used to measure the productivity of the institution in a student's success [Tin06]. Additionally, if a student is retained by the institution, there is a better chance that he/she will graduate on time and enter the workforce to pay off any student loan debt [Lau03].

If a university is unable to retain its students, it may degrade stakeholders' and donors' views about the university [Lau03]. Thus, universities should focus on robust solutions to improve student retention. To do this, a university must determine which students are on the verge of dropping out and help them individually, to increase their academic performance. In a traditional approach to managing the retention problem, many universities use a rule-based system to identify potential "at-risk" students [BS16]. However, it fails to address various factors that affect student performance and often leads to low accuracy. Alternatively, to improve the identification of at-risk students, universities must use standard-based grading and implement models based on it, as standard-based grading provides clear, meaningful and personalized reports for individual students, which can then be used to develop generic models [MDDM16]. All models developed using machine learning have advantages over static, rule-based models and provide improved accuracy. These models can be trained using academic data to find at-risk students and can lead universities towards better solutions to increase retention.

Based on data obtained from University of Nevada Las Vegas' (UNLV) enterprise data warehouse, UNLV Analytics [unl], each year, on average, 7000 first-time full-time new undergraduate students are admitted to the University of Nevada, Las Vegas. Among these first-time full-time students, an average of 3700 students enroll in course work, which is approximately 53% of the admitted students [Unib]; this shows that only half of the admitted students actually enroll in the university. With the data collected from UNLV's enterprise data warehouse, research has found that in the Department of Computer Science (CS), the enrollment to admission ratio has been decreasing over the years. Each year the number of admitted students increases, but enrollment does not increase at the same rate, see Table 1.1.

Table 1.1: Enrollment to admission ratio in Department of Computer Science of first-time full-time students at UNLV

Year	Admitted count	Enrolled count	% enrolled in CS
2012	83	51	61.44
2013	136	75	55.14
2014	189	105	55.55
2015	194	103	53.09
2016	233	97	41.63
2017	184	80	43.47
2018	250	103	41.2
2019	295	117	39.66

Table 1.1 shows an alarming decrease in percentage of enrolled students in first-year over the years in Department of Computer Science at UNLV and has increased the necessity of maintaining the number of currently enrolled students. To maintain the current enrolled students, university administration must focus on locating students at-risk, in order to prevent dropouts and to increase the retention rate. The data collected from UNLV's enterprise data warehouse for first-year first-time full-time students for cohort years 2012 to 2017 shows that the number of students who are dropping out of UNLV is comparatively higher than the students who are switching majors at UNLV (Figure 1.1). However, second-year data shows that students at-risk are more likely to switch majors, instead of dropping out of the university (Figure 1.2).

The retention rate reported in the National Collegiate Retention and Persistence-to-Degree Rates 2018 is that 65.4% of all the undergraduates were retained for cohort year 2016, across the U.S. [ACT]. A federal branch of the U.S. Department of Education, National Center for Education Statistics (NCES) collects educational data, analyzes it, and reports statistical findings about American education. As per the NCES report, 76% of full-time students (including transfer stu-

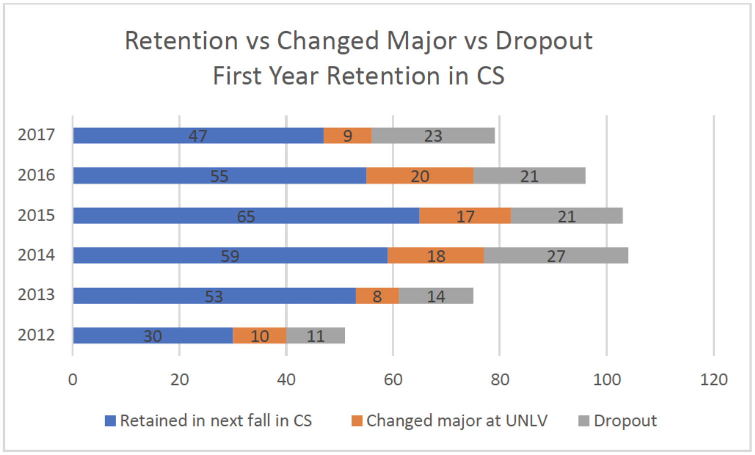


Figure 1.1: Retention vs. Changed major vs. Dropout counts of CS first-time full-time students at UNLV at the end of their first-year.

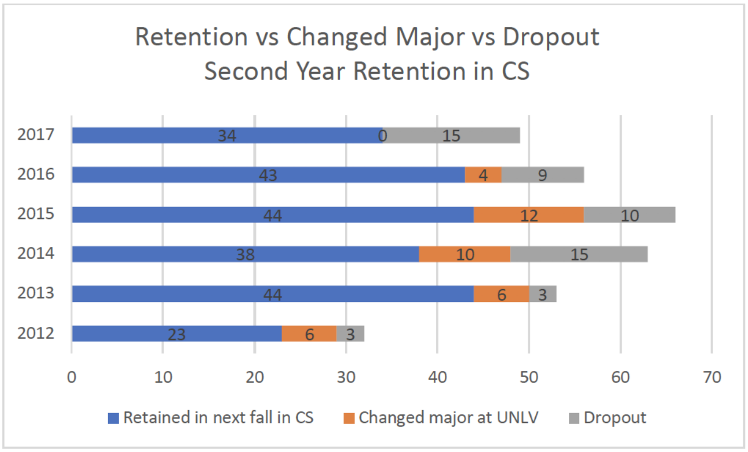


Figure 1.2: Retention vs. Changed major vs. Dropout counts of CS first-time full-time students at UNLV at the end of their second-year.

dents) were retained at UNLV for the cohort year Fall 2018 from Fall 2017 [Unia] (Figure 1.3).

1.1 Objective

The aim of this thesis is to build a predictive model to identify students who are at-risk of dropping out if timely intervention does not occur. To achieve this goal, we are training various machine learning models, including feedforward neural network (FNN), logistic regression (LR), and support vector machine (SVM) on student data collected at UNLV. All of the models are evaluated and compared using evaluation methods like precision, recall, specificity, F_1 score, fallout and accuracy. Using these models, the university, or the Department of Computer Science, can identify students

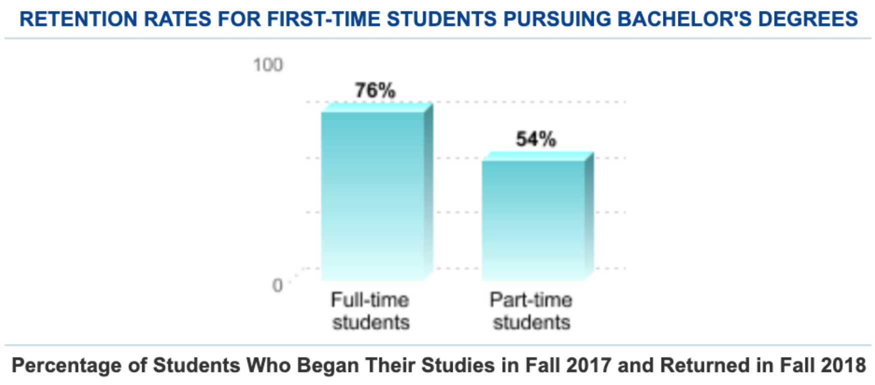


Figure 1.3: Retention of Freshman students of fall 2017 at UNLV

at-risk and help them elevate their academic performance through several methods, which include but are not limited to, academic advising, one-on-one attention, tutoring, collaborative learning [LL12], cooperative learning [Coo95], etc. These approaches will help improve the retention rate, and maintain the enrollment count of Department of Computer Science at UNLV.

Chapter 2

Background and Preliminaries

2.1 Related Work

Prediction of student retention has been a challenging problem for decades, and extensive research is being conducted at many universities to effectively retain students at institutions of higher studies. Tinto is an early researcher, who is one of the pioneers of extensive research on educational factors affecting student dropout. In his 1975 study, he stated that despite having a huge volume of literature on student retention and dropout, many studies do not clearly identify factors affecting dropouts, and because of this administration is unable to track the student population that requires assistance in the education process [Tin75]. Therefore, Tinto developed a theoretical model to analyze the relationship between students and the institution, which led to retaining students in the institution. Moreover, in 1999, Tinto suggested that institutes should follow interactive learning environments, such as communities that involve shared learning activities with students, teachers, and administration, which are helpful in retention [Tin99]. Tinto later involved academic factors into his theoretical model to make it more productive [Tin06]. Later, the input environment model developed by Astin et al., suggested that while determining retention, pre-college factors, such as gender, ethnicity, and high school GPA, should be considered [AA12]. Moreover, Linda suggested by providing funding, academic support and physical facilities to motivate students will increase retention [Lau03].

Along with determining the factors leading to dropout, researchers have started looking for groups of students who are likely to dropout of an institute, so that the institute can find ways to develop programs to retain these groups. Early researchers used traditional models to analyze retention

[Ast84] [Lau03] [Tin75]. However, recent technology has helped researchers to use analytical models, designed using machine learning, to find better results and improved accuracy than traditional models. Analytical models, such as logistic regression, decision tree, support vector machine, artificial neural network, random forest and other models, have been used to accurately locate students at-risk of dropping out [FF08] [KBP12] [MDDM16]. In 2018, Rajuladevi did a research on first-year retention rates at UNLV considering all students from all majors [Raj18]. He used logistic regression, decision tree, random forest classifier, and support vector machine models to find first-year retention at UNLV.

In this thesis, we developed machine learning predictive models to find students at-risk of dropping out, so that institutional intervention and support can be provided to help and retain these students in order to maintain enrollment in the department.

2.2 Preliminaries

2.2.1 What is Machine Learning?

The Oxford Dictionary defines learning as: "something that you learn, especially from your experience of working on something" [Oxf]. The learning process of any living thing is essential for their survival. This learning process starts evolving from birth; for example, lion cubs will not start hunting immediately after they born. They will learn each day by observing their parents hunting and survival skills, and try to do the same in their growing life to make appropriate decisions to survive. The same logic is used in computer applications when they learn based on a given training data, and produce a model that can be used to make decisions on previously unseen data, called test data [BÖ14].

One of the pioneers of machine learning, Arthur Samuel [Sam59], verified the fact that a machine will learn to play a better game than the person who initially wrote the program. Later, Tom Mitchell [Mit97] defined machine learning more formally: "A computer program is said to learn from experience **E** with respect to some task **T** and some performance measure **P**, if its performance on **T**, as measured by **P**, improves with experience **E**."

For example: task **T** can identify if a student will be retained at the same university or not;

performance measure **P** is the percent of students correctly identified as retained; and experience **E** is the previous students' retention information.

2.2.2 Machine Learning Algorithms

A machine learning model can learn in different ways. First, we can give the model a data with its output, so that it can learn which data points have more importance in giving the correct output. Once the model finishes its learning process, we can pass previously unseen data to the model to predict the output. Second, we can pass data without any information about the output, and the machine learning model will automatically find the similarities between different data points, and output a similar kind of data in groups.

2.2.2.1 Supervised Learning

When a model is given data with previously defined output for each data point, then it is called supervised learning [Kot07]. This model trains itself on training data to predict the output on test data. Once the model is trained enough to give the output using training data, and it has been determined how well it is predicting the output using the validation data, newly unseen test data will be given to the model to predict its output. This kind of data is called fully labeled data, which means each data point in the training data has its output tagged in the data.

The input data shown in Figure 2.1 contains features and output labels. It is divided into two sub datasets: (i) training data, and (ii) validation data. The training dataset is passed to the model in order to train the model using actual output labels attached to the data. Once the model is trained, the validation data is passed to the estimator model in order to analyze the performance of the model using numerous evaluation methods. Once we find the best model, this model is used to predict output for unseen test data.

Supervised learning algorithms are used to solve classification and regression problems.

2.2.2.2 Unsupervised Learning

Often, it is very hard to find data that already has output attached to it. To solve such a problem, we use unsupervised learning models. In this scenario, we provide the unlabeled data to the machine learning model, and the model automatically finds the useful information about the data

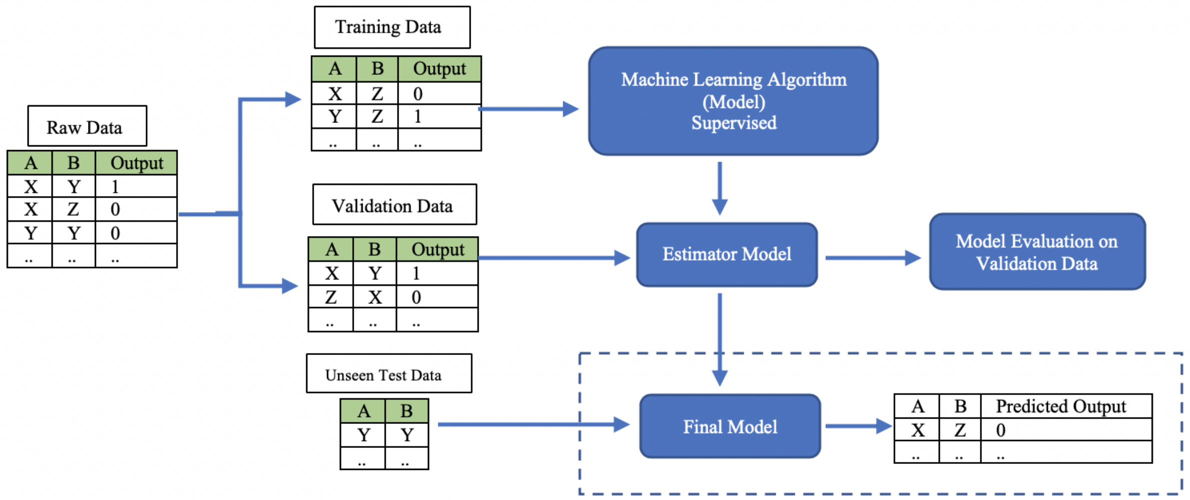


Figure 2.1: Flow of supervised learning algorithm

for us, and solves problems like clustering similar kinds of data, detecting anomalies, and so on [BÖ14]. In this scenario, we cannot compare the output of the model, and hence, it is harder to measure its accuracy and sensitivity.

In Figure 2.2, the input data is a set of images of cats and dogs [Pex]. This dataset is passed to an unsupervised model. This model will automatically extract the features from the given raw data and form clusters of similar data features.

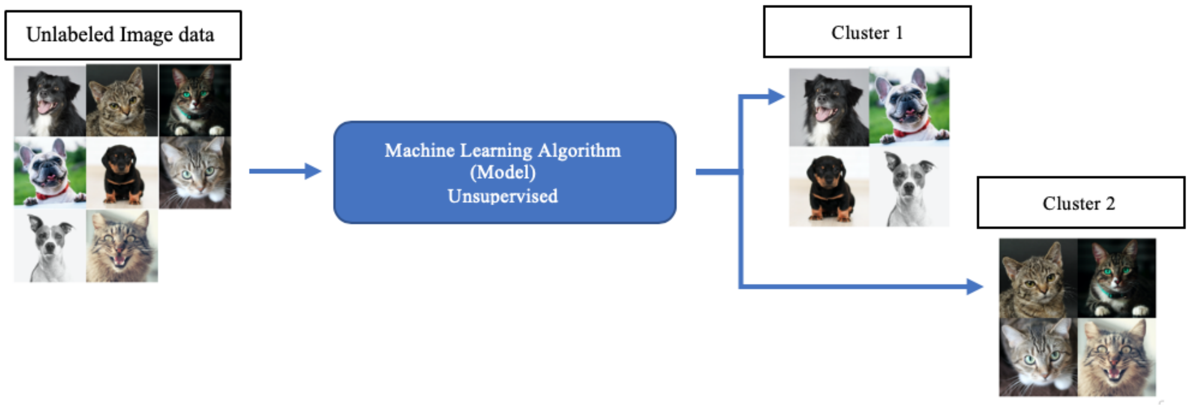


Figure 2.2: Flow of an unsupervised learning algorithm

2.2.3 Selected Models

The data we used was collected from UNLV's enterprise data warehouse, UNLV Analytics [unl], and has output labels that indicate if a student was retained in the next fall term or not. As our data contain labels, we used supervised learning approach to train our models. As our label data is discrete and contains only 0 and 1, our data can be classified as a classification problem. We have selected feedforward neural network, logistic regression, and support vector machine models for our data. These models are explained in next sub-sections.

2.2.3.1 Feedforward Neural Network

An artificial neural network (ANN) is a computer program that simulates of how a human brain works. In human brain, neurons transmit information gained by nerve cells. Brain contains billions of nerve cells that are inter connected which makes a network of nerves. Brain gets its input information from visuals, sounds or touch sensation and output neurons respond with muscle instructions. A similar concept is used in designing an artificial neural network [BMB+18]. In an ANN, neurons are represented by a mathematical function called perceptron. Perceptron is a single layered neural network. It has four unique components: input, bias and weights, summation function, and activation function; as shown in Figure 2.3.

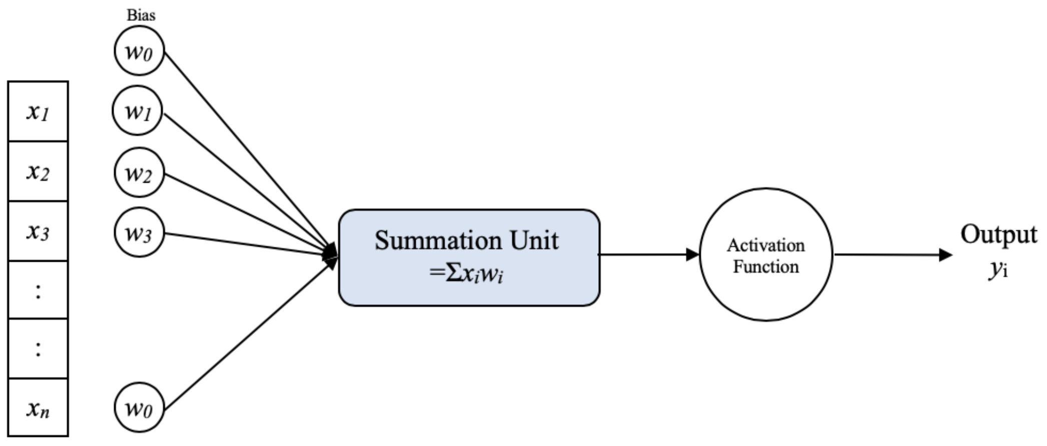


Figure 2.3: A single layered neural network: Perceptron

The logic of a perceptron is that the first and second components, i.e., input vector x and predefined weights, are multiplied and added in the third component to form a weighted sum. This weighted sum is then passed into an activation function. Finally, the output of the activation function maps the input vector to the output.

Different components of a perceptron

Input: Input x is nothing but a vector with numeric values of all the features selected for designing a neural network.

Weights: A randomly chosen value is assigned to each feature as a weight. Weights are the essential factors for finding the predicted output. The weight of each feature determines how important that feature is in predicting the output of that input vector x.

Summation: The summation function gives the intermediate output by taking the multiplication of feature value and its weights, and finally, adding all of the multiplications. The summation function is usually referred as a node.

Activation function: Activation function is used to determine the final output, i.e., the predicted output of input vector x. There are different types of activation functions such as sigmoid, tanh, rectified linear unit (ReLU), etc. Each activation function has different way of interpreting the output of summation component. For example, to find the predicted output between -1 and 1, we can use the tanh activation function.

Feedforward Neural Network also known as Multilayer Perceptron (MLP)

In this neural network, there are more than one layer of perceptrons. These layers are called hidden layers. The output of each hidden layer is taken as input to the next hidden layer, and the cycle goes on until the last layer of the network, which is the output layer. The complexity of the model depends on number of hidden layers and the number of nodes in each hidden layer. When all the nodes are connected to each other, it is called as a fully connected network. A simple fully connected multilayer perceptron is shown in Figure 2.4.

The feedforward neural network starts by calculating the summation (a_n) of the multiplications of input features and their initial random weights for all nodes in the first hidden layer. Each hidden layer will have initial weights for all input features. The final output is then calculated using Equation 2.1.

$$a_n = W.x_n \tag{2.1}$$

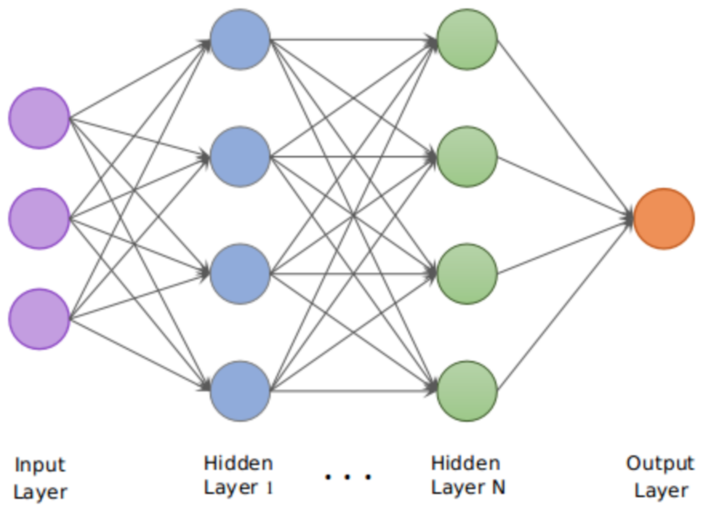


Figure 2.4: Fully connected multilayer perceptron [Mal].

where W is the matrix of the initial weights of all the nodes of a hidden layer and x_n is a row of an input data.

In this thesis, once we find the summation for all the nodes, we use a *ReLU* activation function to get the output of the hidden layers. We will define this output as Z_n, which is calculated using the mathematical function defined in Equation 2.2.

$$Z_n = ReLU(a_n) \tag{2.2}$$

ReLU is a mathematical function that maps the linear combination into the range of (0, x) and is defined in Equation 2.3. The *ReLU* function takes identity for all positive values of input, and zero for all negative values of input.

$$ReLU(x) = max(0, x) \tag{2.3}$$

Similarly, for the next hidden layer, Z_n will be the input vector. Using this input, we will find the computation of the summation of the multiplication of the input and weights (a_n). Once all the hidden layers are computed, we will use an activation function to find the predicted output ŷ for the input data using Equation 2.4.

$$\hat{y} = sigm(a_n) \tag{2.4}$$

There are various activation functions that can be used in output layer. For this thesis, we are using a sigmoid function. A sigmoid function *sigm* is a mathematical function that maps the linear combination into the range of (0,1) and is defined in Equation 2.5.

$$sigm(t) = \frac{1}{1+e^{-t}} \tag{2.5}$$

Feedforward Neural Network with Backpropagation

The feedforward neural network initially selects random weights for each hidden layer, and hence, the predicted output at the end of first iteration may not be correct. In 1986, Geoffrey E. Hinton proposed a way to update initial randomly selected weights to reduce the difference between actual output and predicted output by neural networks [RHW86]. The weight adjustment occurs in all the hidden layers. To perform the backpropagation logic, we need to continue from Equation 2.5. Once we get the final output of the first run of the feedforward neural network, we will find the difference (δ) between the actual output and predicted output using Equation 2.6. This difference is called as "error."

$$\delta_1 = \hat{y} - y \tag{2.6}$$

Once we compute the error, we will send this error backward to the previous hidden layer to compute the error using Equation 2.7. All the weights (W) and a_n used here are from the same hidden layer.

$$\delta_2 = (\delta_1 * W) * sigm(a_n) * (1 - sigm(a_n)) \tag{2.7}$$

The output from Equation 2.7 is used to compute the error in the second to the last hidden layer. Once we get the δ_1 and δ_2, we will update the weights for each hidden layer. Equation 2.8 updates the weights in the hidden layers by using the error and the input to the hidden layer. This intermediate weights update is called gradients.

$$W_\delta = \sum \delta_1 * Z_n \tag{2.8}$$

The final weights are updated by using the learning rate (α) or steps size of the neural network, denoted in Equation 2.9. A neural network model with lower learning rate takes more time to train.

$$W = W - \alpha * \frac{1}{N} * W_\delta \tag{2.9}$$

Finally, this whole process runs n number of times to reduce the error in the predicted and actual output. The final weights are used on previously unseen data to predict the output, using Equations 2.1 and 2.4.

2.2.3.2 Logistic Regression

Logistic regression is a classification algorithm used when output labels are binary (0 and 1) [Hen06]. It is a supervised machine learning model for classification which is built based on linear regression. Linear regression is built using available input data with actual output variables, which is later used to find the predicted output of unknown data. The linear regression model linearly combines input data (X) using weights to predict a real-value output (y). An explanatory plot of linear regression is shown in Figure 2.5. A similar method is used in logistic regression; however, it calculates the probability of the outcome of the categorical output.

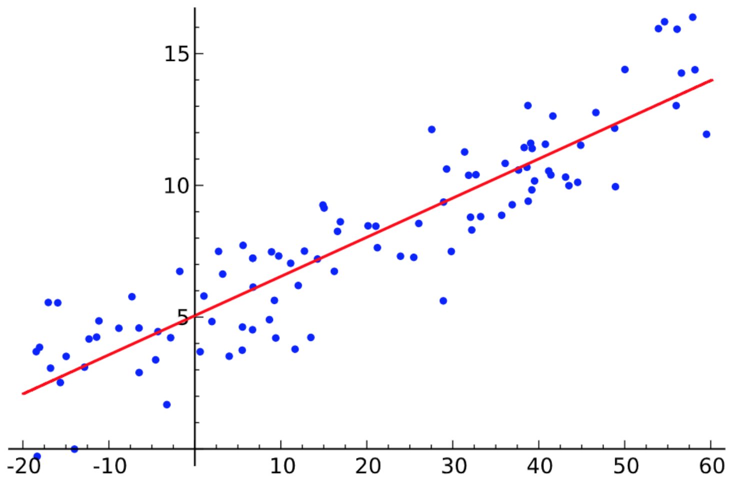


Figure 2.5: Plot of the linear regression model vs input data [Wik]

For example, it can give a result like the probability of the outcome of the output = 1 is 0.91, which means there is a 91% chance that outcome will be 1. The equation for logistic regression for one feature is expressed using Equation 2.10.

$$P(y = 1) = sigm(w_0 + w_1 * x) \tag{2.10}$$

The logistic function is used to convert the log-odds to probabilities. In the Equation 2.10, *sigm* is a mathematical function called sigmoid function. This function maps the linear combination in the range of (0,1), which is probability of the output (Figure 2.6). Equation 2.5 defines the *sigm* function. For each independent variable, a coefficient is calculated, which defines the importance of that variable in prediction. In Equation 2.10, w_1 is a coefficient for x variable and w_0 is the intercept. The value of the intercept determines how all features are useful in finding the predicted output. If the value of intercept is huge, then the model is under-fitting, which means the features are less valuable in predicting the output.

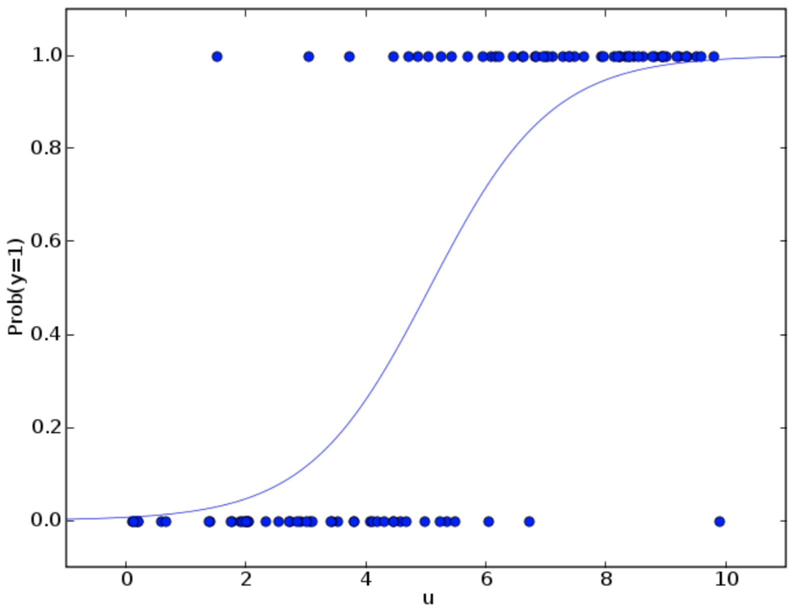


Figure 2.6: Logistic regression plot of predicted output [cvx]

2.2.3.3 Support Vector Machine

Support vector machine (SVM) is a supervised algorithm used in classification and regression machine learning problems. An SVM algorithm finds the boundary between different classes by maximizing the perpendicular distance between the boundary and the class points. This boundary is called hyperplane [CV95]. For 2-dimensional data, only a line, i.e. a 1-dimensional hyperplane, is found. Similarly, for 3-dimensional data, a 2-dimensional hyperplane is found. The equation for the hyperplane is defined in Equation 2.11.

$$w_0 + w_1 * x_1 + w_2 * x_2 + ... + w_n * x_n = 0 \tag{2.11}$$

In Figure 2.7, the SVM model will find the hyperplane of linearly separable data. This line will separate two features completely. When it is possible to find a hyperplane that linearly separates data, we can optimize the hyperplane by finding the margin and the support vectors for each class. The margin is found by calculating the perpendicular distance of data points to the hyperplane, and choosing the data points that have the minimum distance from the hyperplane. The data point inside the margin are called support vectors. In reality, there is a very low chance that a dataset is linearly separable. The problem with such non-linearly separable data can also be solved using SVM. For this, SVM has different methods that are used to find the hyperplane and support vectors. We can use soft margin or kernal trick to solve such problems.

When there are a few data points that are mixed into incorrect classes and because of which, we

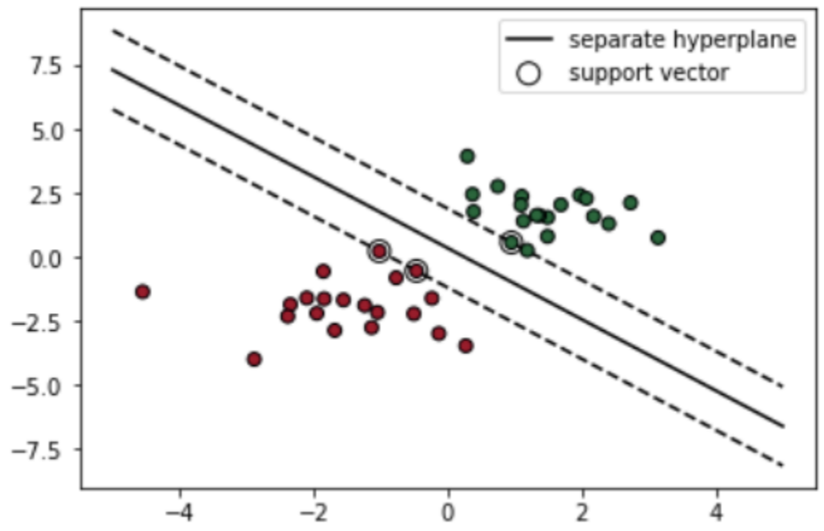


Figure 2.7: Plot of an SVM model after finding the hyperplane [Che]

cannot find a linearly separable hyperplane, by using soft margin, SVM can permit a few points that are misclassified and find the maximum margin. The degree of soft margin will decide the number of support vectors. The higher the soft margin, the more penalties the SVM gets when it makes a misclassification.

Kernel trick is used when the data points are linearly non-separable. Kernel trick will generate new features using old features to find the nonlinear hyperplane. There are various kernel functions available with SVM. We can use a linear, polynomial, radial basis function (rbf), sigmoid, or a self defined kernel function. Figure 2.8, shows different plots generated using kernel functions.

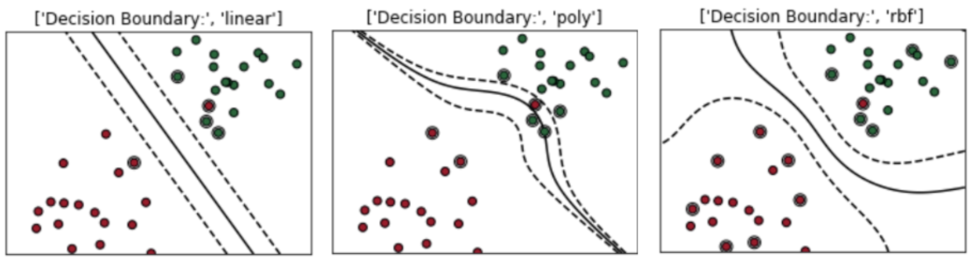


Figure 2.8: SVM hyperplane plots of various kernel functions [Che]

For example, if you have a feature with values like X=[3, 1, 0, -1, -3] and the actual output labels are y=[1, 1, 0, 1, 1], the data is linearly non separable, as all points lies on the same line. To make them linearly separable, we can use a polynomial kernel trick, where we can use X^2 to create a new feature that can be used with the old features to linearly separate the two classes.

2.2.3.4 Oversampling

Classes that are under-represented due to a fewer number of records available in the training data will shift the results to the high density classes. To overcome this problem of under representation of a class, we use oversampling techniques to increase the under-sampled class size to be equal to the other classes by replacing the current available samples of that class in the dataset. There are various ways to over-sample the classes: naive random over-sampling, Synthetic Minority Over-Sampling Technique (SMOTE) [CBHK02], and Adaptive Synthetic (ADASYN) [HBGL08]. In this thesis, we have used the random over-sampling technique to increase the number of data points in under-represented class.

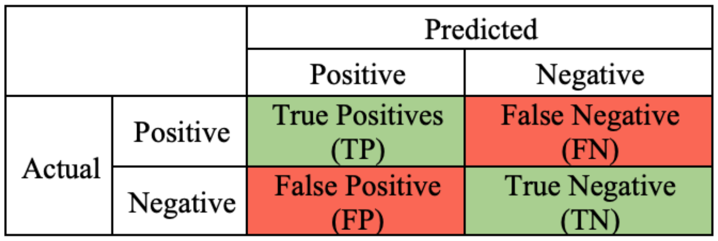
2.2.4 Evaluation Methods

Various evaluation methods are used in this thesis to compare the effectiveness and the performance of the selected machine learning models. The best model is considered based on how well a trained model performs on previously unseen data. If a model is performing exceptionally well on predicting near to equal predictions, it is considered as well trained, or the best model. To determine how well a model is working on previously unseen data, we divide our raw data into two parts: (i) training data, and (ii) test data. Both datasets will have actual output attached to them. Using training data, we will train the model with its features and actual outputs. The test data, on the other hand, is used to test the model trained on the training data. Later, the predicted output of test data is compared with the test data's actual output to determine the effectiveness of the model.

There are many ways to measure the performance of a machine learning model. The performance measures help us to improve the machine learning models that perform poorly in the initial phase. There are various metrics available in machine learning to measure performance on the basis of the problem type. This thesis is based on a classification problem, therefore, to evaluate the models, we use confusion matrix, accuracy, specificity, precision, and many other evaluation methods to define the effectiveness of the model. The next sub-section will define these metrics in detail.

2.2.4.1 Confusion Matrix

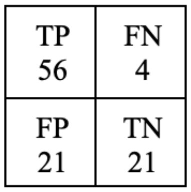
A confusion matrix represents the predicted output compared with the actual output. The confusion matrix also shows how many times a model is correctly predicting the output compared to the actual output. The confusion matrix reports the true positives, false positives, true negatives, and false negatives. These values are used to compute other metrics such as accuracy, precision, specificity and recall. The representation of a confusion matrix is shown in Figure 2.9.



		Predicted	
		Positive	Negative
Actual	Positive	True Positives (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Figure 2.9: Representation of a confusion matrix

To get a better understanding about how a confusion matrix works, an example confusion matrix for binary classification is shown in Figure 2.10. A confusion matrix is always a $N \times N$ matrix, where N represents the number of classes. A confusion matrix always calculates the positives and negatives of model prediction by comparing one class with the rest of the classes. The confusion matrix shown in Figure 2.10 represents the output of 102 data points.



TP 56	FN 4
FP 21	TN 21

Figure 2.10: Example of a confusion matrix

Out of total data points, 60 data points are actual positives, and 42 data points are actual negatives. Explanations and examples of true positive, true negative, false positive, and false negative are given below.

True Positive: The number of true positives represents how many actual true outputs are predicted as true by the model. Figure 2.10 shows that 56 out of 102 data points are true positive.

False Negative: The number of false negatives represents how many actual true outputs are predicted as false by the model. A false negative example represents a **type II** error. A **type II** error shows how many times a model failed to reject the false output. Figure 2.10 shows that four out of 102 data points are false negative.

True Negative: The number of true negatives represents how many actual false outputs are predicted as false by the model. Figure 2.10 shows that 21 out of 102 data points are true negative.
False Positive: The number of false positives represents how many actual false outputs are predicted as true by the model. A false positive example represents a **type I** error. A **type I** error shows how many times a model failed to reject the true output. Figure 2.10 shows that 21 out of 102 data points are false positive.

2.2.4.2 Accuracy

The accuracy of a classification problem is the most common evaluation metric in machine learning. Accuracy is defined as the number of correctly classified data points in a dataset to the whole population of the dataset. The accuracy can be found by using a confusion matrix and is calculated by taking ratio of the total number of true positives and true negatives to the total number of data points in the dataset. The formula for accuracy using a confusion matrix is shown in Equation 2.12.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{2.12}$$

For example, based on the confusion matrix depicted in Figure 2.10, the accuracy can be calculated as (56+21)/102, which is 0.7549.

2.2.4.3 Precision

Precision is defined as the ratio of number of correctly predicted positive examples to the total number of predicted positive examples. Equation 2.13 defines the calculation of precision.

$$Precision = \frac{TP}{TP + FP} \tag{2.13}$$

For example, based on the confusion matrix depicted in Figure 2.10, the precision can be calculated as 56/(56+21), which is 0.7272.

2.2.4.4 Specificity

Specificity is defined as the ratio of number of correctly predicted negative examples to the total number of negative examples. Equation 2.14 defines the calculation of specificity.

$$Specificity = \frac{TN}{TN + FP} \tag{2.14}$$

For example, based on the confusion matrix depicted in Figure 2.10, the specificity can be calculated as 21/(21+21), which is 0.50.

2.2.4.5 Recall

Recall is defined as the ratio of number of correctly predicted positive examples to the total number of positive examples. Equation 2.15 defines the calculation of recall.

$$Recall = \frac{TP}{TP + FN} \tag{2.15}$$

For example, based on the confusion matrix depicted in Figure 2.10, the recall can be calculated as 56/(56+21), which is 0.7272.

2.2.4.6 Fallout

Fallout is defined as the ratio of number of incorrectly predicted negative examples to the total number of negative examples. Equation 2.16 defines the calculation of fallout.

$$Fallout = \frac{FP}{FP + TN} \tag{2.16}$$

For example, based on the confusion matrix depicted in Figure 2.10, the fallout can be calculated as 21/(21+21), which is 0.50.

2.2.4.7 F_1 Score

The harmonic mean of precision and recall gives the F_1 score. It is calculated by using the Equation 2.17.

$$F_1 = \frac{2 * (precision * recall)}{Precision + Recall} \tag{2.17}$$

For example, based on the confusion matrix depicted in Figure 2.10, the F_1 Score can be calculated as 2*(0.7272* 0.7272)/(0.7272+0.7272), which is 0.7271.

2.2.4.8 Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC)

A receiver operating characteristic curve (ROC) is a graphical plot illustrating the connection between sensitivity and specificity for every possible cut-off. Most binary classification models use a 0.5 threshold on probability to predict classification. On the other hand, ROC will plot a graph with different threshold values from 0.0 to 1.0, using their true positive rates and false positive rates. A false positive rate is obtained from the specificity of the model, as 1-specificity. The illustrative ROC curve is shown in Figure 2.11. The ROC curve is used to compute the performance of the model by using Area Under the Curve (AUC). The greater the area, the better the model in prediction.

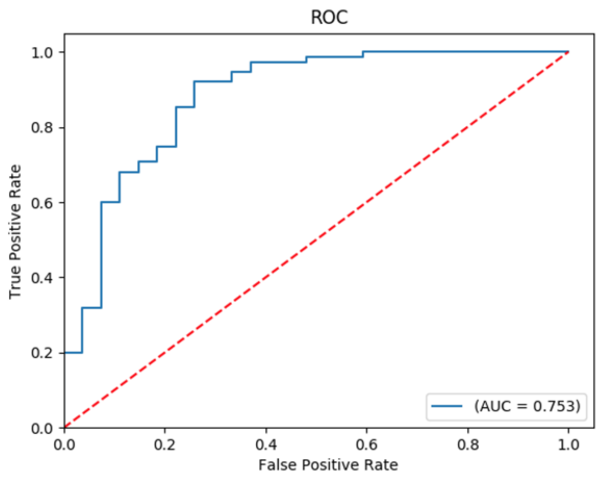


Figure 2.11: Illustration of ROC and AUC

2.2.4.9 Cross Validation

The most common method to train and validate a machine learning model is to divide the input data into two parts; generally, the ratio will be 80% training data and 20% validation data. This split of data is generally done by random sampling. This is a holdout validation technique to check the proficiency of the trained model on the validation data. However, random sampling, may or may not split data evenly, and can create combinations. Due to this uneven data, there is a chance that, all of one class of data may end up in only the test data, and hence, the model will not see such data while training, and will not predict the correct output for such data. To manage this issue, we use cross validation.

The best cross-validation methods include k-fold and stratified k-fold cross-validation. In these, the training data is divided into several chunks, and then excluding one chunk, all other chunks are used to train the model, and the excluded chunk is used to validate the model's performance. This process is repeated for all chunks.
In k-fold cross-validation, shown in Figure 2.12, the data is split into k parts. Within these, k-1

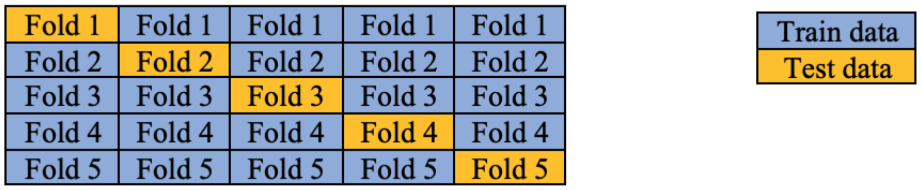


Figure 2.12: K-Fold cross validation model (K=5)

parts are used as training data, and one part is used as test data. Then holdout cross-validation is done on test data to evaluate the performance of the model. The same process is repeated k times with different test data. At the end, an analysis of the average scores is used to decide the model's performance on all of the data. Similarly, in stratified k-fold, the subsets are created by equally dividing the imbalanced data. Each fold will contain approximately the same percentage of data from each class.

Chapter 3

Data Description

3.1 Data Collection

Data collection was one of the most difficult aspects of this thesis, and required substantial literature review as well as expert help to select the features. The data for this thesis was collected from UNLV's enterprise data warehouse, UNLV Analytics [unl]. This data warehouse is used to generate a large number of reports. These reports are used to provide retention, graduation, and other information about the university to the IPEDS. The following subsection describes the collection of data for this thesis.

3.2 Data Extraction form UNLV Analytics

UNLV Analytics provides student data such as admission, enrollment, etc. for reporting. The data is stored for each census date. We extracted the admission and enrollment data of first-time full-time students from the years 2012 to 2017. This data contains students' enrollment history for each term, as well as their grade point average (GPA). Along with current academic data, we collected each student's high school GPA and their demographic information.

3.2.1 SAS Data Integration Studio (DIS) for Data Transformation

The SAS Data Integration Studio (DIS) [LF15] graphical user interface allows for the creation of complex transformation flows to transform raw data into warehouse data for reporting. We created several DIS applications to combine different data files collected from UNLV Analytics to create two final output files: first-year retention and second-year retention. Finally, the combined files were used as raw data for machine learning models. Figure 3.1 shows one of the applications designed in SAS DIS for this thesis.

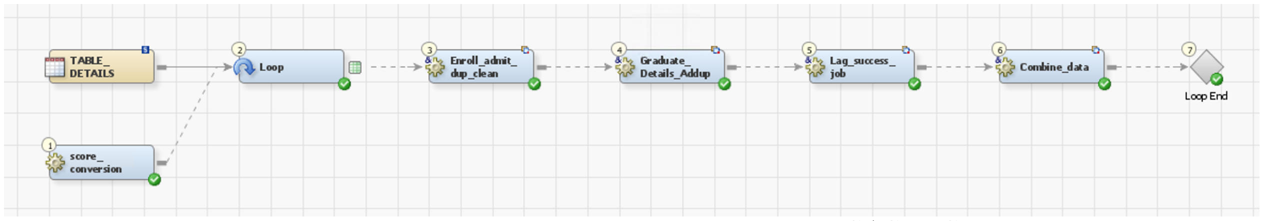


Figure 3.1: Master application designed in SAS DIS

3.2.2 Data Description

The data of computer science undergraduate students for years 2012 to 2017 was collected. This data contains several .csv files, which include admission and enrollment, along with demographical and high school information. The data instances which matched the criteria set on the academic plan field (CSPRE, CSCBS, CSCBA) were extracted from UNLV Analytics. The raw data contains a total of 508 first-time full-time students enrolled in academic plan such as CSPRE, CSCBS, and CSCBA at UNLV from 2012 to 2017. The data includes a variety of features about each student, such as demo-graphical information like age at admission, ethnicity, high school academic data, American College Testing (ACT) score, and each term's GPA, as well as total credits enrolled in each term and course grades for each course taken in each term; it also shows whether the student was retained in the second/third year in the CS major or at UNLV. We captured 25 features and grades in all courses taken by all first-time full-time students enrolled from 2012 to 2017. In total, we captured 320 features in first-year retention dataset and 387 features in second-year retention dataset. Table 3.1 shows the features captured for each student for first-year and second-year retention

Table 3.1: Description of features for student retention data.

No.	Feature	Type	Description
1	Age	Numerical	Age at the time of admission
2	Gender	Binary Nominal	Student Gender (M, F)
3	USA_Citizen	Binary Nominal	Citizen status of a student (Y, N)
4	Non_Resident_Alien	Binary Nominal	Alien status of a student (Y, N)

Continued on next page

Table 3.1 – continued from previous page

No.	Feature	Type	Description
5	IPEDS_Race_Ethnicity	Multi Nominal	Ethnicity of a student
6	Core_High_School_GPA	Numerical	Core High School GPA of a Student
7	Unweighted_High_School_GPA	Numerical	Unweighted High School GPA of a Student
8	Weighted_High_School_GPA	Numerical	Weighted High School GPA of a Student
9	At_risk_First_Generation	Multi Nominal	Status of first-generation student at risk (Y, N, U)
10	At_risk_Pell	Binary Nominal	Status of a student at risk of not getting Pell Scholarship
11	Application_Term	Multi Nominal	Admitted term year number
12	ACT_score_final	Numerical	ACT score of a student
13	F1_GPA	Numerical	GPA of first year fall term
14	S1_GPA	Numerical	GPA of first year spring term
15	F2_GPA	Numerical	GPA of second year fall term
16	S2_GPA	Numerical	GPA of second year spring term
17	Cum_GPA	Numerical	Cumulative GPA
18	F1_Crd_Enrl	Numerical	Credits enrolled in first year in fall
19	S1_Crd_Enrl	Numerical	Credits enrolled in first year in spring
20	F2_Crd_Enrl	Numerical	Credits enrolled in second year in fall
21	S2_Crd_Enrl	Numerical	Credits enrolled in second year in spring
22	S1_Ret_CS	Binary Nominal	If student retained for first year spring in CS major {0, 1}
23	S1_Ret_UNLV	Binary Nominal	If student retained for first year spring at UNLV {0, 1}

Continued on next page

Table 3.1 – continued from previous page

No.	Feature	Type	Description
24	S2_Ret_CS	Binary Nominal	If student retained for second year spring in CS major {0, 1}
25	S2_Ret_UNLV	Binary Nominal	If student retained for second year spring at UNLV {0, 1}

The first-time full-time admitted CS students can enroll into courses offered by other departments. Using our data, we found most of these students enroll into courses required by the CS department to begin their academic careers. Table 3.2 and 3.3 show the top 20 courses taken by students in their first and second years for fall and spring terms.

Table 3.2: First Year Course Count

Course	Count	Course	Count
F1_EGG_101_Grd	325	S1_ENG_102_Grd	197
F1_ENG_101_Grd	249	S1_COM_101_Grd	130
F1_SOC_101_Grd	152	S1_CS_135_Grd	108
F1_MATH_127_Grd	126	S1_CPE_100_Grd	98
F1_THTR_100_Grd	119	S1_MATH_181_Grd	94
F1_MATH_126_Grd	113	S1_EGG_101_Grd	84
F1_HIST_101_Grd	110	S1_CS_202_Grd	76
F1_CS_135_Grd	107	S1_ENG_101F_Grd	72
F1_MATH_181_Grd	96	S1_MATH_127_Grd	71
F1_ENG_101E_Grd	95	S1_PSY_101_Grd	67
F1_MUS_125_Grd	71	S1_MATH_182_Grd	66
F1_PSY_101_Grd	69	S1_PHIL_114_Grd	62
F1_CPE_100_Grd	49	S1_MATH_126_Grd	60
F1_ENG_102_Grd	44	S1_HIST_101_Grd	52
F1_DAN_166_Grd	43	S1_HIST_102_Grd	51

Continued on next page

Table 3.2 – continued from previous page

Course	Count	Course	Count
F1_COM_101_Grd	39	S1_CPE_100L_Grd	50
F1_MATH_95_Grd	36	S1_SOC_101_Grd	42
F1_MATH_182_Grd	34	S1_ENG_101_Grd	41
F1_PHIL_114_Grd	31	S1_PSC_100_Grd	38
F1_MATH_96_Grd	30	S1_PSC_101_Grd	38

Table 3.3: Second Year Course Count

Course	Count	Course	Count
F2_PHIL_114_Grd	87	S2_CS_302_Grd	60
F2_CPE_100L_Grd	75	S2_CS_202_Grd	55
F2_CPE_100_Grd	74	S2_MATH_251_Grd	46
F2_CS_202_Grd	74	S2_GEOL_101_Grd	44
F2_ENG_102_Grd	68	S2_CS_218_Grd	42
F2_COM_101_Grd	65	S2_MATH_182_Grd	40
F2_MATH_182_Grd	61	S2_ENG_407B_Grd	39
F2_CS_135_Grd	60	S2_CPE_100_Grd	38
F2_MATH_181_Grd	52	S2_MATH_181_Grd	38
F2_PHIL_242_Grd	46	S2_PHIL_114_Grd	38
F2_CS_218_Grd	43	S2_COM_101_Grd	35
F2_MATH_251_Grd	42	S2_CS_219_Grd	35
F2_GEOL_101_Grd	38	S2_PHIL_242_Grd	34
F2_MATH_126_Grd	30	S2_CS_135_Grd	26
F2_MATH_127_Grd	27	S2_PSY_101_Grd	26
F2_CS_302_Grd	23	S2_CPE_100L_Grd	25
F2_HIST_102_Grd	22	S2_ENG_102_Grd	21
F2_COE_202_Grd	21	S2_MATH_126_Grd	21

Continued on next page

Table 3.3 – continued from previous page

Course	Count	Course	Count
F2_ENG_231_Grd	21	S2_ENG_231_Grd	18
F2_WMST_113_Grd	19	S2_HIST_102_Grd	18

3.2.3 Feature Extraction

The process of creating or extracting important features from raw data is known as feature extraction. We collected a full history of all of the first-time full-time students in the CS major from 2012 to 2017. This data contains information about students' ACT scores, course letter grades, credits taken, and other features. We converted a few features into the required format to use them in our machine learning models. The following sub-section describes the computation of few new features.

3.2.3.1 Computation of ACT_score_final Feature

Campus wide, all students are required to report their standardized exam scores, such as ACT and Scholastic Assessment Test (SAT), when applying for any program at the university. There are number of students who submit only one of these scores, which makes it difficult to combine these scores as one feature. The SAT exam has three variations: before 2005, SAT had a 1600 maximum score; after 2005, the maximum score was changed to 2400; and recently, in 2016, it was again changed to 1600. To convert these score variations, we used the College Board's SAT concordance tables [Colb] [Cola]. We converted all SAT scores into the 2016 SAT variation. After this, we still had some missing values, either in SAT scores or in ACT scores. We then converted all the SAT scores into ACT scores. The maximum ACT score is 36. The ACT_score_final feature was then used to represent this conversion of all SAT and ACT scores into ACT scores.

3.2.3.2 Computation of S1_Ret_CS Feature

S1_Ret_CS is the feature that is calculated on the basis of students' enrollment in the first-year spring term in the CS major after completing the first-year fall term in the CS major. This feature has a binary flag value of 0 and 1.

3.2.3.3 Computation of S1_Ret_UNLV Feature

S1_Ret_UNLV is the feature that is calculated on the basis of students' enrollment in the first-year spring term after completing the first-year fall term at UNLV. This feature has a binary flag value of 0 and 1. If a student changes their major or stays in the CS major at UNLV and has enrolled, then flag to this feature is set to 1. Otherwise, if a student has dropped out of UNLV, then the flag is set to 0.

3.2.3.4 Computation of S2_Ret_CS Feature

S2_Ret_CS is the feature that is calculated on the basis of students' enrollment in the second-year spring term in the CS major after completing the second-year fall term in the CS major. This feature has a binary flag value of 0 and 1.

3.2.3.5 Computation of S2_Ret_UNLV Feature

S2_Ret_UNLV is the feature that is calculated on the basis of students' enrollment in the second-year spring term after completing the second-year fall term at UNLV. This feature has a binary flag value of 0 and 1. If a student changes their major or stays in the CS major at UNLV and has enrolled, then the flag to this feature is set to 1. Otherwise, if a student has dropped out of UNLV, then the flag is set to 0.

3.2.3.6 Computation of F2_Ret_CS Output Label

The main goal of this thesis is to identify the students who are not going to be retained in CS in their second-year. This information is stored in the output label F2_Ret_CS and is calculated on the basis of students' enrollment in the second-year fall term in CS after completing the first-year in CS. This output label has a binary flag value of 0 and 1.

3.2.3.7 Computation of F2_Ret_UNLV Output Label

F2_Ret_UNLV is the output label which is calculated on the basis of students' enrollment in the second-year fall term after completing their first-year at UNLV. This output label has a binary flag value of 0 and 1.

3.2.3.8 Computation of F3_Ret_CS Output Label

The main goal of this thesis is to identify the students who are not going to be retained in CS in their third-year. This information is stored in the output label F3_Ret_CS and is calculated on the basis of students' enrollment in the third-year fall term in CS after completing the second-year in CS. This output label has a binary flag value of 0 and 1.

3.2.3.9 Computation of F3_Ret_UNLV Output Label

F3_Ret_UNLV is the output label which is calculated on the basis of students' enrollment in the third-year fall term after completing their second-year at UNLV. This output label has a binary flag value of 0 and 1.

3.3 Data Pre-processing

Most features in a real-world dataset cannot be used as an input to a machine learning model. We need to transform these data features into the format required by the model. After transformation, we get a clean dataset, which can be used for analysis. The following sub-sections explain the transformations done on various features.

3.3.1 Handling Missing Values

A machine learning model works on finding a weight for each feature in the dataset using mathematical formulas; hence, it is not acceptable to have data with missing values for the computation. Generally, if there is a large dataset with countable missing values, we can remove this data from our dataset and continue with the machine learning computations. However, in most of the cases, we need to fill in these missing values using some kind of algorithm. The process of filling in missing data is called imputation. Table 3.4 shows the count of missing values for features in our dataset.

Table 3.4: Count of missing values for features

Feature	Number of missing values
Core_High_School_GPA	47
Unweighted_High_School_GPA	23
Weighted_High_School_GPA	22

3.3.1.1 Imputing all High_School_GPAs

Table 3.4 shows the total number of missing values for each feature. We have a total of 508 records in first-year retention and 319 records in second-year retention datasets. Table 3.5 and 3.6 show the impact of missing values on output variables in first-year retention and second-year retention. The impact of the missing values of high school GPAs on retention is very limited. More than 1/3 of the missing value population were retained in CS and at UNLV in their first year. Moreover, in the second year, more than 2/3 of the missing value population was retained in CS and at UNLV in their second year. Looking at the impact, we used k-nearest neighbors (KNN) imputation with k=10 to fill all of the missing values in high school GPAs.

Table 3.5: Missing value impact of high school GPAs on first-year retention

	Missing?	Count of records	F2_Ret_CS	F2_Ret_UNLV
Core_High_School_GPA	No	461	287	363
	Yes	47	22	28
Unweighted_High_School_GPA	No	485	301	381
	Yes	23	8	10
Weighted_High_School_GPA	No	486	302	382
	Yes	22	7	9

Table 3.6: Missing value impact of high school GPAs on second-year retention

	Missing?	Count of records	F2_Ret_CS	F2_Ret_UNLV
Core_High_School_GPA	No	296	211	245
	Yes	23	15	19
Unweighted_High_School_GPA	No	311	220	257
	Yes	8	6	7
Weighted_High_School_GPA	No	312	221	258
	Yes	7	5	6

3.3.1.2 Imputing and Standardizing Grade Feature

We collected students' grades for all the courses they took in their first and second years. These grades range from A to D-, and S, AD, I, U, F, and W. It is not necessary that each student to take all courses in their first and second years. We collected data on 286 first-year and 352 second-year courses. There were many null values in each column. To overcome this issue, we converted all of the letter grades into numerical values, where A is the highest with 17, and null is the lowest with 0. Table 3.7 shows all of the possible grades in CS. After converting letter grades to numbers, we used a standardization technique to standardize the data in the range of 0 and 1. We used min-max standardization, defined in Equation 3.1.

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

Table 3.7: Grades to numerical conversion using min-max standardization

Grade	Numerical equivalent
A	17
A-	16
B+	15
B	14
B-	13
C+	12
C	11
C-	10
D+	9
D	8
D-	7
S	6
AD	5
I	4
U	3
F	2
W	1
Null (NaN)	0

3.3.1.3 Converting Categorical Features

In our dataset, we have four features with binary categorical values (‘Y’ and ‘N’). We converted all of these features into numerical vectors by assigning 1 to ‘Y’ and 0 to ‘N’. All other categorical features that have more than two values were converted using the one-hot-encoding technique which results in a one-hot vector. Each value in the feature will have one non-zero element in that vector. There are three such categorical features, which were encoded to find 17 new features.

3.4 Data Visualization and Analysis

To extract the important information from the data, we used several data visualization techniques. We found some important patterns in the data with the help of various plots.

3.4.1 Application_Term vs Retention

To understand how retention and dropout counts are distributed in each application term (cohort year), we plotted a line graph, shown in Figures 3.2 and 3.3. Figure 3.2 shows that for first-year, retention in both the CS and at UNLV steadily decreases compared to the total enrollment. However, the non-retention count increases with total enrollment. In the second-year retention (Figure 3.3), the count of non-returning students has increased in recent years.

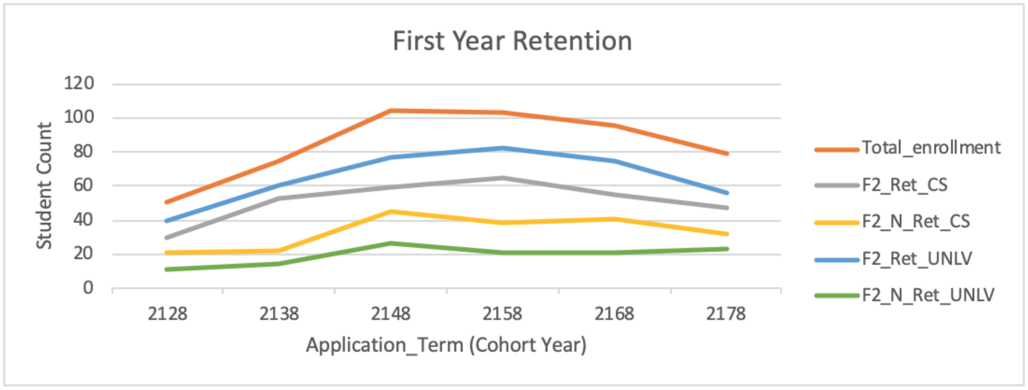


Figure 3.2: First year retention over application terms

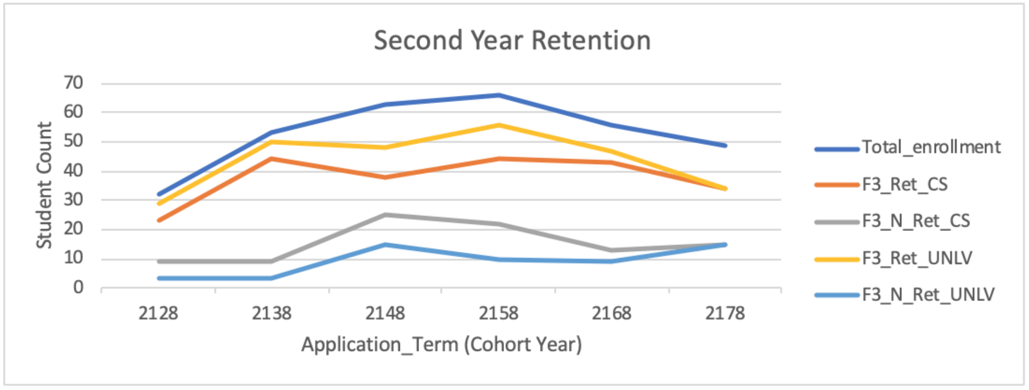


Figure 3.3: Second year retention over application terms

3.4.2 IPEDS_Race_Ethnicity vs Retention

We looked into the impact of ethnicity on CS enrollment and retention. We plotted pie charts to see the diversity of student population in CS (Figure 3.4 and 3.5). We found that more than 75% of enrolled students in CS are Asian, Hispanic, or White, in both first and second years. Later, we plotted a bar chart for first-year and second-year retention in CS and UNLV. The plots are shown in Figures 3.6 and 3.7. The plots show that, retention is highest among Asian students in both CS and UNLV, followed by Hispanic and White students. The plots also show that, switching majors within UNLV is highest among White students. We included ethnicity in our machine learning model to analyze its correlation with dropout.

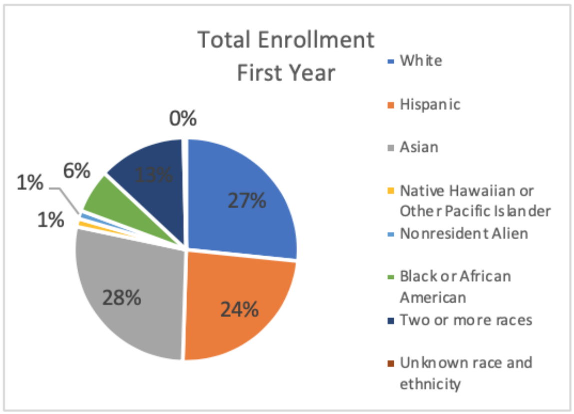


Figure 3.4: Pie chat of first year on ethnicity

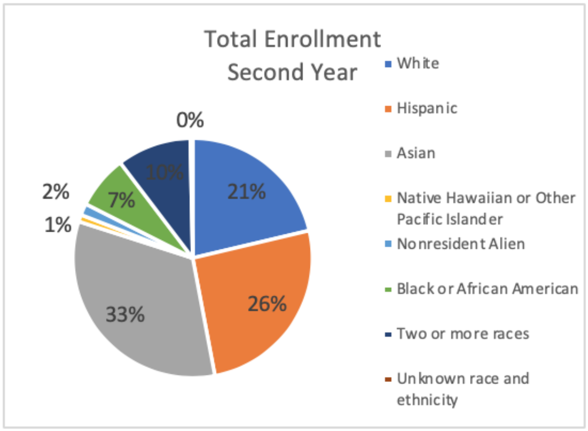


Figure 3.5: Pie chat of second year on ethnicity

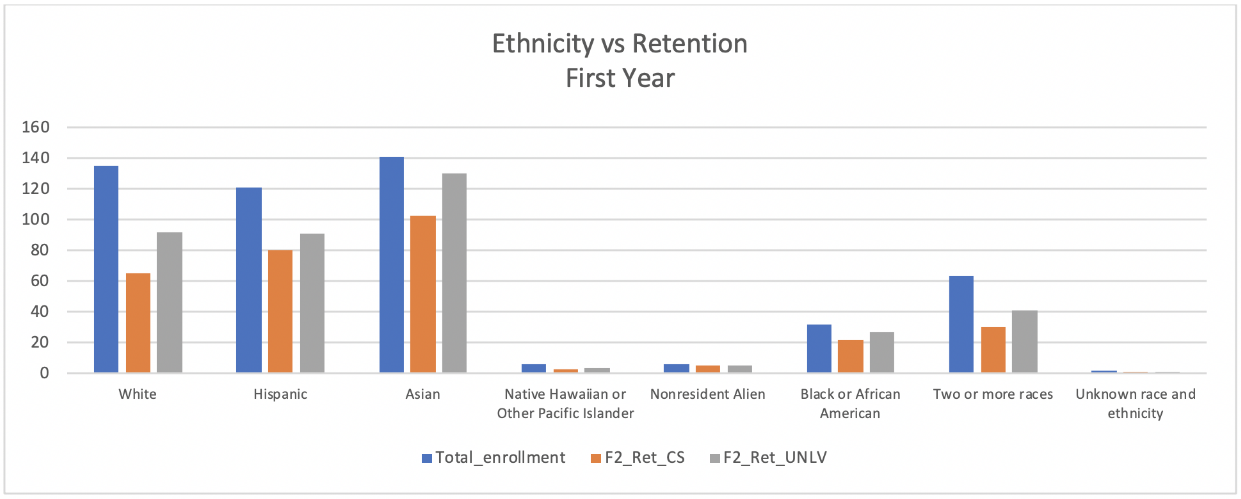


Figure 3.6: Count plot of retention on ethnicity in first year

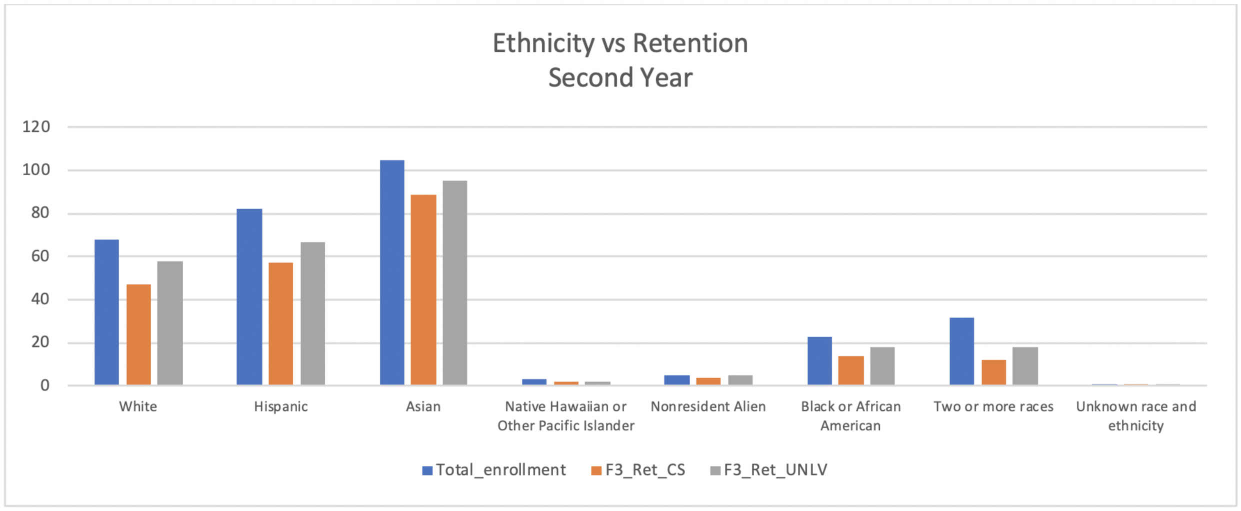


Figure 3.7: Count plot of retention on ethnicity in second year

3.5 Feature Importance

Based on the input and output feature types, there are three different ways to determine the importance of the input features for the output features:

Correlation: In this test, we use the Pearson correlation coefficient to find the relationship between two features. If the correlation coefficient is high, then the correlation of that feature with the output is high. We can perform a correlation test when the features are quantitative and not categorical.

Analysis Of Variance (ANOVA): ANOVA does the analysis of variance. This test can be performed on datasets, in which the input features are categorical and the output is quantitative.

Chi-Square: A chi-square test (x^2 test) is used to find the difference between expected and observed frequencies in one or more features. This test can be performed on categorical and quantitative features, in which the output can be both categorical or quantitative.

Our datasets has both categorical and quantitative features in them. To find the importance of features, we used the chi-square test. Using the chi-square test we found that, for both first and second year, the number of credits enrolled during the spring term is the most important feature for retention. Tables 3.8 and 3.9 shows the top 20 features for each output label in first-year and second-year retention datasets, based on the chi-square test scores. To indicate positive or negative relationship between features and output, we included the weights found by logistic regression model.

Table 3.8: Top 20 features in first-year found by chi-square test

F2_Ret_CS			F2_Ret_UNLV		
Feature	Score	LR Weights	Feature	Score	LR Weights
S1_Crd_Enrl	156.14	0.01212595	S1_Crd_Enrl	320.8	0.10128638
S1_GPA	77.93	0.266302692	S1_GPA	125.3	0.290619175
Cum_GPA	76.54	0.477541192	Cum_GPA	121.7	0.722194521
F1_GPA	43.96	0.196228609	F1_GPA	63.8	0.400289889
S1_Ret_CS	34.97	1.845044632	S1_COM_101_Grd	22.44	0.828191364
S1_CPE_100_Grd	16.52	0.734318167	S1_Ret_CS	20.99	0.146189931
S1_CS_135_Grd	15.57	0.637061098	IPEDS_Race_Ethnicity_Asian	18.45	0.797933375
S1_COM_101_Grd	15.14	0.306787624	S1_CPE_100_Grd	17.77	0.671842607
S1_CS_202_Grd	12.39	0.023027528	S1_ENG_102_Grd	17.12	0.618179634
S1_MATH_251_Grd	11.87	0.4516521	F1_Crd_Enrl	16.15	-0.114354731
F1_MATH_182_Grd	11.06	0.390913769	F1_EGG_101_Grd	15.82	0.278939903

Continued on next page

Table 3.8 – continued from previous page

F2_Ret_CS			F2_Ret_UNLV		
S1_ENG_102_Grd	10.58	0.146807246	S1_Ret_UNLV	14.48	-0.245841558
F1_CPE_100_Grd	9.44	0.874711176	S1_CS_202_Grd	12.22	0.229536758
IPEDS_Race_Ethnicity_ White	9.11	-0.646481376	S1_CS_135_Grd	12.12	0.357082161
F1_EGG_101_Grd	9.06	0.158473483	S1_MATH_181_Grd	11.03	0.598154507
IPEDS_Race_Ethnicity_ Asian	8.84	-0.016925165	S1_CPE_100L_Grd	10.03	0.19072058
S1_PHIL_114_Grd	8.66	0.609011196	F1_CS_135_Grd	8.96	0.423598765
F1_Crd_Enrl	8.36	-0.084967951	F1_SOC_101_Grd	8.73	0.282095947
S1_CPE_100L_Grd	7.93	-0.197442212	S1_HIST_101_Grd	8.64	0.734062171
S1_MATH_181_Grd	6.85	0.268284628	F1_ANTH_105_Grd	7.67	-0.231693493

Table 3.9: Top 20 features in second-year found by chi-square test

F3_Ret_CS			F3_Ret_UNLV		
Feature	Score	LR Weights	Feature	Score	LR Weights
S2_Crd_Enrl	171.16	0.047359625	S2_Crd_Enrl	266.23	-0.003080136
S2_GPA	48.18	1.170845081	S2_GPA	75.12	1.789928694
F2_Crd_Enrl	41.34	0.029942285	Cum_GPA	50.56	1.793225308
Cum_GPA	34.15	-0.324440599	F2_Crd_Enrl	47.26	0.230599644
S1_GPA	18.46	-0.013499539	F2_GPA	23.21	-0.441421346
F2_GPA	18.31	-0.261929578	S1_GPA	17.26	0.086069566
S2_Ret_CS	17.44	5.003769296	IPEDS_Race_Ethnicity_ Two or more races	15.76	-1.408365223
IPEDS_Race_Ethnicity_ Two or more races	17.23	-0.816001741	S2_Ret_UNLV	10.93	-0.243613014
S2_CS_302_Grd	14.32	0.377876059	S2_Ret_CS	10.2	1.376091587

Continued on next page

Table 3.9 – continued from previous page

F3_Ret_CS			F3_Ret_UNLV		
S2_CS_219_Grd	11.93	1.151935156	F2_CHEM_103_Grd	8.47	-1.083118449
S2_CS_218_Grd	10.09	1.593010445	F1_GPA	7.14	-0.355147827
IPEDS_Race_Ethnicity_Asian	9.84	0.976450155	S2_CS_302_Grd	6.2	-0.366364378
S2_GEOL_101_Grd	9.39	2.12165791	Application_Term_2178	6.14	-1.737736503
S2_PHIL_242_Grd	9.38	1.806289092	S2_CS_219_Grd	6.04	0.29642137
F2_CS_218_Grd	9.22	0.491283485	F2_CS_218_Grd	5.71	0.451189144
S2_CPE_100L_Grd	8.86	1.74961295	S2_CS_218_Grd	5.6	1.034559859
F2_MATH_251_Grd	8.01	0.643828803	S2_COM_101_Grd	5.14	1.04100901
F2_PHIL_114_Grd	7.16	-0.163351624	Application_Term_2138	4.98	1.203883942
F2_CPE_100L_Grd	6.75	-0.172285276	F2_COM_216_Grd	4.8	-1.045223443
S2_MATH_251_Grd	6.74	0.099316903	F2_DAN_301_Grd	4.8	-1.566925705

Chapter 4

Experiments and Results

4.1 Data Splitting

Once the datasets were created, there were around 320 independent features in first-year retention dataset, excluding output labels, and 387 independent features in second-year retention dataset. We used 80-20 split on these datasets to make training and test data. We used training data to create validation data using the 80-20 split.

4.2 Experiments with Machine Learning Models

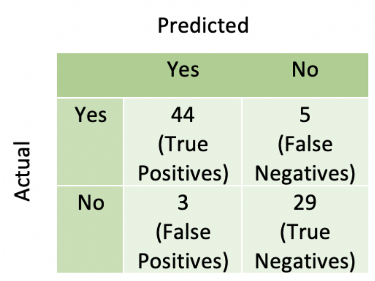
After creating training, validation, and test data from the entire dataset, each of the selected models were trained by passing training data and evaluated using the validation data. We trained three models: feedforward neural network, logistic regression, and support vector machine and generated confusion matrix on the validation and test data, which then were used to calculate classification accuracy, recall, precision, specificity, F_1 score, fallout, AUC and ROC curve of the model. We used k-fold cross-validation to generate the average results of AUC.

4.2.1 Feedforward Neural Network

A feedforward neural network was designed using Tensorflow and Keras packages in Python. We used these designed models to fit our training data and the models were used to make predictions on validation and test data. We designed two different models to predict the first-year and the second-year retention. All of the evaluation metrics were calculated separately for each of the models. The selected model has one input layer, one hidden layer, and one output layer.

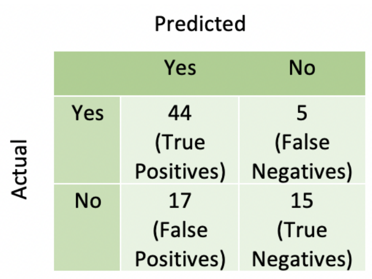
4.2.1.1 First-Year Retention for F2_ret_CS using FNN Model

The confusion matrix generated on validation data is shown in Figure 4.1. Specificity calculated using this confusion matrix shows that without oversampling output labels, FNN model gave good results predicting output for label F2_ret_CS.



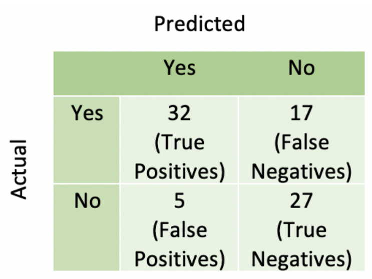
Actual \ Predicted	Yes	No
Yes	44 (True Positives)	5 (False Negatives)
No	3 (False Positives)	29 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	44 (True Positives)	5 (False Negatives)
No	17 (False Positives)	15 (True Negatives)

(b) Oversampling F2_ret_CS

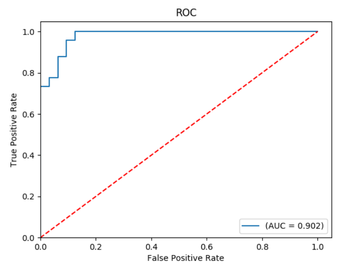


Actual \ Predicted	Yes	No
Yes	32 (True Positives)	17 (False Negatives)
No	5 (False Positives)	27 (True Negatives)

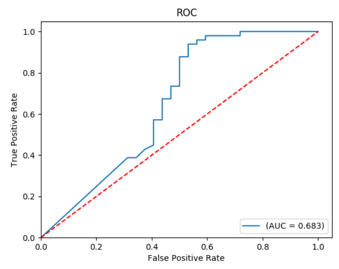
(c) Oversampling F2_ret_UNLV

Figure 4.1: Confusion matrix of FNN model on validation data (first-year retention for F2_ret_CS)

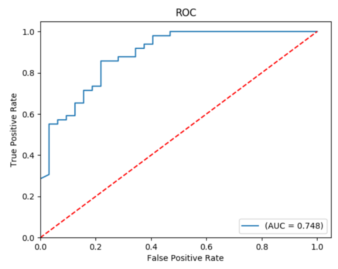
The ROC curves generated on validation data in Figure 4.2 also show that without oversampling we got AUC score of 0.902 which is closer to AUC score of 1.000 found by k-fold cross-validation using k=10 for output label F2_ret_CS.



(a) Without oversampling



(b) Oversampling F2_ret_CS



(c) Oversampling F2_ret_UNLV

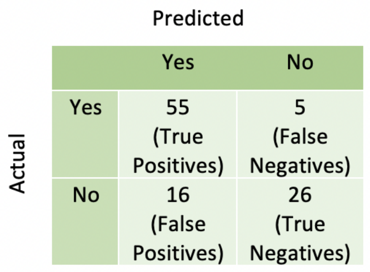
Figure 4.2: ROC curves of FNN model on validation data (first-year retention for F2_ret_CS)

All other metrics calculated using confusion matrix are shown in Table 4.1. Accuracy, specificity, and AUC results show that FNN model without oversampling outperformed oversampled results.

Table 4.1: Results of FNN model on Validation data (first-year retention for F2_ret_CS)

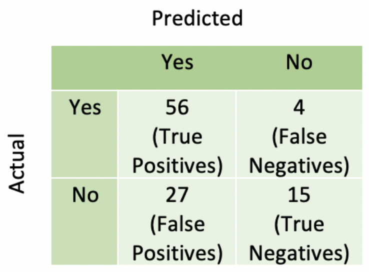
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.898	0.936	0.906	0.917	0.094	0.902	0.901
F2_ret_CS	0.898	0.721	0.469	0.800	0.531	0.683	0.728
F2_ret_UNLV	0.796	0.672	0.406	0.729	0.594	0.628	0.642

The confusion matrix generated on test data is shown in Figure 4.3. Specificity calculated using this confusion matrix shows that, without oversampling output labels, FNN model gave good results predicting output for label F2_ret_CS.



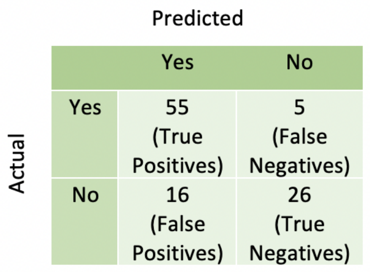
		Predicted	
		Yes	No
Actual	Yes	55 (True Positives)	5 (False Negatives)
	No	16 (False Positives)	26 (True Negatives)

(a) Without oversampling



		Predicted	
		Yes	No
Actual	Yes	56 (True Positives)	4 (False Negatives)
	No	27 (False Positives)	15 (True Negatives)

(b) Oversampling F2_ret_CS

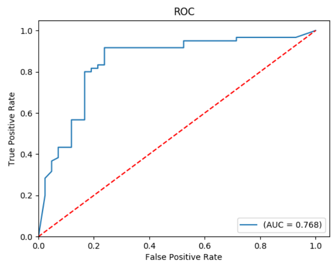


		Predicted	
		Yes	No
Actual	Yes	55 (True Positives)	5 (False Negatives)
	No	16 (False Positives)	26 (True Negatives)

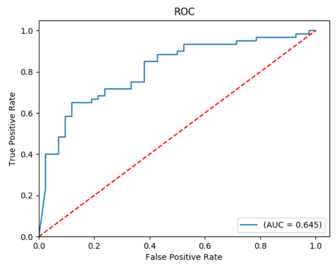
(c) Oversampling F2_ret_UNLV

Figure 4.3: Confusion matrix of FNN model on test data (first-year retention for F2_ret_CS)

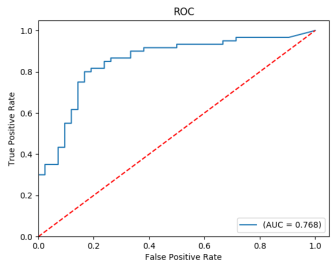
The ROC curves generated on test data in Figure 4.4 also show that without oversampling we found AUC score of 0.768 which is better than or equal to the AUC score found by oversampled results for output label F2_ret_CS.



(a) Without oversampling



(b) Oversampling F2_ret_CS



(c) Oversampling F2_ret_UNLV

Figure 4.4: ROC curves of FNN model on validation data (first-year retention for F2_ret_CS)

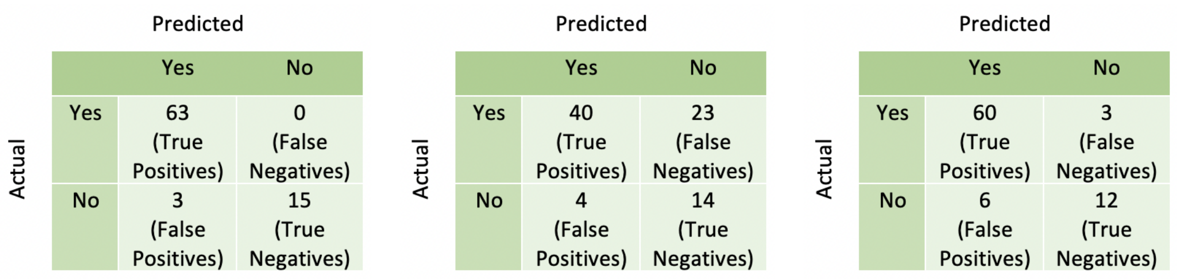
All other metrics calculated using confusion matrix (Figure 4.3) are shown in Table 4.3. Accuracy, specificity, and AUC results show that FNN model without oversampling outperformed oversampled results.

Table 4.2: Results of FNN model on test data (first-year retention for F2_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.917	0.775	0.619	0.840	0.381	0.768	0.794
F2_ret_CS	0.933	0.675	0.357	0.783	0.643	0.645	0.696
F2_ret_UNLV	0.917	0.775	0.619	0.840	0.381	0.768	0.794

4.2.1.2 First-Year Retention for F2_ret_UNLV using FNN Model

The confusion matrix generated on validation data is shown in Figure 4.5. Specificity calculated using this confusion matrix shows that without oversampling output labels, FNN model gave good results predicting output for label F2_ret_UNLV.



(a) Without oversampling

Actual \ Predicted	Yes	No
Yes	63 (True Positives)	0 (False Negatives)
No	3 (False Positives)	15 (True Negatives)

(b) Oversampling F2_ret_CS

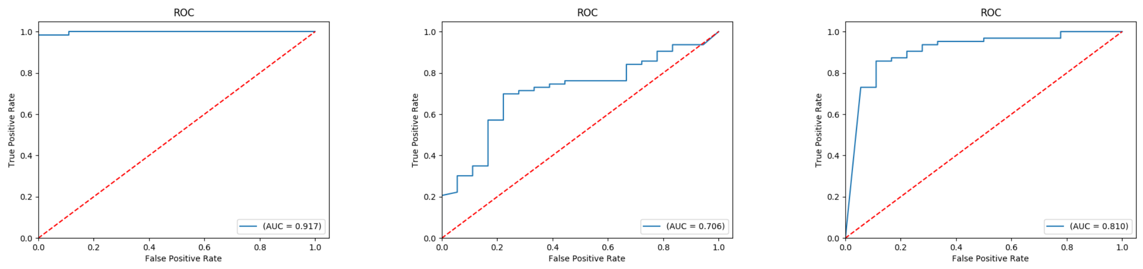
Actual \ Predicted	Yes	No
Yes	40 (True Positives)	23 (False Negatives)
No	4 (False Positives)	14 (True Negatives)

(c) Oversampling F2_ret_UNLV

Actual \ Predicted	Yes	No
Yes	60 (True Positives)	3 (False Negatives)
No	6 (False Positives)	12 (True Negatives)

Figure 4.5: Confusion matrix of FNN model on validation data (first-year retention for F2_ret_UNLV)

The ROC curves generated on validation data in Figure 4.6 also show that without oversampling we got AUC score of 0.902 which is closer to AUC score of 1.000 found by k-fold cross-validation using k=10 for output label F2_ret_UNLV.



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

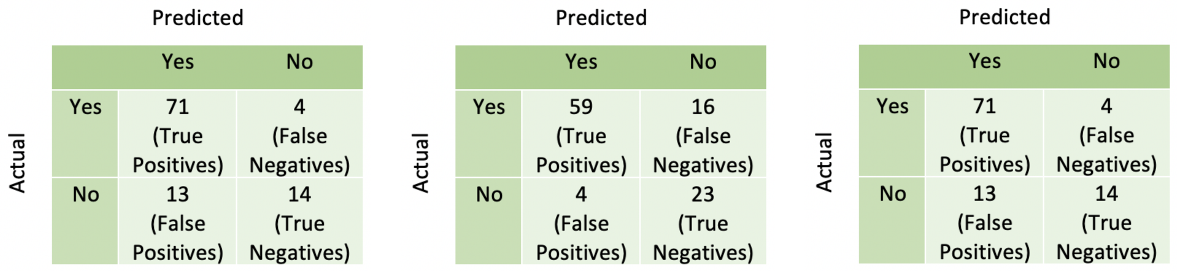
Figure 4.6: ROC curves of FNN model on validation data (first-year retention for F2_ret_UNLV)

All other metrics calculated using confusion matrix are shown in Table 4.3. Accuracy, specificity, and AUC results show that FNN model without oversampling outperformed oversampled results.

Table 4.3: Results of FNN model on Validation data (first-year retention for F2_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	1.000	0.954	0.833	0.977	0.167	0.917	0.963
F2_ret_CS	0.635	0.909	0.778	0.748	0.222	0.706	0.667
F2_ret_UNLV	0.952	0.909	0.667	0.930	0.333	0.809	0.889

The confusion matrix generated on test data is shown in Figure 4.7. Specificity calculated using this confusion matrix shows that, without oversampling output labels, FNN model gave good results predicting output for label F2_ret_UNLV.



(a)

		Predicted	
		Yes	No
Actual	Yes	71 (True Positives)	4 (False Negatives)
	No	13 (False Positives)	14 (True Negatives)

(b)

		Predicted	
		Yes	No
Actual	Yes	59 (True Positives)	16 (False Negatives)
	No	4 (False Positives)	23 (True Negatives)

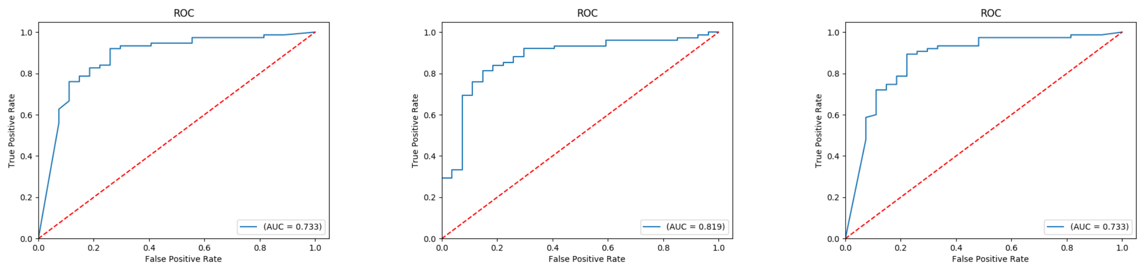
(c)

		Predicted	
		Yes	No
Actual	Yes	71 (True Positives)	4 (False Negatives)
	No	13 (False Positives)	14 (True Negatives)

(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

Figure 4.7: Confusion matrix of FNN model on test data (first-year retention for F2_ret_UNLV)

The ROC curves generated on test data in Figure 4.8 also show that without oversampling we found AUC score of 0.768 which is better than or equal to the AUC score found by oversampled results for output label F2_ret_UNLV.



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

Figure 4.8: ROC curves of FNN model on validation data (first-year retention for F2_ret_UNLV)

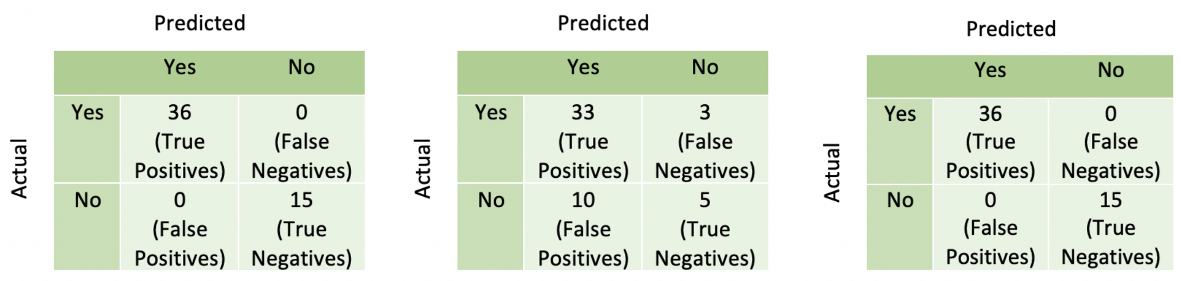
All other metrics calculated using confusion matrix (Figure 4.7) are shown in Table 4.4. Accuracy, specificity, and AUC results show that FNN model without oversampling outperformed oversampled results.

Table 4.4: Results of FNN model on test data (first-year retention for F2_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.947	0.845	0.518	0.893	0.481	0.733	0.833
F2_ret_CS	0.787	0.936	0.852	0.855	0.148	0.819	0.804
F2_ret_UNLV	0.947	0.845	0.518	0.893	0.481	0.733	0.833

4.2.1.3 Second-Year Retention for F3_ret_CS using FNN Model

The confusion matrix generated on validation data is shown in Figure 4.9. Specificity calculated using this confusion matrix shows that without oversampling output labels, FNN model gave good results predicting output for label F3_ret_CS.



Predicted / Actual

		Yes	No
Actual	Yes	36 (True Positives)	0 (False Negatives)
	No	0 (False Positives)	15 (True Negatives)

(a) Without oversampling

		Yes	No
Actual	Yes	33 (True Positives)	3 (False Negatives)
	No	10 (False Positives)	5 (True Negatives)

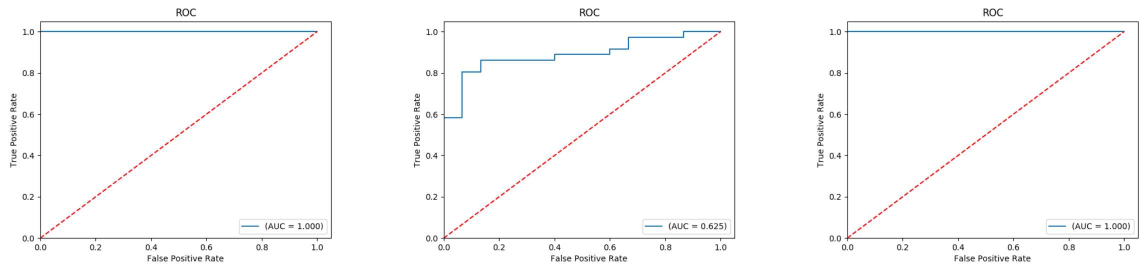
(b) Oversampling F3_ret_CS

		Yes	No
Actual	Yes	36 (True Positives)	0 (False Negatives)
	No	0 (False Positives)	15 (True Negatives)

(c) Oversampling F3_ret_UNLV

Figure 4.9: Confusion matrix of FNN model on validation data (second-year retention for F3_ret_CS)

The ROC curves generated on validation data in Figure 4.10 also show that without oversampling we got AUC score of 1.000 which is equal to AUC score of 1.000 found by k-fold cross-validation using k=10 for output label F3_ret_CS.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

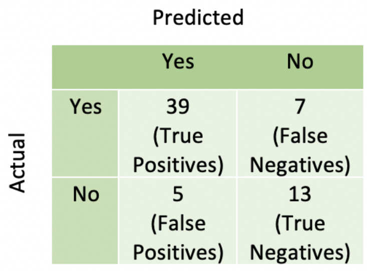
Figure 4.10: ROC curves of FNN model on validation data (second-year retention for F3_ret_CS)

All other metrics calculated using confusion matrix are shown in Table 4.5. Accuracy, specificity, and AUC results show that FNN model without oversampling performed similar to other results.

Table 4.5: Results of FNN model on Validation data (second-year retention for F3_ret_CS)

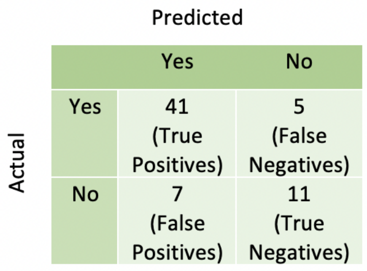
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	1.000	1.000	1.000	1.000	0.000	1.000	1.000
F3_ret_CS	0.917	0.767	0.333	0.835	0.667	0.625	0.745
F3_ret_UNLV	1.000	1.000	1.000	1.000	0.000	1.000	1.000

The confusion matrix generated on test data is shown in Figure 4.11. Specificity calculated using this confusion matrix shows that, without oversampling output labels, FNN model gave good results predicting output for label F3_ret_CS.



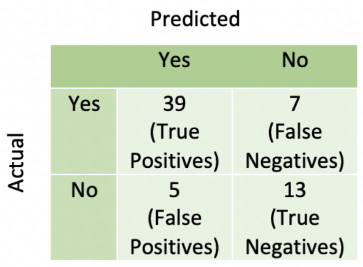
Actual \ Predicted	Yes	No
Yes	39 (True Positives)	7 (False Negatives)
No	5 (False Positives)	13 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	41 (True Positives)	5 (False Negatives)
No	7 (False Positives)	11 (True Negatives)

(b) Oversampling F3_ret_CS

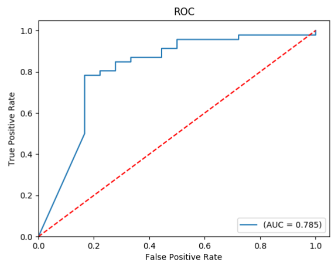


Actual \ Predicted	Yes	No
Yes	39 (True Positives)	7 (False Negatives)
No	5 (False Positives)	13 (True Negatives)

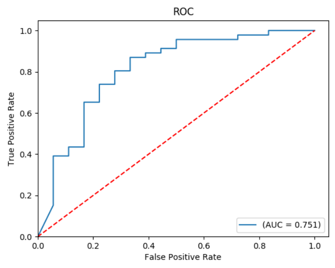
(c) Oversampling F3_ret_UNLV

Figure 4.11: Confusion matrix of FNN model on test data (second-year retention for F3_ret_CS)

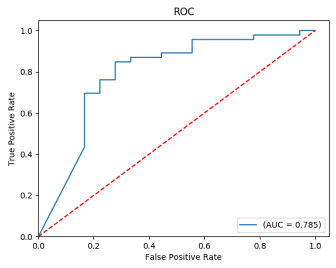
The ROC curves generated on test data in Figure 4.12 also show that without oversampling we found AUC score of 0.785 which is better than or equal to the AUC score found by oversampled results for output label F3_ret_CS.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

Figure 4.12: ROC curves of FNN model on validation data (second-year retention for F3_ret_CS)

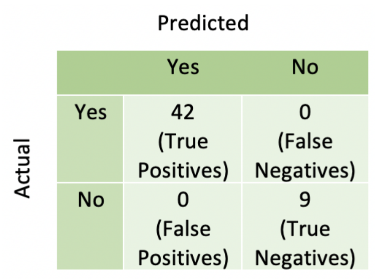
All other metrics calculated using confusion matrix (Figure 4.11) are shown in Table 4.6. Accuracy, specificity, and AUC results show that FNN model without oversampling performed similar to other results.

Table 4.6: Results of FNN model on test data (second-year retention for F3_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.848	0.886	0.722	0.867	0.278	0.785	0.812
F3_ret_CS	0.891	0.854	0.611	0.872	0.389	0.751	0.812
F3_ret_UNLV	0.848	0.886	0.722	0.867	0.278	0.785	0.812

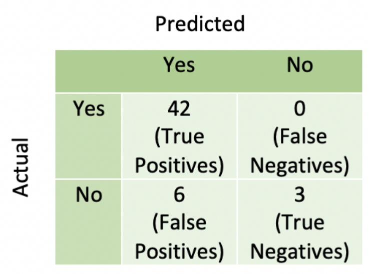
4.2.1.4 Second-Year Retention for F3_ret_UNLV using FNN Model

The confusion matrix generated on validation data is shown in Figure 4.13. Specificity calculated using this confusion matrix shows that without oversampling output labels, FNN model gave good results predicting output for label F3_ret_UNLV.



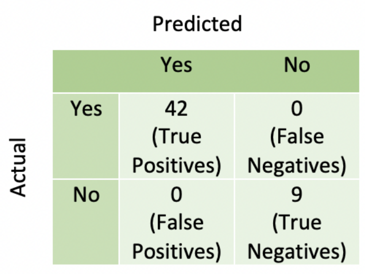
		Predicted	
		Yes	No
Actual	Yes	42 (True Positives)	0 (False Negatives)
	No	0 (False Positives)	9 (True Negatives)

(a) Without oversampling



		Predicted	
		Yes	No
Actual	Yes	42 (True Positives)	0 (False Negatives)
	No	6 (False Positives)	3 (True Negatives)

(b) Oversampling F3_ret_CS

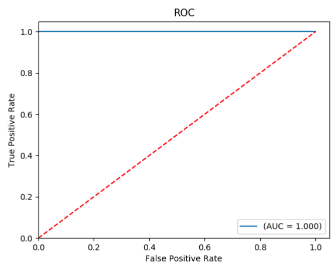


		Predicted	
		Yes	No
Actual	Yes	42 (True Positives)	0 (False Negatives)
	No	0 (False Positives)	9 (True Negatives)

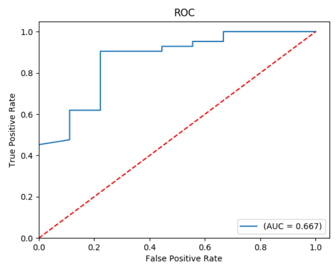
(c) Oversampling F3_ret_UNLV

Figure 4.13: Confusion matrix of FNN model on validation data (second-year retention for F3_ret_UNLV)

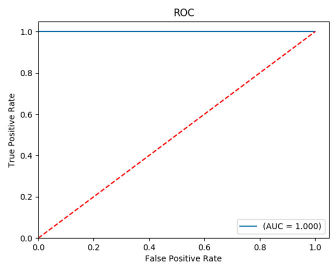
The ROC curves generated on validation data in Figure 4.14 also show that without oversampling we got AUC score of 1.000 which is closer to AUC score of 1.000 found by k-fold cross-validation using k=10 for output label F3_ret_UNLV.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

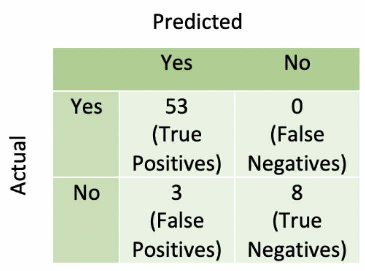
Figure 4.14: ROC curves of FNN model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix are shown in Table 4.7. Accuracy, specificity, and AUC results show that FNN model without oversampling performed similar to other results.

Table 4.7: Results of FNN model on Validation data (second-year retention for F3_ret_UNLV)

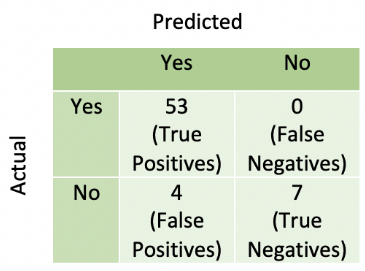
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	1.000	1.000	1.000	1.000	0.000	1.000	1.000
F3_ret_CS	1.000	0.875	0.333	0.933	0.667	0.667	0.882
F3_ret_UNLV	1.000	1.000	1.000	1.000	0.000	1.000	1.000

The confusion matrix generated on test data is shown in Figure 4.15. Specificity calculated using this confusion matrix shows that without oversampling output labels, FNN model gave good results predicting output for label F3_ret_UNLV.



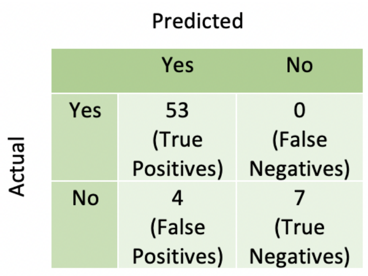
Actual \ Predicted	Yes	No
Yes	53 (True Positives)	0 (False Negatives)
No	3 (False Positives)	8 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	53 (True Positives)	0 (False Negatives)
No	4 (False Positives)	7 (True Negatives)

(b) Oversampling F3_ret_CS

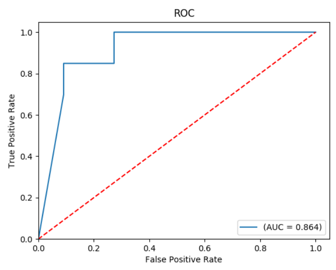


Actual \ Predicted	Yes	No
Yes	53 (True Positives)	0 (False Negatives)
No	4 (False Positives)	7 (True Negatives)

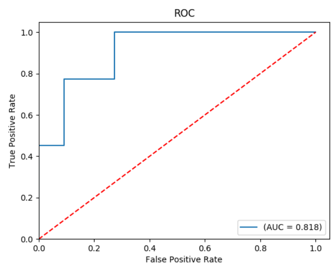
(c) Oversampling F3_ret_UNLV

Figure 4.15: Confusion matrix of FNN model on test data (second-year retention for F3_ret_UNLV)

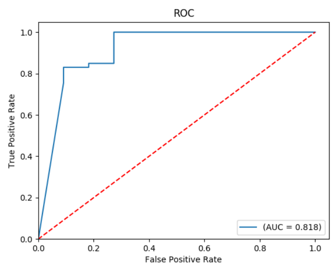
The ROC curves generated on test data in Figure 4.16 also show that without oversampling output labels we found AUC score of 0.864 which is better than or equal to the AUC score found by other results for output label F3_ret_UNLV.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

Figure 4.16: ROC curves of FNN model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix (Figure 4.15) are shown in Table 4.8. Accuracy, specificity, and AUC results show that FNN model without oversampling outperformed oversampled results.

Table 4.8: Results of FNN model on test data (second-year retention for F3_ret_UNLV)

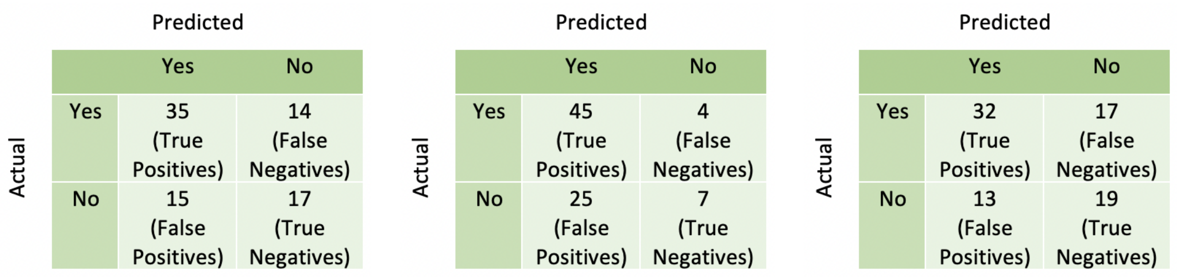
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	1.000	0.946	0.727	0.972	0.273	0.864	0.953
F3_ret_CS	1.000	0.930	0.636	0.964	0.364	0.818	0.937
F3_ret_UNLV	1.000	0.930	0.636	0.964	0.364	0.818	0.937

4.2.2 Logistic Regression

A logistic regression model was designed using the sklearn package in Python. We used these designed models to fit our training data. The models were used to make predictions on validation and test data. We designed two different models to predict the first-year and the second-year retention. All of the evaluation metrics were calculated separately for each of the models.

4.2.2.1 First-Year Retention for F2_ret_CS using LR Model

The confusion matrix generated on validation data is shown in Figure 4.17. Specificity calculated using this confusion matrix shows that without oversampling output labels, LR model gave good results predicting output for label F2_ret_CS compare to oversampled results.



(a) Without oversampling

Actual \ Predicted	Yes	No
Yes	35 (True Positives)	14 (False Negatives)
No	15 (False Positives)	17 (True Negatives)

(b) Oversampling F2_ret_CS

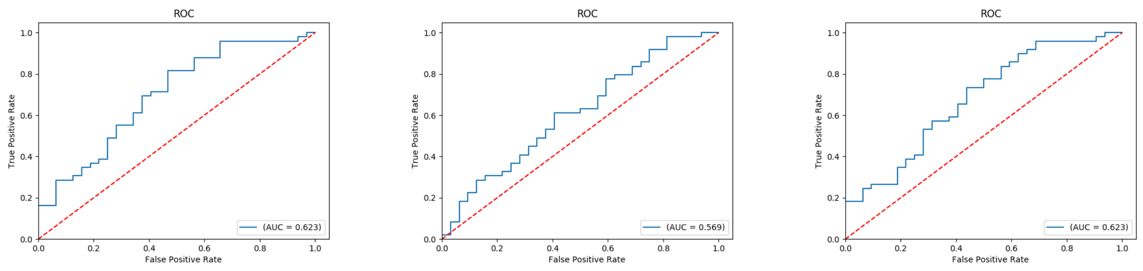
Actual \ Predicted	Yes	No
Yes	45 (True Positives)	4 (False Negatives)
No	25 (False Positives)	7 (True Negatives)

(c) Oversampling F2_ret_UNLV

Actual \ Predicted	Yes	No
Yes	32 (True Positives)	17 (False Negatives)
No	13 (False Positives)	19 (True Negatives)

Figure 4.17: Confusion matrix of LR model on validation data (first-year retention for F2_ret_CS)

The ROC curves generated on validation data in Figure 4.18 also show that without oversampling we got AUC score of 0.623 which is nearly equal to AUC score of 0.747 found by k-fold cross-validation using k=10 for output label F2_ret_CS.



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

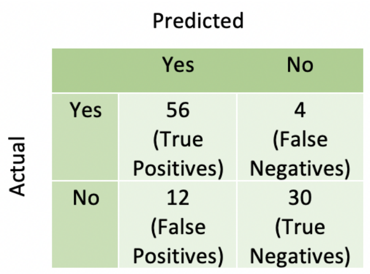
Figure 4.18: ROC curves of LR model on validation data (first-year retention for F2_ret_CS)

All other metrics calculated using confusion matrix are shown in Table 4.9. Accuracy, specificity, and AUC results show that LR model without oversampling outperformed oversampled results.

Table 4.9: Results of LR model on Validation data (first-year retention for F2_ret_CS)

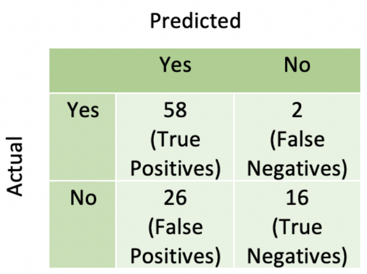
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.714	0.700	0.531	0.707	0.469	0.623	0.642
F2_ret_CS	0.918	0.643	0.219	0.756	0.781	0.569	0.642
F2_ret_UNLV	0.653	0.711	0.594	0.681	0.406	0.623	0.630

The confusion matrix generated on test data is shown in Figure 4.19. Specificity calculated using this confusion matrix shows that, without oversampling output labels, LR model gave good results predicting output for label F2_ret_CS.



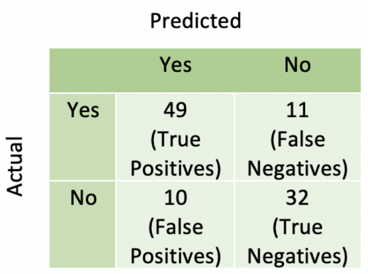
Actual \ Predicted	Yes	No
Yes	56 (True Positives)	4 (False Negatives)
No	12 (False Positives)	30 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	58 (True Positives)	2 (False Negatives)
No	26 (False Positives)	16 (True Negatives)

(b) Oversampling F2_ret_CS

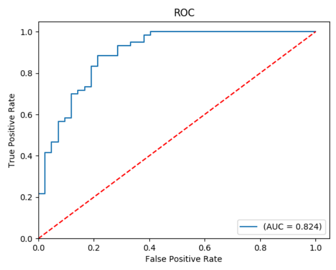


Actual \ Predicted	Yes	No
Yes	49 (True Positives)	11 (False Negatives)
No	10 (False Positives)	32 (True Negatives)

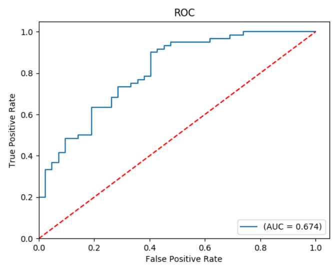
(c) Oversampling F2_ret_UNLV

Figure 4.19: Confusion matrix of LR model on test data (first-year retention for F2_ret_CS)

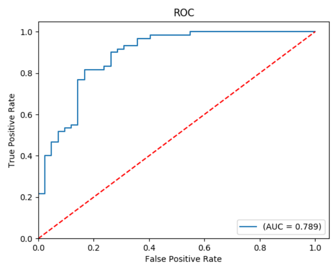
The ROC curves generated on test data in Figure 4.20 also show that without oversampling we found AUC score of 0.824 which is better than the AUC score found by oversampled results for output label F2_ret_CS.



(a) Without oversampling



(b) Oversampling F2_ret_CS



(c) Oversampling F2_ret_UNLV

Figure 4.20: ROC curves of LR model on validation data (first-year retention for F2_ret_CS)

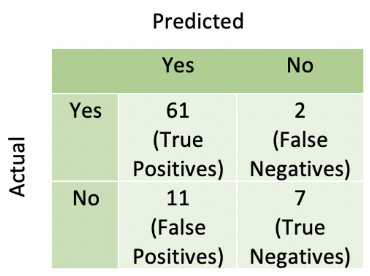
All other metrics calculated using confusion matrix (Figure 4.19) are shown in Table 4.10. Accuracy, specificity, and AUC results show that LR model without oversampling outperformed oversampled results.

Table 4.10: Results of LR model on test data (first-year retention for F2_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.933	0.823	0.714	0.875	0.286	0.824	0.843
F2_ret_CS	0.967	0.690	0.381	0.806	0.619	0.674	0.725
F2_ret_UNLV	0.817	0.830	0.762	0.823	0.238	0.789	0.794

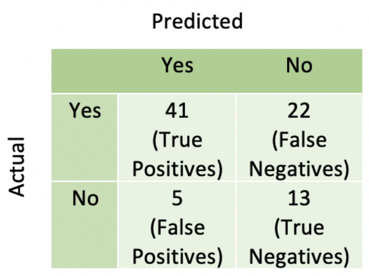
4.2.2.2 First-Year Retention for F2_ret_UNLV using LR Model

The confusion matrix generated on validation data is shown in Figure 4.21. Specificity calculated using this confusion matrix shows that LR model gave good results predicting output for label F2_ret_UNLV when output labels F2_ret_CS was oversampled.



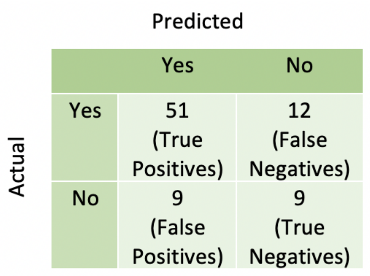
Actual \ Predicted	Yes	No
Yes	61 (True Positives)	2 (False Negatives)
No	11 (False Positives)	7 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	41 (True Positives)	22 (False Negatives)
No	5 (False Positives)	13 (True Negatives)

(b) Oversampling F2_ret_CS

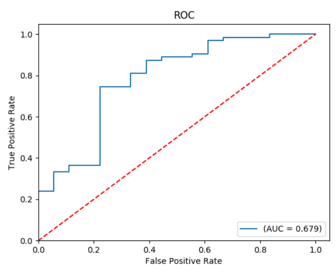


Actual \ Predicted	Yes	No
Yes	51 (True Positives)	12 (False Negatives)
No	9 (False Positives)	9 (True Negatives)

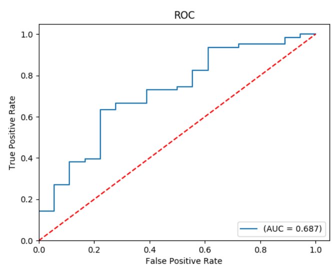
(c) Oversampling F2_ret_UNLV

Figure 4.21: Confusion matrix of LR model on validation data (first-year retention for F2_ret_UNLV)

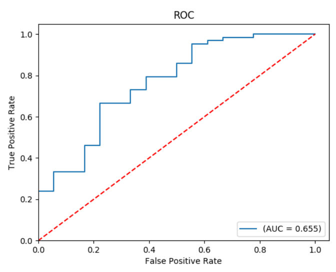
The ROC curves generated on validation data in Figure 4.22 also show that without oversampling we got AUC score of 0.679 which is closer to AUC score of 0.785 found by k-fold cross-validation using k=10 for output label F2_ret_UNLV.



(a) Without oversampling



(b) Oversampling F2_ret_CS



(c) Oversampling F2_ret_UNLV

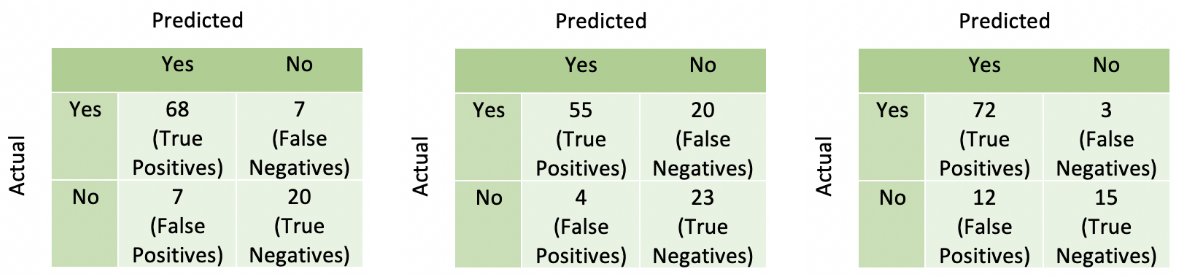
Figure 4.22: ROC curves of LR model on validation data (first-year retention for F2_ret_UNLV)

All other metrics calculated using confusion matrix are shown in Table 4.11. Accuracy, Recall, and AUC results show that LR model without oversampling outperformed oversampled results.

Table 4.11: Results of LR model on Validation data (first-year retention for F2_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.968	0.847	0.389	0.904	0.611	0.679	0.839
F2_ret_CS	0.651	0.891	0.722	0.752	0.278	0.686	0.667
F2_ret_UNLV	0.809	0.850	0.500	0.829	0.500	0.655	0.741

The confusion matrix generated on test data is shown in Figure 4.23. Specificity calculated using this confusion matrix shows that LR model gave good results predicting output for label F2_ret_UNLV without oversampling any output labels.



Actual \ Predicted	Yes	No
Yes	68 (True Positives)	7 (False Negatives)
No	7 (False Positives)	20 (True Negatives)

(a) Without oversampling

Actual \ Predicted	Yes	No
Yes	55 (True Positives)	20 (False Negatives)
No	4 (False Positives)	23 (True Negatives)

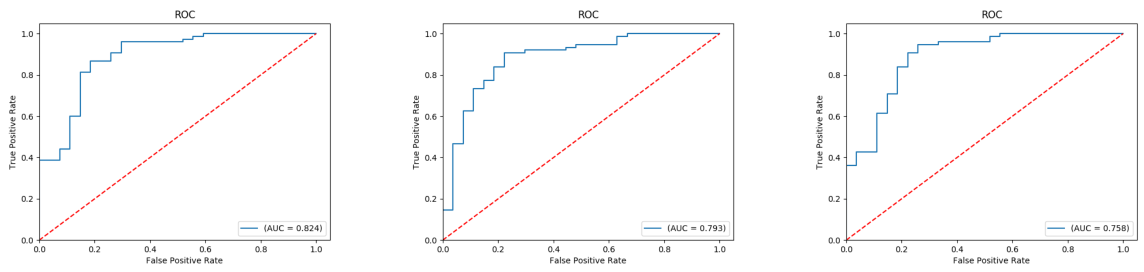
(b) Oversampling F2_ret_CS

Actual \ Predicted	Yes	No
Yes	72 (True Positives)	3 (False Negatives)
No	12 (False Positives)	15 (True Negatives)

(c) Oversampling F2_ret_UNLV

Figure 4.23: Confusion matrix of LR model on test data (first-year retention for F2_ret_UNLV)

The ROC curves generated on test data in Figure 4.24 also show that without oversampling, we found AUC score of 0.824 which is better than or equal to the AUC score found by other results for output label F2_ret_UNLV.



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

Figure 4.24: ROC curves of LR model on validation data (first-year retention for F2_ret_UNLV)

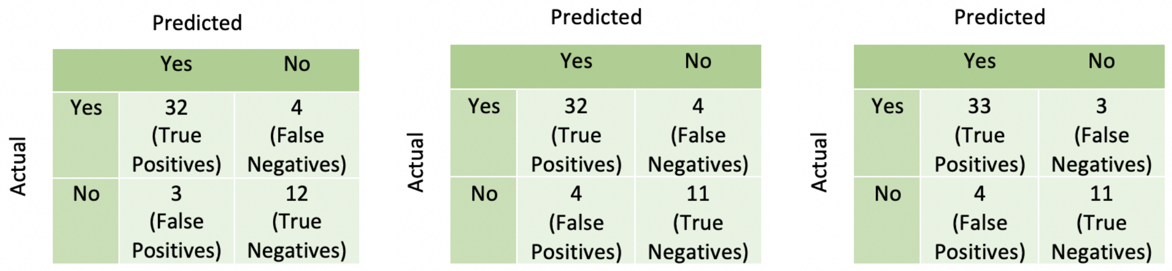
All other metrics calculated using confusion matrix (Figure 4.23) are shown in Table 4.12. Accuracy, specificity, and AUC results show that LR model without oversampling outperformed oversampled results.

Table 4.12: Results of LR model on test data (first-year retention for F2_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.907	0.907	0.741	0.907	0.259	0.824	0.863
F2_ret_CS	0.733	0.932	0.852	0.821	0.148	0.793	0.765
F2_ret_UNLV	0.960	0.857	0.556	0.906	0.444	0.758	0.853

4.2.2.3 Second-Year Retention for F3_ret_CS using LR Model

The confusion matrix generated on validation data is shown in Figure 4.25. Specificity calculated using this confusion matrix shows that without oversampling output labels, LR model gave good results predicting output for label F3_ret_CS.



(a) Without oversampling

Actual \ Predicted	Yes	No
Yes	32 (True Positives)	4 (False Negatives)
No	3 (False Positives)	12 (True Negatives)

(b) Oversampling F3_ret_CS

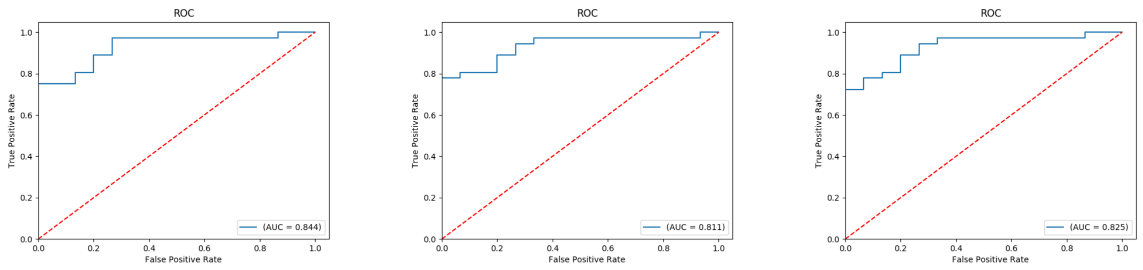
Actual \ Predicted	Yes	No
Yes	32 (True Positives)	4 (False Negatives)
No	4 (False Positives)	11 (True Negatives)

(c) Oversampling F3_ret_UNLV

Actual \ Predicted	Yes	No
Yes	33 (True Positives)	3 (False Negatives)
No	4 (False Positives)	11 (True Negatives)

Figure 4.25: Confusion matrix of LR model on validation data (second-year retention for F3_ret_CS)

The ROC curves generated on validation data in Figure 4.26 also show that without oversampling we got AUC score of 0.844 which is better than AUC score of 0.774 found by k-fold cross-validation using k=10 for output label F3_ret_CS.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

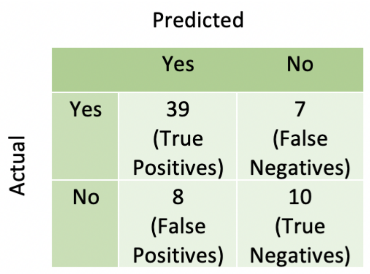
Figure 4.26: ROC curves of LR model on validation data (second-year retention for F3_ret_CS)

All other metrics calculated using confusion matrix are shown in Table 4.13. Accuracy, specificity, and AUC results show that LR model without oversampling performed similar to other results.

Table 4.13: Results of LR model on Validation data (second-year retention for F3_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.889	0.914	0.800	0.901	0.200	0.844	0.863
F3_ret_CS	0.889	0.889	0.733	0.889	0.267	0.811	0.843
F3_ret_UNLV	0.917	0.892	0.733	0.904	0.267	0.825	0.863

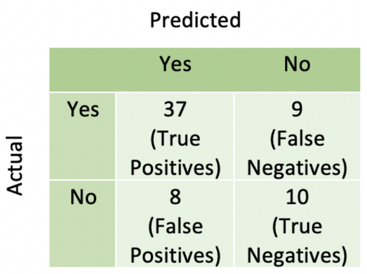
The confusion matrix generated on test data is shown in Figure 4.27. Specificity calculated using this confusion matrix shows that, without oversampling output labels, LR model gave good results predicting output for label F3_ret_CS.



Predicted

Actual	Yes	No
Yes	39 (True Positives)	7 (False Negatives)
No	8 (False Positives)	10 (True Negatives)

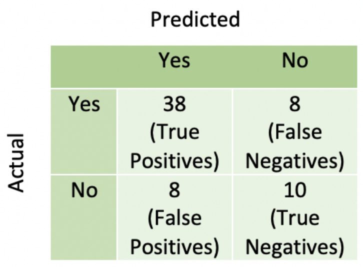
(a) Without oversampling



Predicted

Actual	Yes	No
Yes	37 (True Positives)	9 (False Negatives)
No	8 (False Positives)	10 (True Negatives)

(b) Oversampling F3_ret_CS



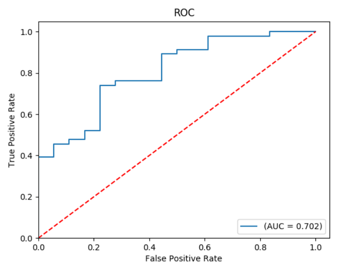
Predicted

Actual	Yes	No
Yes	38 (True Positives)	8 (False Negatives)
No	8 (False Positives)	10 (True Negatives)

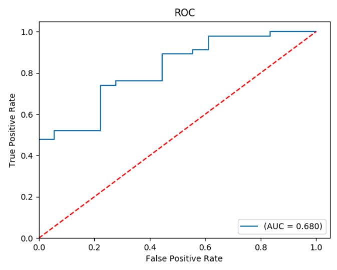
(c) Oversampling F3_ret_UNLV

Figure 4.27: Confusion matrix of LR model on test data (second-year retention for F3_ret_CS)

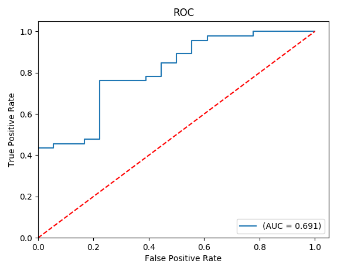
The ROC curves generated on test data in Figure 4.28 also show that without oversampling we found AUC score of 0.701 which is better than or equal to the AUC score found by oversampled results for output label F3_ret_CS.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

Figure 4.28: ROC curves of LR model on validation data (second-year retention for F3_ret_CS)

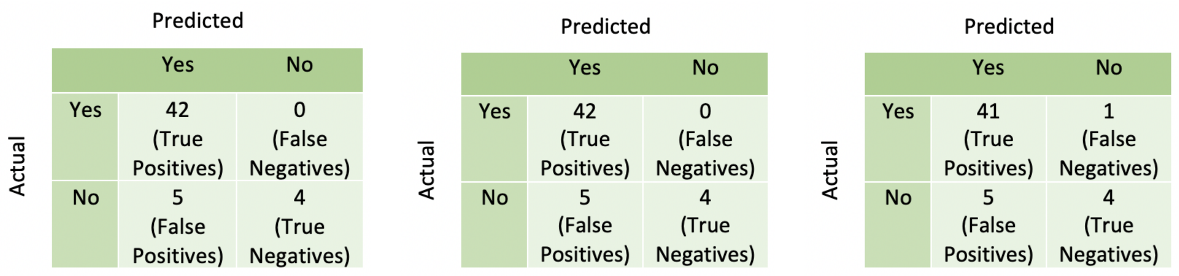
All other metrics calculated using confusion matrix (Figure 4.27) are shown in Table 4.14. Accuracy, specificity, and AUC results show that LR model without oversampling performed similar to other results.

Table 4.14: Results of LR model on test data (second-year retention for F3_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.848	0.830	0.556	0.839	0.444	0.701	0.766
F3_ret_CS	0.804	0.822	0.556	0.813	0.444	0.680	0.734
F3_ret_UNLV	0.826	0.826	0.556	0.826	0.444	0.691	0.750

4.2.2.4 Second-Year Retention for F3_ret_UNLV using LR Model

The confusion matrix generated on validation data is shown in Figure 4.29. Specificity calculated using this confusion matrix shows that without oversampling output labels, LR model gave good results predicting output for label F3_ret_UNLV.



Predicted

Actual	Yes	No
Yes	42 (True Positives)	0 (False Negatives)
No	5 (False Positives)	4 (True Negatives)

(a) Without oversampling

Predicted

Actual	Yes	No
Yes	42 (True Positives)	0 (False Negatives)
No	5 (False Positives)	4 (True Negatives)

(b) Oversampling F3_ret_CS

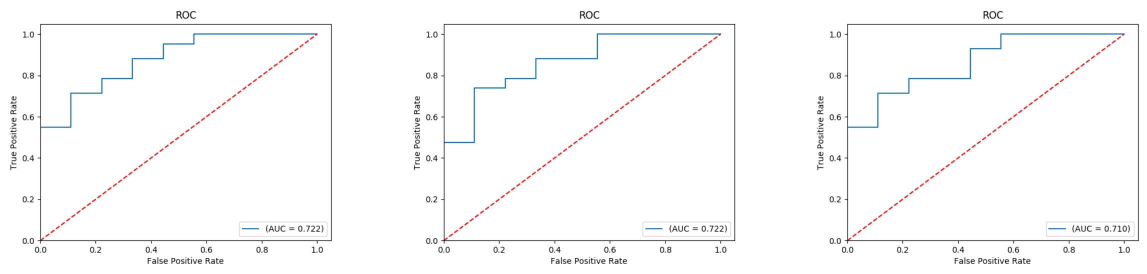
Predicted

Actual	Yes	No
Yes	41 (True Positives)	1 (False Negatives)
No	5 (False Positives)	4 (True Negatives)

(c) Oversampling F3_ret_UNLV

Figure 4.29: Confusion matrix of LR model on validation data (second-year retention for F3_ret_UNLV)

The ROC curves generated on validation data in Figure 4.30 also show that without oversampling we got AUC score of 0.722 which is closer to AUC score of 0.815 found by k-fold cross-validation using k=10 for output label F3_ret_UNLV.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

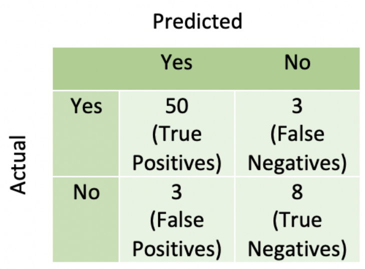
Figure 4.30: ROC curves of LR model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix are shown in Table 4.15. Accuracy, specificity, and AUC results show that LR model without oversampling performed similar to other results.

Table 4.15: Results of LR model on Validation data (second-year retention for F3_ret_UNLV)

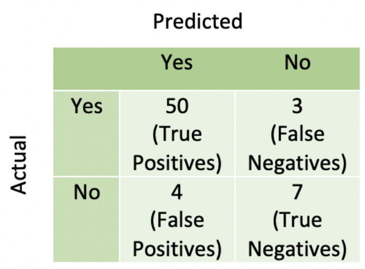
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	1.000	0.894	0.444	0.944	0.556	0.722	0.902
F3_ret_CS	1.000	0.894	0.444	0.944	0.556	0.722	0.902
F3_ret_UNLV	0.976	0.891	0.444	0.932	0.556	0.710	0.882

The confusion matrix generated on test data is shown in Figure 4.31. Specificity calculated using this confusion matrix shows that without oversampling output labels, LR model gave good results predicting output for label F3_ret_UNLV.



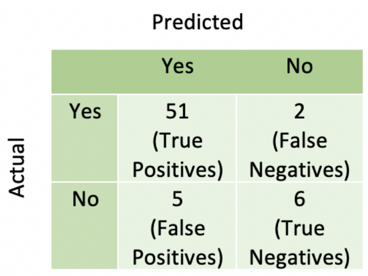
Actual \ Predicted	Yes	No
Yes	50 (True Positives)	3 (False Negatives)
No	3 (False Positives)	8 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	50 (True Positives)	3 (False Negatives)
No	4 (False Positives)	7 (True Negatives)

(b) Oversampling F3_ret_CS

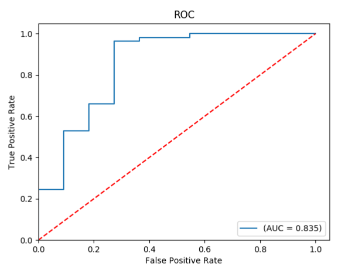


Actual \ Predicted	Yes	No
Yes	51 (True Positives)	2 (False Negatives)
No	5 (False Positives)	6 (True Negatives)

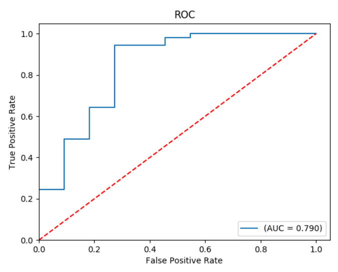
(c) Oversampling F3_ret_UNLV

Figure 4.31: Confusion matrix of LR model on test data (second-year retention for F3_ret_UNLV)

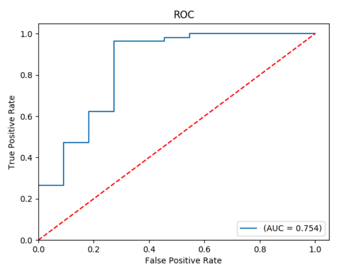
The ROC curves generated on test data in Figure 4.32 also show that without oversampling output labels we found AUC score of 0.835 which is better than or equal to the AUC score found by other results for output label F3_ret_UNLV.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

Figure 4.32: ROC curves of LR model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix (Figure 4.31) are shown in Table 4.16. Accuracy, specificity, and AUC results show that LR model without oversampling outperformed oversampled results.

Table 4.16: Results of LR model on test data (second-year retention for F3_ret_UNLV)

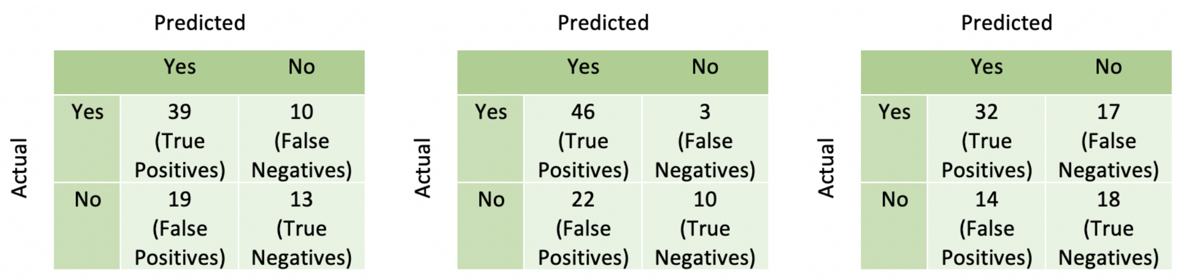
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.943	0.943	0.727	0.943	0.273	0.835	0.906
F3_ret_CS	0.943	0.926	0.636	0.935	0.364	0.790	0.891
F3_ret_UNLV	0.962	0.911	0.545	0.936	0.454	0.754	0.891

4.2.3 Support Vector Machine

A support vector machine model was designed using the sklearn package in Python. We used these designed models to fit our training data. The models were used to make predictions on validation and test data. We designed two different models to predict the first-year and the second-year retention. All of the evaluation metrics were calculated separately for each of the models.

4.2.3.1 First-Year Retention for F2_ret_CS using SVM Model

The confusion matrix generated on validation data is shown in Figure 4.33. Specificity calculated using this confusion matrix shows that with oversampling output label F2_ret_UNLV, SVM model gave good results predicting output for label F2_ret_CS.



		Predicted	
		Yes	No
Actual	Yes	39 (True Positives)	10 (False Negatives)
	No	19 (False Positives)	13 (True Negatives)

(a) Without oversampling

		Predicted	
		Yes	No
Actual	Yes	46 (True Positives)	3 (False Negatives)
	No	22 (False Positives)	10 (True Negatives)

(b) Oversampling F2_ret_CS

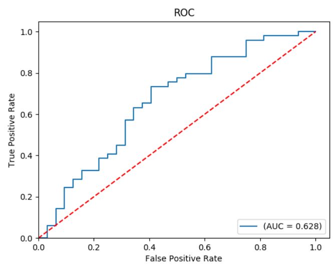
		Predicted	
		Yes	No
Actual	Yes	32 (True Positives)	17 (False Negatives)
	No	14 (False Positives)	18 (True Negatives)

(c) Oversampling F2_ret_UNLV

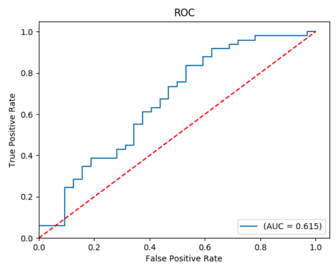
Figure 4.33: Confusion matrix of SVM model on validation data (first-year retention for F2_ret_CS)

The ROC curves generated on validation data in Figure 4.34 also show that without oversampling we got AUC score of 0.628 which is nearly equal to AUC score of 0.736 found by k-fold cross-validation using k=10 for output label F2_ret_CS.

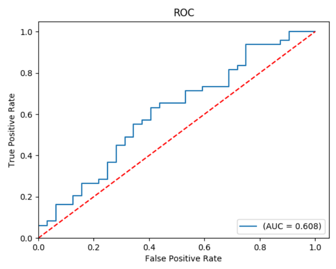
All other metrics calculated using confusion matrix are shown in Table 4.17. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar to oversampled results.



(a) Without oversampling



(b) Oversampling F2_ret_CS



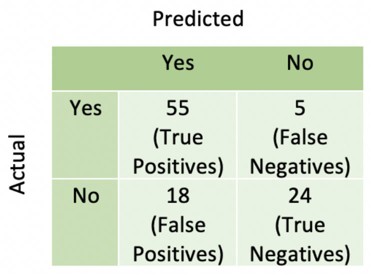
(c) Oversampling F2_ret_UNLV

Figure 4.34: ROC curves of SVM model on validation data (first-year retention for F2_ret_CS)

Table 4.17: Results of SVM model on Validation data (first-year retention for F2_ret_CS)

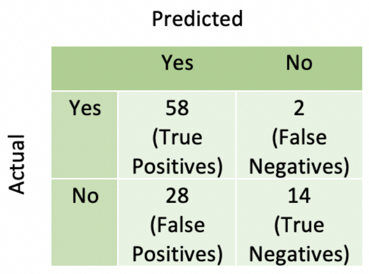
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.796	0.672	0.406	0.729	0.594	0.628	0.642
F2_ret_CS	0.939	0.676	0.312	0.786	0.687	0.615	0.691
F2_ret_UNLV	0.653	0.696	0.562	0.674	0.437	0.608	0.617

The confusion matrix generated on test data is shown in Figure 4.35. Specificity calculated using this confusion matrix shows that, with oversampling output label F2_ret_UNLV, SVM model gave good results predicting output for label F2_ret_CS.



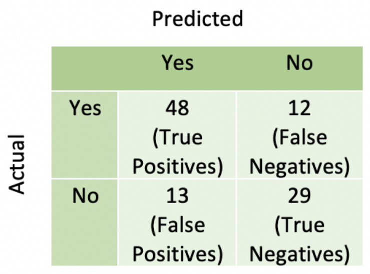
Actual		Predicted: Yes	Predicted: No
	Yes	55 (True Positives)	5 (False Negatives)
	No	18 (False Positives)	24 (True Negatives)

(a) Without oversampling



Actual		Predicted: Yes	Predicted: No
	Yes	58 (True Positives)	2 (False Negatives)
	No	28 (False Positives)	14 (True Negatives)

(b) Oversampling F2_ret_CS



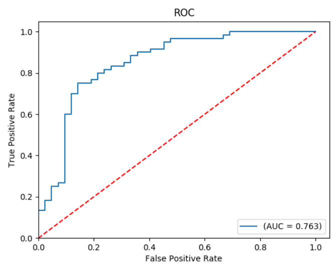
Actual		Predicted: Yes	Predicted: No
	Yes	48 (True Positives)	12 (False Negatives)
	No	13 (False Positives)	29 (True Negatives)

(c) Oversampling F2_ret_UNLV

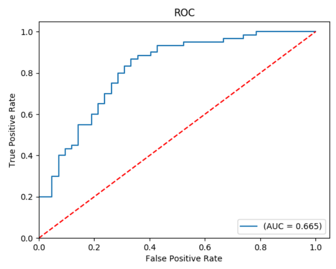
Figure 4.35: Confusion matrix of SVM model on test data (first-year retention for F2_ret_CS)

The ROC curves generated on test data in Figure 4.36 also show that without oversampling we found AUC score of 0.763 which is better than the AUC score found by oversampled results for output label F2_ret_CS.

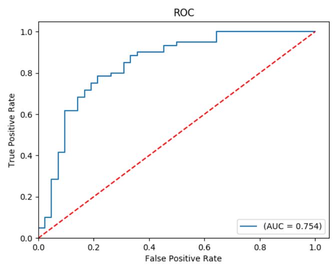
All other metrics calculated using confusion matrix (Figure 4.35) are shown in Table 4.18. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar oversampled results.



(a) Without oversampling



(b) Oversampling F2_ret_CS



(c) Oversampling F2_ret_UNLV

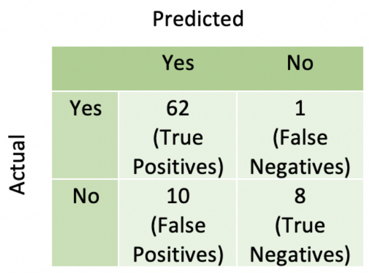
Figure 4.36: ROC curves of SVM model on validation data (first-year retention for F2_ret_CS)

Table 4.18: Results of SVM model on test data (first-year retention for F2_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.917	0.754	0.571	0.827	0.429	0.763	0.774
F2_ret_CS	0.967	0.674	0.333	0.794	0.667	0.665	0.706
F2_ret_UNLV	0.800	0.787	0.690	0.793	0.309	0.754	0.755

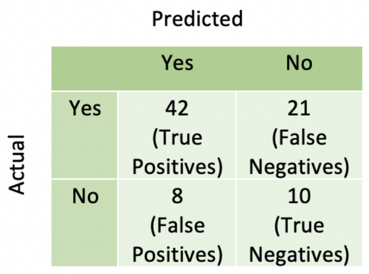
4.2.3.2 First-Year Retention for F2_ret_UNLV using SVM Model

The confusion matrix generated on validation data is shown in Figure 4.37. Specificity calculated using this confusion matrix shows that with oversampling output label F2_ret_UNLV, SVM model gave good results predicting output for label F2_ret_UNLV.



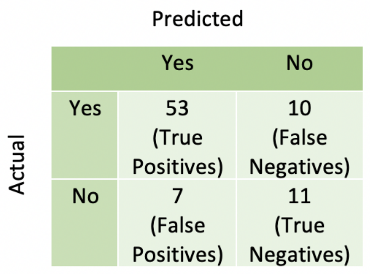
Actual	Predicted	Yes	No
	Yes	62 (True Positives)	1 (False Negatives)
	No	10 (False Positives)	8 (True Negatives)

(a) Without oversampling



Actual	Predicted	Yes	No
	Yes	42 (True Positives)	21 (False Negatives)
	No	8 (False Positives)	10 (True Negatives)

(b) Oversampling F2_ret_CS



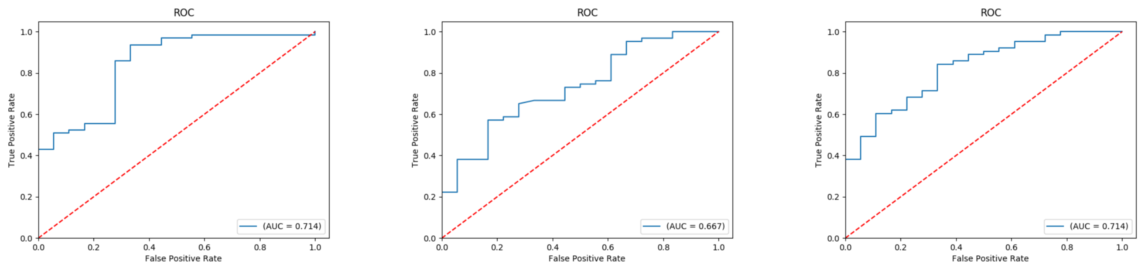
Actual	Predicted	Yes	No
	Yes	53 (True Positives)	10 (False Negatives)
	No	7 (False Positives)	11 (True Negatives)

(c) Oversampling F2_ret_UNLV

Figure 4.37: Confusion matrix of SVM model on validation data (first-year retention for F2_ret_UNLV)

The ROC curves generated on validation data in Figure 4.38 also show that without oversampling we got AUC score of 0.714 which is similar to AUC score of 0.785 found by k-fold cross-validation using k=10 for output label F2_ret_UNLV.

All other metrics calculated using confusion matrix are shown in Table 4.19. Accuracy, specificity,



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

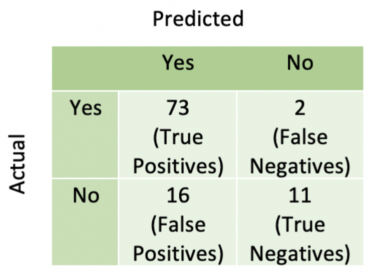
Figure 4.38: ROC curves of SVM model on validation data (first-year retention for F2_ret_UNLV)

and AUC results show that SVM model without oversampling performed similar to oversampled results.

Table 4.19: Results of SVM model on Validation data (first-year retention for F2_ret_UNLV)

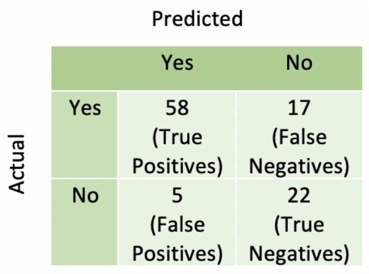
Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.984	0.861	0.444	0.918	0.556	0.714	0.864
F2_ret_CS	0.667	0.840	0.556	0.743	0.444	0.667	0.642
F2_ret_UNLV	0.841	0.883	0.611	0.862	0.389	0.714	0.790

The confusion matrix generated on test data is shown in Figure 4.39. Specificity calculated using this confusion matrix shows that with oversampling output label F2_ret_CS, SVM model gave good results predicting output for label F2_ret_UNLV.



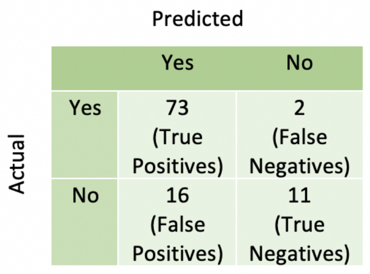
Actual \ Predicted	Yes	No
Yes	73 (True Positives)	2 (False Negatives)
No	16 (False Positives)	11 (True Negatives)

(a) Without oversampling



Actual \ Predicted	Yes	No
Yes	58 (True Positives)	17 (False Negatives)
No	5 (False Positives)	22 (True Negatives)

(b) Oversampling F2_ret_CS



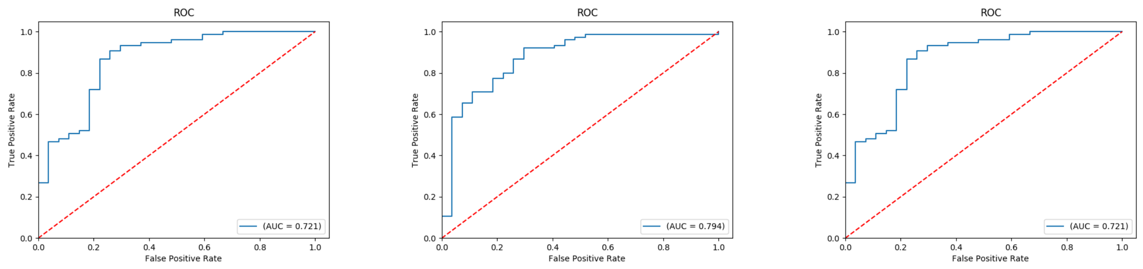
Actual \ Predicted	Yes	No
Yes	73 (True Positives)	2 (False Negatives)
No	16 (False Positives)	11 (True Negatives)

(c) Oversampling F2_ret_UNLV

Figure 4.39: Confusion matrix of SVM model on test data (first-year retention for F2_ret_UNLV)

The ROC curves generated on test data shown in Figure 4.40 also show that without oversampling we found AUC score of 0.721 which is better than or equal to the AUC score found by oversampled results for output label F2_ret_UNLV.

All other metrics calculated using confusion matrix (Figure 4.39) are shown in Table 4.20. Ac-



(a) Without oversampling (b) Oversampling F2_ret_CS (c) Oversampling F2_ret_UNLV

Figure 4.40: ROC curves of SVM model on validation data (first-year retention for F2_ret_UNLV)

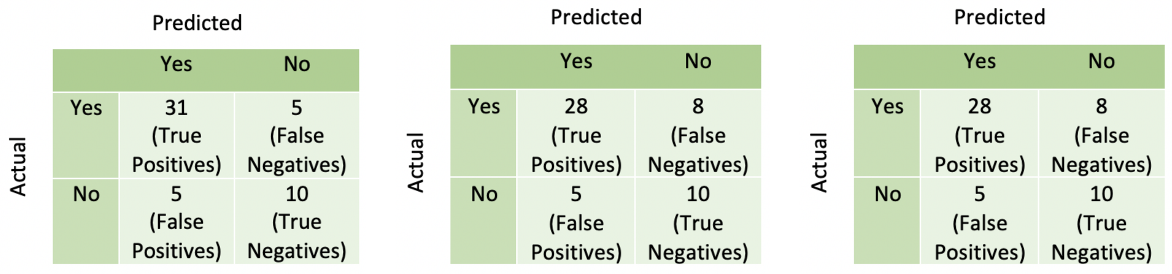
curacy, specificity, and AUC result shows that SVM model without oversampling outperformed oversampled results.

Table 4.20: Results of SVM model on test data (first-year retention for F2_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.973	0.820	0.407	0.890	0.593	0.721	0.823
F2_ret_CS	0.773	0.921	0.815	0.841	0.185	0.794	0.784
F2_ret_UNLV	0.973	0.820	0.407	0.890	0.593	0.720	0.821

4.2.3.3 Second-Year Retention for F3_ret_CS using SVM Model

The confusion matrix generated on validation data is shown in Figure 4.41. Specificity calculated using this confusion matrix shows that without oversampling output labels, SVM model gave good results predicting output for label F3_ret_CS.



Predicted

Actual	Yes	No
Yes	31 (True Positives)	5 (False Negatives)
No	5 (False Positives)	10 (True Negatives)

Predicted

Actual	Yes	No
Yes	28 (True Positives)	8 (False Negatives)
No	5 (False Positives)	10 (True Negatives)

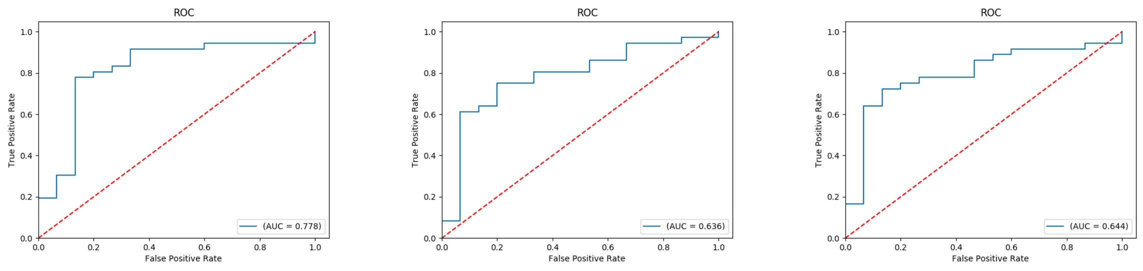
Predicted

Actual	Yes	No
Yes	28 (True Positives)	8 (False Negatives)
No	5 (False Positives)	10 (True Negatives)

(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

Figure 4.41: Confusion matrix of SVM model on validation data (second-year retention for F3_ret_CS)

The ROC curves generated on validation data in Figure 4.42 also show that without oversampling we got AUC score of 0.778 which is better than AUC score of 0.703 found by k-fold cross-validation using k=10 for output label F3_ret_CS.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

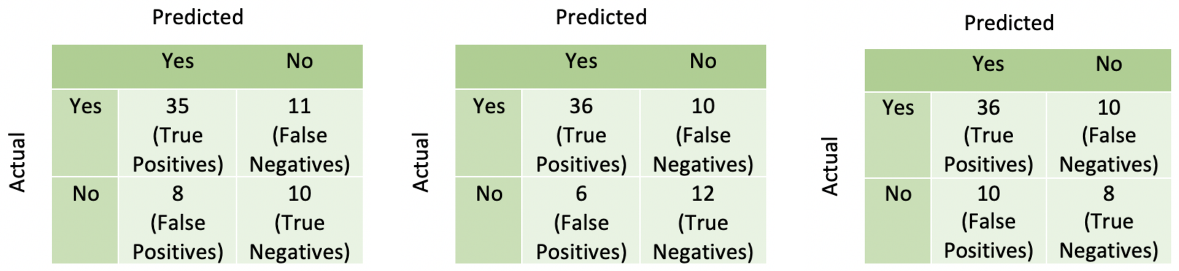
Figure 4.42: ROC curves of SVM model on validation data (second-year retention for F3_ret_CS)

All other metrics calculated using confusion matrix are shown in Table 4.21. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar to other results.

Table 4.21: Results of SVM model on Validation data (second-year retention for F3_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.861	0.861	0.667	0.861	0.333	0.778	0.804
F3_ret_CS	0.778	0.848	0.667	0.812	0.333	0.636	0.745
F3_ret_UNLV	0.778	0.848	0.667	0.812	0.333	0.644	0.745

The confusion matrix generated on test data is shown in Figure 4.43. Specificity calculated using this confusion matrix shows that, without oversampling output labels, SVM model gave good results predicting output for label F3_ret_CS.



Predicted / Actual

	Yes	No
Yes	35 (True Positives)	11 (False Negatives)
No	8 (False Positives)	10 (True Negatives)

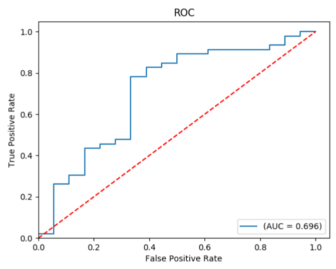
	Yes	No
Yes	36 (True Positives)	10 (False Negatives)
No	6 (False Positives)	12 (True Negatives)

	Yes	No
Yes	36 (True Positives)	10 (False Negatives)
No	10 (False Positives)	8 (True Negatives)

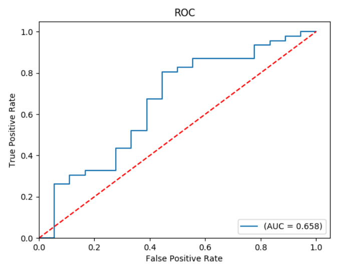
(a) Without oversampling (b) Oversampling F3_ret_UNLV (c) Oversampling F3_ret_CS

Figure 4.43: Confusion matrix of SVM model on test data (second-year retention for F3_ret_CS)

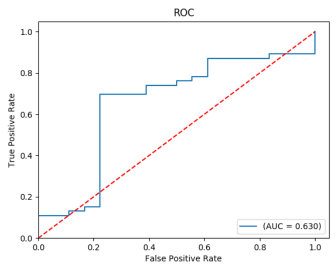
The ROC curves generated on test data in Figure 4.44 also show that without oversampling we found AUC score of 0.696 which is better than or equal to the AUC score found by oversampled results for output label F3_ret_CS.



(a) Without oversampling



(b) Oversampling F3_ret_CS



(c) Oversampling F3_ret_UNLV

Figure 4.44: ROC curves of SVM model on validation data (second-year retention for F3_ret_CS)

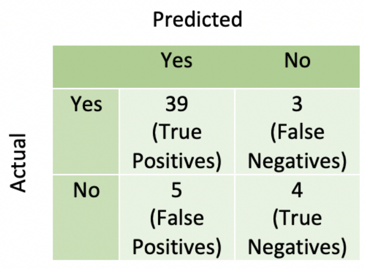
All other metrics calculated using confusion matrix (Figure 4.43) are shown in Table 4.22. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar to other results.

Table 4.22: Results of SVM model on test data (second-year retention for F3_ret_CS)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.783	0.857	0.667	0.818	0.333	0.696	0.750
F3_ret_CS	0.761	0.814	0.556	0.786	0.444	0.658	0.703
F3_ret_UNLV	0.783	0.783	0.444	0.783	0.556	0.630	0.687

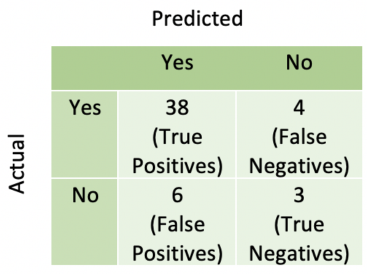
4.2.3.4 Second-Year Retention for F3_ret_UNLV using SVM Model

The confusion matrix generated on validation data is shown in Figure 4.45. Specificity calculated using this confusion matrix shows that without oversampling output labels, SVM model gave good results predicting output for label F3_ret_UNLV.



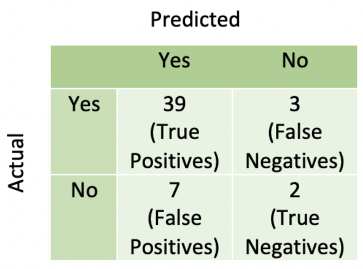
		Predicted	
		Yes	No
Actual	Yes	39 (True Positives)	3 (False Negatives)
	No	5 (False Positives)	4 (True Negatives)

(a) Without oversampling



		Predicted	
		Yes	No
Actual	Yes	38 (True Positives)	4 (False Negatives)
	No	6 (False Positives)	3 (True Negatives)

(b) Oversampling F3_ret_CS

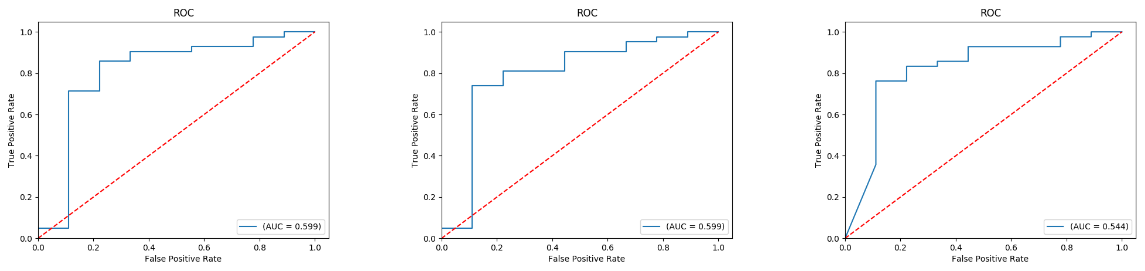


		Predicted	
		Yes	No
Actual	Yes	39 (True Positives)	3 (False Negatives)
	No	7 (False Positives)	2 (True Negatives)

(c) Oversampling F3_ret_UNLV

Figure 4.45: Confusion matrix of SVM model on validation data (second-year retention for F3_ret_UNLV)

The ROC curves generated on validation data in Figure 4.46 also show that without oversampling we got AUC score of 0.599 which is closer to AUC score of 0.801 found by k-fold cross-validation using k=10 for output label F3_ret_UNLV.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

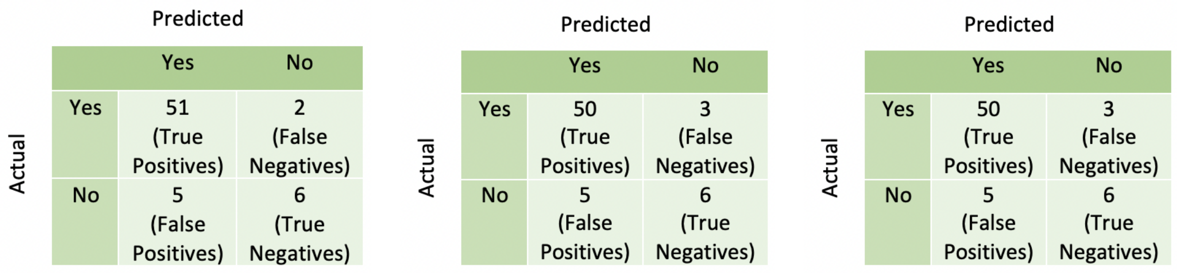
Figure 4.46: ROC curves of SVM model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix are shown in Table 4.23. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar to other results.

Table 4.23: Results of SVM model on Validation data (second-year retention for F3_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.929	0.886	0.444	0.907	0.556	0.599	0.843
F3_ret_CS	0.905	0.864	0.333	0.884	0.667	0.599	0.804
F3_ret_UNLV	0.929	0.848	0.222	0.886	0.778	0.544	0.804

The confusion matrix generated on test data is shown in Figure 4.47. Specificity calculated using this confusion matrix shows that without oversampling output labels, SVM model gave good results predicting output for label F3_ret_UNLV.



		Predicted	
		Yes	No
Actual	Yes	51 (True Positives)	2 (False Negatives)
	No	5 (False Positives)	6 (True Negatives)

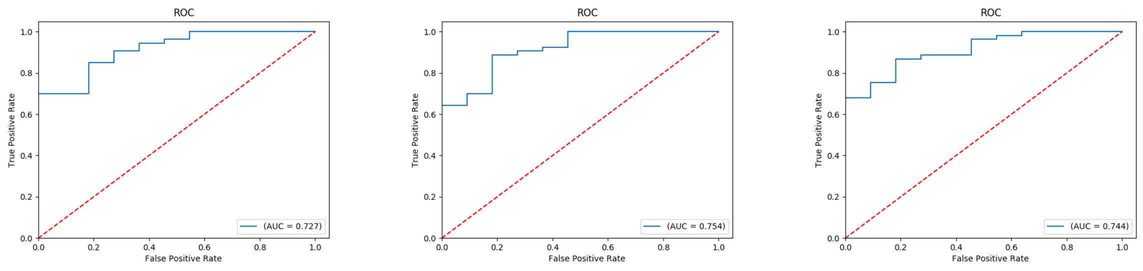
		Predicted	
		Yes	No
Actual	Yes	50 (True Positives)	3 (False Negatives)
	No	5 (False Positives)	6 (True Negatives)

		Predicted	
		Yes	No
Actual	Yes	50 (True Positives)	3 (False Negatives)
	No	5 (False Positives)	6 (True Negatives)

(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

Figure 4.47: Confusion matrix of SVM model on test data (second-year retention for F3_ret_UNLV)

The ROC curves generated on test data in Figure 4.48 also show that without oversampling output labels we found AUC score of 0.835 which is better than or equal to the AUC score found by other results for output label F3_ret_UNLV.



(a) Without oversampling (b) Oversampling F3_ret_CS (c) Oversampling F3_ret_UNLV

Figure 4.48: ROC curves of SVM model on validation data (second-year retention for F3_ret_UNLV)

All other metrics calculated using confusion matrix (Figure 4.47) are shown in Table 4.24. Accuracy, specificity, and AUC results show that SVM model without oversampling performed similar to other results.

Table 4.24: Results of SVM model on test data (second-year retention for F3_ret_UNLV)

Oversampling	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
Without	0.962	0.911	0.545	0.936	0.454	0.727	0.891
F3_ret_CS	0.943	0.909	0.545	0.926	0.454	0.754	0.875
F3_ret_UNLV	0.943	0.909	0.545	0.926	0.454	0.744	0.875

4.3 Comparison Study of Machine Learning Models

4.3.1 Choosing best model to predict first-year retention

For predicting output label F2_ret_CS, oversampling output labels gave almost similar results to without oversampling output labels. Hence, we used all the results found by without oversampling to choose the best model to predict first-year retention in Department of Computer Science. Table 4.25 shows that logistic regression model worked best on predicting output for label F2_ret_CS with accuracy of 0.843 and specificity of 0.714.

Table 4.25: Best model results to predict first-year retention for F2_ret_CS

Model	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
FNN	0.917	0.775	0.619	0.840	0.381	0.768	0.794
LR	0.933	0.823	0.714	0.875	0.286	0.824	0.843
SVM	0.917	0.754	0.571	0.827	0.429	0.763	0.774

Similarly, for predicting output label F2_ret_UNLV, we used all the results found by without oversampling to choose the best model to predict first-year retention at UNLV. Table 4.26 shows that logistic regression model worked best on predicting output for label F2_ret_UNLV with accuracy of 0.863 and specificity of 0.741.

Table 4.26: Best model results to predict first-year retention for F2_ret_UNLV

Model	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
FNN	0.947	0.845	0.518	0.893	0.481	0.733	0.833
LR	0.907	0.907	0.741	0.907	0.259	0.824	0.863
SVM	0.973	0.820	0.407	0.890	0.593	0.721	0.823

4.3.2 Choosing best model to predict second-year retention

For predicting output label F3_ret_CS, oversampling output labels gave almost similar results to without oversampling output labels. Hence, we used all the results found by without oversampling to choose the best model to predict second-year retention in Department of Computer Science. Table 4.27 shows that feedforwared neural network model worked best on predicting output for label F3_ret_CS with accuracy of 0.843 and specificity of 0.714.

Table 4.27: Best model results to predict second-year retention for F3_ret_CS

Model	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
FNN	0.848	0.886	0.722	0.867	0.278	0.785	0.812
LR	0.848	0.830	0.556	0.839	0.444	0.701	0.766
SVM	0.783	0.857	0.667	0.818	0.333	0.696	0.750

Similarly, for predicting output label F3_ret_UNLV, we used all the results found by without oversampling to choose the best model to predict second-year retention at UNLV. Table 4.28 shows that feedforwared neural network model worked best on predicting output for label F3_ret_UNLV with accuracy of 0.953 and specificity of 0.727.

Table 4.28: Best model results to predict second-year retention for F3_ret_UNLV

Model	Recall	Precision	Specificity	F_1 score	Fallout	AUC	Accuracy
FNN	1.000	0.946	0.727	0.972	0.273	0.864	0.953
LR	0.943	0.943	0.727	0.943	0.273	0.835	0.906
SVM	0.962	0.911	0.545	0.936	0.454	0.727	0.891

Chapter 5

Conclusion and Future work

In this thesis, we worked on the problem of predicting first-year and second-year retention in the Department of Computer Science at the University of Nevada, Las Vegas. We determined and collected features including students' pre-university academic information, demographic information, and standardized test scores (like SAT and ACT), along with current academic information including grades obtained in each course and total credits enrolled in for each term, as well as GPA, and cumulative GPA. We collected this data from UNLV's enterprise data warehouse, UNLV Analytics. We used SAS DIS to clean and transform the data.

The cleaned and transformed data was used to train our selected machine learning models, feedforward neural network, logistic regression, and support vector machine. We evaluated all the models using various classification evaluation metrics to determine the best model, which would work on first-year and second-year students' academic data to identify students who are at-risk of dropping out. Our main focus was to reduce the number of falsely classified negative examples, which we evaluated using specificity metric calculated from the confusion matrix.

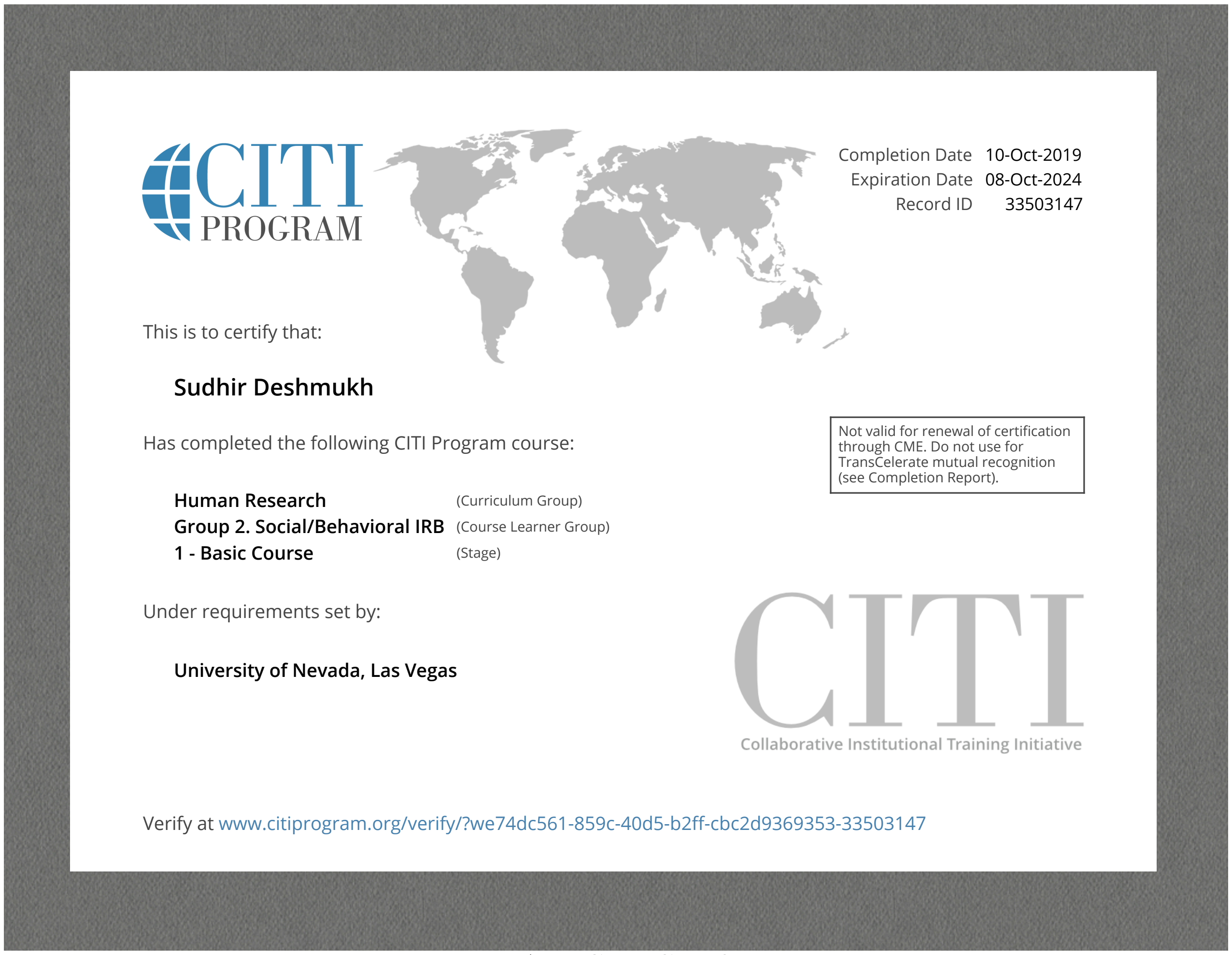
The results of the first-year retention reveal that logistic regression works best to predict retention in the Department of Computer Science, with accuracy of 0.843 and specificity of 0.714, as well as retention at UNLV, with accuracy of 0.863 and specificity of 0.741 [Table 4.25 and 4.26]. The results of second-year retention reveal that feedforward neural network works best to predict retention in the Department of Computer Science, with accuracy of 0.812 and specificity of 0.722, as well as retention at UNLV, with accuracy of 0.953 and specificity of 0.727 [Table 4.27 and 4.28]. We also performed chi-square (x^2) test to determine the importance of features in predicting the output labels. Results show that the most important features in predicting retention are total credits enrolled in spring and fall terms, GPAs, and the required CS courses [Table 3.8 and 3.9]. This feature importance can help in determining which courses are important in students' retention. Using this information, the department can create programs to update the course structure, so that students can increase their academic performance. Based on the results generated by our best models, the university can reach all students who are predicted to be at-risk of not being retained to reduce the number of dropouts.

For future work, we would like to gather more data about students, including student/teacher relations, students' campus events involvement, hours spent in the library, midterm grades, assignment grades, financial situation, scholarship and funds information, participation in collaborative studies and other features so that we can boost the prediction accuracy and improve the specificity metric to better identify the students who are at-risk of dropping out.

The selected models performed well, however, in the future, we would like to explore other machine learning models. These datasets, based on their feature characteristics, would lend themselves to be modeled well on algorithms like decision trees and random forest. So, we would like to implement those to compare and improve predictive power.

Appendix A

CITI Course Completion Certificate



Completion Date 10-Oct-2019
Expiration Date 08-Oct-2024
Record ID 33503147

This is to certify that:

Sudhir Deshmukh

Has completed the following CITI Program course:

Not valid for renewal of certification through CME. Do not use for TransCelerate mutual recognition (see Completion Report).

Human Research	(Curriculum Group)
Group 2. Social/Behavioral IRB	(Course Learner Group)
1 - Basic Course	(Stage)

Under requirements set by:

University of Nevada, Las Vegas

Collaborative Institutional Training Initiative

Verify at www.citiprogram.org/verify/?we74dc561-859c-40d5-b2ff-cbc2d9369353-33503147

Figure A.1: CITI Certificate

Bibliography

[AA12] Alexander Astin and Anthony Antonio. *Assessment for Excellence: The Philosophy and Practice of Assessment and Evaluation in Higher Education*. Rowman & Littlefield Publishers, Inc., 2 edition, 2012.

[ACT] National Collegiate Retention and Persistence-to-Degree Rates, 2018 https://www.act.org/content/dam/act/unsecured/documents/MS2807rev1-retention-persistence-%0D2018-07.pdf [retrieved 2020-03-23].

[Ast84] Alexander Astin. Student involvement: A developmental theory for higher education. *Journal of College Student Personnel*, 25(3):297–308, 1984.

[BMB+18] Paolo Massimoand Buscema, Giuliaand Massini, Weldon A. Breda, Marcoand Lodwick, Francisand Newman, and Masoud Asadi-Zeydabadi. *Artificial Neural Networks*, pages 11–35. Springer International Publishing, Cham, 2018.

[BÖ14] Yalinand Baştanlar and Mustafa Özuysal. *Introduction to Machine Learning*, pages 105–128. Humana Press, Totowa, NJ, 2014.

[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.

[CBHK02] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, Jun 2002.

[Che] Lujing Chen. Support Vector Machine — Simply Explained, https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496 [retrieved 2020-04-18].

[Cola] ACT and SAT® Concordance Tables https://files.eric.ed.gov/fulltext/ED562594.pdf [retrieved 2020-04-18].

[Colb] Concordance Tables for the New and Old SAT https://www.collegeboard.org/pdf/technical-information-on-sat-concordance.pdf [retrieved 2020-04-18].

[Coo95] Melanie M Cooper. Cooperative Learning: An Approach for Large Enrollment Courses. *Journal of Chemical Education*, 72(2):162–164, feb 1995.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[cvx] Logistic Regression https://cvxopt.org/examples/book/logreg.html [retrieved 2020-04-19].

[DAM02] Stephen L. DesJardins, Dennis A. Ahlburg, and Brian P. McCall. A temporal investigation of factors related to timely degree completion. *Journal of Higher Education*, 73(5):555–581, 2002.

[FF08] David S. Fike and Renea Fike. Predictors of first-year student retention in the community college. *Community College Review*, 36(2):68–88, 2008.

[HBGL08] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328, 2008.

[Hen06] Darwin Hendel. Efficacy of participating in a first-year seminar on studnet satisfaction and retention. *Journal of College Student Retention: Research, Theory Practice*, 8(4):413–423, 2006.

[KBP12] Surjeet Kumar, Brijesh Bharadwaj, and Saurabh Pal. Mining Education Data to Predict Student's Retention: A comparative Study. *International Journal of Computer Science and Information Security*, 10(2):113–117, mar 2012.

[Kot07] S.B. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica (Ljubljana)*, 31:249–268, 2007.

[Lau03] Linda Lau. Institutional Factors Affecting Student Retention. *Education*, 124(1):126–137, 2003.

[LF15] Erik Larsen and Frank Ferriola. A Practical Introduction to SAS® Data Integration Studio. In *SAS Conference Proceedings: Western Users of SAS Software*, San Diego, 2015.

[LL12] Marjan Laal and Mozhgan Laal. Collaborative learning: What is it? *Procedia - Social and Behavioral Sciences*, 31(2011):491–495, 2012.

[Mal] Heather Maloney. How the pursuit of Artificial Intelligence is changing our world, https://blog.contactpoint.com.au/2018/04/how-the-pursuit-of-artificial-intelligence-is-changing-our-world/ [retrieved 2020-03-15].

[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 and Education*, 103:1–15, 2016.

[Mit97] Tom Mitchell. *Machine Learning*. New York : McGraw Hill, New York, 1997.

[Nas96] Ian Nash. Spotlight falls on drop-outs (United Kingdom Colleges of Further Education, Recruitment, and Retention Plans). *Times Educational Supplement*, 1996.

[Oxf] Oxford Advanced Learner's Dictionary. Defination of learning, https://www.oxfordlearnersdictionaries.com/us/definition/english/learning?q=learning [retrieved 2020-03-23].

[Pex] Images: cats,dogs https://www.pexels.com [retrieved 2020-04-17].

[PT05] Ernest T Pascarella and Patrick T Terenzini. *How college affects students: A third decade of research.* 2005.

[Raj18] Aditya Rajuladevi. A Machine Learning Approach to Predict First-Year Student Retention Rates at University of Nevada, Las Vegas. *Thesis, Department of Computer Science, University of Nevada, Las Vegas*, 2018.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[Sam59] A L Samuel. Some Studies in Machine Learning Using the Game of Checkers. *IBM J. Res. Dev.*, 3(3):210–229, jul 1959.

[Tin75] Vincent Tinto. Dropout from Higher Education: A Theoretical Synthesis of Recent Research. *Review of Educational Research*, 45(1):89–125, 1975.

[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 and Practice*, 8(1):1–19, 2006.

[TRL+94] Patrick T Terenzini, Laura I Rendon, M Lee Upcraft, Susan B Millar, Kevin W Allison, Patricia L Gregg, and Romero Jalomo. The transition to college: Diverse students, diverse stories. *Research in Higher Education*, 35(1):57–73, 1994.

[Unia] College Navigator - National Center for Education Statistics https://nces.ed.gov/collegenavigator/?q=university+of+nevada+las+vegas&s=NV&id=182281 [retrieved 2020-03-23].

[Unib] Headcount Enrollment by College and Major, Summarized by Degree Type https://ir.unlv.edu/IAP/Reports/Content/HeadcountByCollegeAndMajorSummarized ByDegreeT%0Dype.aspx [retrieved 2020-04-18].

[unl] UNLV Analytics, https://analytics.unlv.nevada.edu/analytics/saw.dll?BIEEHomestartPage=1 [retrieved 2020-04-18].

[Wik] Linear Regression https://commons.wikimedia.org/wiki/File:Linear_regression.svg [retrieved 2020-04-18].

[YL04] Mantz Yorke and Bernard Longden. *Retention and student success in higher education.* Society for Research into Higher Education Open University Press, 2004.

Curriculum Vitae

Graduate College

University of Nevada, Las Vegas

Sudhir Deshmukh

sudhirdesh92@gmail.com

Degrees:

Bachelor of Engineering in Computer Engineering 2015

University of Mumbai, India

Thesis Title: A Machine Learning Approach to Predict Retention of Computer Science Students at University of Nevada, Las Vegas

Thesis Examination Committee:

Chairperson, Dr. Fatma Nasoz, Ph.D.

Committee Member, Dr. Kazem Taghva, Ph.D.

Committee Member, Dr. Laxmi Gewali, Ph.D.

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