

5-2016

# Predicting Student Retention: A Comparative Study of Predictive Models for Predicting Student Retention at St. Cloud State University

Hasith U. Dissanayake

St Cloud State University, diha1201@stcloudstate.edu

Follow this and additional works at: [https://repository.stcloudstate.edu/csit\\_etds](https://repository.stcloudstate.edu/csit_etds)



Part of the [Computer Sciences Commons](#)

---

## Recommended Citation

Dissanayake, Hasith U., "Predicting Student Retention: A Comparative Study of Predictive Models for Predicting Student Retention at St. Cloud State University" (2016). *Culminating Projects in Computer Science and Information Technology*. 10.  
[https://repository.stcloudstate.edu/csit\\_etds/10](https://repository.stcloudstate.edu/csit_etds/10)

This Thesis is brought to you for free and open access by the Department of Computer Science and Information Technology at theRepository at St. Cloud State. It has been accepted for inclusion in Culminating Projects in Computer Science and Information Technology by an authorized administrator of theRepository at St. Cloud State. For more information, please contact [rswexelbaum@stcloudstate.edu](mailto:rswexelbaum@stcloudstate.edu).

**Predicting Student Retention: A Comparative Study of Predictive Models for Predicting  
Student Retention at St. Cloud State University**

by

Hasith Dissanayake

A Thesis

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science in

Computer Science

May, 2016

Thesis Committee:  
Omar Al-Azzam, Chairperson  
David Robinson  
Jie Meichsner

### **Abstract**

Student graduation rates have always taken prominence in academic studies since it is considered a major factor that reflects the performance of any university. Using accurate models for predicting student retention is important to a university's strategic planning and decision making process. Students' enrollment behavior and retention rate are also relevant factors in the measurement of the effectiveness of universities. This thesis provides a comparison of predictive models for predicting student retention at Saint Cloud State University, St. Cloud, MN (SCSU). The models were trained and tested using a set of features reflecting the readiness of students for college education, their academic capacities, financial situation, and academic results during their freshman year. Principle Component Analysis (PCA) was used for feature selection. Six predictive models were built and a comparison of the prediction results was made using all and selected features using PCA analysis. Having analyzed the experimental results it was concluded that either Bayesian Neural Networks or Random Forest can be used with the PCA structure to predict student retention at SCSU as both produced results that were close to each other. Also, considering the performance of all models, it was determined that the use of data mining and an algorithmic approach to predict retention can yield results with high accuracy.

Table of Contents

	Page
List of Tables .....	5
List of Figures .....	8
Chapter	
1. Introduction .....	9
2. Literature Review .....	11
2.1 Predictive Models .....	14
3. Research Methodology .....	25
3.1 Data Used in the Research .....	25
3.2 Data Cleaning Process .....	27
3.3 Data Type Conversion .....	27
3.4 Aggregate Data to Student ID Level .....	29
3.5 Removing Outliers .....	29
3.6 Cleaning Missing Values .....	37
4. Building Models .....	49
4.1 Confusion Matrix .....	49
4.2 Preprocessing .....	52
4.3 Identifying Near-Zero Variance Variables .....	53
4.4 Principal Component Analysis .....	54
4.5 Training Models .....	55
5. Comparing Models and Experimental Results .....	69

Chapter	Page
5.1 Measures Used to Compare Models and Calculated Statistics .....	69
5.2 Top Predictors and Retention Probabilities .....	72
6. Discussion .....	77
References .....	79
Additional Reading .....	82
Appendix .....	83

### List of Tables

Table	Page
3.1 Number of missing values for each variable in the dataset .....	37
3.2 Summary statistics of the QPP variable .....	38
3.3 Summary statistics of the T1_TermGPA variable .....	39
3.4 Summary statistics of the log (MilesToSCSU) variable .....	40
3.5 Retained probabilities for the FAFSADaysBeforeTerm with respect to the availability of data points .....	41
3.6 Summary statistics of the FAFSADaysBeforeTerm variable .....	42
3.7 Summary statistics of the FAFSADayOfYear variable .....	42
3.8 Frequency table of the FAFSA_Nbr variable .....	43
3.9 Summary statistics of the EnglTotal variable .....	44
3.10 Summary statistics of the HS_GPA_4Scale variable .....	44
3.11 Summary statistics of the ACT_Composite variable .....	45
3.12 Summary statistics of the HSPct variable .....	45
3.13 Summary statistics of the Total CRHR_TRSF variable .....	46
3.14 Frequency table of the HS-MnSCURegion variable .....	47
4.1 Distribution of T3_Enrolled variable .....	49
4.2 Example Confusion Matrix .....	49
4.3 Confusion Matrix of KNN using PCA data structure .....	56
4.4 Computed statistics based on actual values and predicted values of KNN (PCA data structure) .....	56

Table	Page
4.5 Confusion Matrix of KNN using original set of variables .....	57
4.6 Computed statistics based on actual values and predicted values of KNN (original set of variables) .....	57
4.7 Confusion Matrix of Classification Tree using PCA data structure .....	58
4.8 Computed statistics based on actual values and predicted values of Classification Tree (PCA data structure) .....	58
4.9 Confusion Matrix of Classification Tree using original set of variables .....	59
4.10 Computed statistics based on actual values and predicted values of Classification Tree (original set of variables) .....	59
4.11 Confusion Matrix of Random Forest using the PCA data structure .....	60
4.12 Computed statistics based on actual values and predicted values of Random Forest (PCA data structure) .....	60
4.13 Confusion Matrix of Random Forest using original set of variables .....	61
4.14 Computed statistics based on actual values and predicted values of Random Forest (original set of variables) .....	61
4.15 Confusion Matrix of Binomial GLM using PCA data structure .....	62
4.16 Computed statistics based on actual values and predicted values of Binomial GLM (PCA data structure) .....	62
4.17 Confusion Matrix of Binomial GLM using original set of variables .....	63
4.18 Computed statistics based on actual values and predicted values of Binomial GLM (original set of variables) .....	63

Table	Page
4.19 Confusion Matrix of Neural Network using PCA data structure .....	64
4.20 Computed statistics based on actual values and predicted values of Neural Network (PCA data structure) .....	64
4.21 Confusion Matrix of Neural Network using original set of variables .....	65
4.22 Computed statistics based on actual values and predicted values of Neural Network (original set of variables) .....	65
4.23 Confusion Matrix of BNN using PCA data structure .....	66
4.24 Computed statistics based on actual values and predicted values of BNN (PCA data structure) .....	67
4.25 Confusion Matrix of BNN using original set of variables .....	67
4.26 Computed statistics based on actual values and predicted values of BNN (original set of variables) .....	68
5.1 Calculated statistics of predictions of models using principal component analysis structure .....	69
5.2 Calculated statistics of predictions of models using original set of variables .....	70
5.3 Performance deference when using PCA structure over original set of variables .....	70



### List of Figures

Figure	Page
2.1 Classification tree of product outcome for solar cooker and pressure cooker .....	17
2.2 Schematic diagram of a hidden layer in feed-forward neural network .....	24
3.1 Histogram and data plot for the ACT_Composite scores variable .....	30
3.2 Histogram and data plot for the MilesToSCSU variable .....	31
3.3 Histogram and data plot for the Houseing AppDaysBeforeTerm variable .....	32
3.4 Histogram and the data plot of the AdmitDaysBeforeTerm variable .....	33
3.5 Bar chart for HS_MnSCURegion variable .....	34
3.6 Histogram and data plot of the T1_TermCululativeLocalCreditsEarned variable .....	35
3.7 Histogram and data plot of the T2_TermCumulativeLocalCreditsEarned variable .....	36
3.8 Boxplot of the FAFSADaysBeforeTerm vs T3_Enrolled .....	41
5.1 Top 15 predictors for BNN .....	73
5.2 Top 15 predictors for Random Forest .....	74
5.3 Retention probability by gender .....	75
5.4 Retention probability by college .....	76

## **Chapter 1: Introduction**

Student enrollment behavior remains a significant focus in institutional research. Student retention is a serious national issue, and some academic areas experience it more than the others. Knight et al. (2003) argued that student retention became a nationwide problem in the 1980s, when retention reached 40%. This statistics suggest that every 4th student leaves their college or university before graduation, and Knight called this “leakage from the engineering pipeline” [1].

While having an understanding of the reasons behind student drop-out or transfer behavior is crucial for effective enrollment management, what is even more important is the ability to predict these types of behavior to develop preventive measures. This will allow more precise tuition revenue forecasts, enrollment rate predictions, and help develop successful intervention and recruiting programs [2]. Theoretical data indicates using algorithmic approach and data mining can provide more accurate results when predicting student retention compared to traditional statistical methods [3].

This research uses data mining and an algorithmic approach to predict retention with high accuracy, while identifying the variables that could affect student dropout. These identified variables were used to create predictive models that would predict the likeliness of students’ retention to their sophomore year. Considering the nature and the diversity of data available for this study, it was assumed that highly accurate predictions could be made.

Data related to the admissions process were obtained from the Admissions Office at St. Cloud State University (SCSU). These data were originally collected by SCSU from 2006-2010. Data related to academic records of students were obtained from the Office of Record and

Registration. This dataset was extracted from ISRS in 2013 by the Office of Strategy, Planning and Effectiveness at SCSU.

The models that are used in this study are k-NN (k Nearest Neighbor), Classification Tree, Random Forest, Binomial GLM, Neural Network, and Bayesian Neural Network. During the process of training the predictive models, the data were first examined for outliers and missing values. Next, these values were replaced with appropriate values to minimize the effect on incorrect predictions. Since the data were obtained through different sources, the datasets were then aggregated to one dataset linking them through the Student ID of each dataset. Out of the combined dataset, 70 variables were identified as relevant factors to student retention. At this point it was determined that grouping these variables through a Principal Component Analysis (PCA) could yield better results than using the entire set of variables to train the models. But the models were trained both ways to test the validity of this hypothesis. Finally the models were compared using their accuracy, sensitivity, specificity, positive predictive value, and negative predictive value, to identify the best model that suited the set of variables that was available.

## **Chapter 2: Literature Review**

Student retention is a widely researched area in the higher education sector, and it spans over four decades of research. In Tinto's (2006) study, he states that "there has also been a concomitant increase in the number of businesses and consulting firms that have sprung up, each of which claims unique capacity to help institutions increase the retention of their students." This shows that the amount of research that goes into student retention in higher education sector is large. It has eventually become a business to consult institutions on how to retain students. But in the same study he later states that even though this is the case, the 2005 National Center for Education Statistics reveal that the "national rate of student retention has shown disappointingly little change over the past decade" [4].

According to Tinto (2006), the reason for this is that there is still much work to be done in this field and that there is a lack of translating the research and theory into effective practice. Tinto (2006) further states that before the 1970s the reason identified for low student retention rates was the failure on the part of the student and not the institution. Students who dropped out were thought to be less able, less motivated, and willing to defer the benefits of college education. Research carried out after 1970s, mostly by Alexander Astin (1975, 1984), Ernest Pascarella (1980), and Patrick Terenzini (1980), focused more on the environment, in this case the institutions and the people who govern them, when determining reasons for student retention [4]. With the focus shifting to the institutions, they began to pay more attention to the student-faculty relationships mostly outside the classroom and the transition to college, by introducing extended orientation, freshman seminars, and a variety of extracurricular programs for first-year students [4]. These programs were introduced to make them feel welcome to the new culture,

community or the new environment in college. Around the 1970s, institutions held the view that students needed to break away from the society that they were in, in order to adapt to college and thereby remain in college. But later, they discovered that the gap between breaking away from society and adapting to college should be bridged through orientation programs and extracurricular programs, making them feel part of their past communities, families, churches or tribes [4].

Predictive modeling for student retention can be seen as early as 1975 with Tinto's model. Following Tinto's model, there have been many models introduced by researchers that consider different factors and variables to predict student retention. Some of these models focused on identifying students with high risk of dropping out from college [5], [6]. Furthermore, Alkhasawneh and Hargraves (2014) state that according to Ben Gaskin (2009), traditional methods of statistical analysis such as logistic regression have been used to predict student retention. In most recent years, data mining, which is recognizing patterns in large sets and then understanding those patterns, has been used to study student retention because of high accuracy and the robustness with missing data [5].

As mentioned earlier, in the start of predictive modeling of student retention in 1975, Tinto's model considered social and academic impacts on a student's decision (voluntarily or involuntarily) to drop out from college. The model is based on Durkheim's (1961) theory of suicide, especially its notions of the cost-benefit analysis of individual decisions about investment in alternative educational activities, which comes from the field of economics of education. Tinto (1975) makes the connection with Durkheim's suicide theory by considering the case of dropping out as committing suicide and views college as a social system. In his study, he

then relates all the reasons behind committing suicide to that of dropping out of a college when it is viewed as a social system. Furthermore, as colleges also consist of an academic system, he combines the academic factors to the model to be more effective and to shape it to be more suitable to the college structure [7]. These factors are valid even for today's college structure and should be considered as inputs in the predictive models.

In addition to Tinto's ideas in 1975, Astin (1993), in his I-E-O (Input-Environment-output) model, suggests that in addition to factors that affect student retention during their college life, researchers should also explore factors that affect student retention before entering college, such as race/ethnicity, gender, family background (which he calls "pre-college characteristics"), high school GPA, and student self-reported data. In his study, he discusses how these factors affect one's academic and social life while that person is in college and how it could affect the decision s/he makes [8]. Other than the factors Tinto focused on in his research, the factors that Astin discusses in his research also could be variables that might change the outcome of a retention model and make it more accurate.

This study is an attempt to identify factors that contribute to student retention, primarily taking into consideration the models of both Tinto (1975) and Astin (1993), but also by using data obtained from the SCSU Admissions Office on student admission and first year grades, and from first year student surveys. Knowledge and research of time to degree (TTD) completion remains one of the blind spots of student retention studies. A number of models have been developed to build successful predictions: k-NN, decision trees, Bayesian networks and neural networks, among others [9].

## 2.1 Predictive Models

This section discusses various models that were used in this thesis to find the best suited model to predict the retention rate at St Cloud State University. The models that were selected for the comparison were k-NN (k-Nearest Neighbor), Classification Tree, Random Forest, Binomial GLM, Neural Network and Bayesian Neural Network.

### 2.1.1 k-Nearest Neighbor Algorithm (k-NN)

The k-Nearest Neighbor algorithm is an instance-based-learning non-parametric method to build classification [12]. A “matching matrix” is used as a tool to evaluate k-NN model accuracy. The method is based on the assignment of the unclassified sample classification, which is based on a set of classified points [13]. The k-NN models are based on the nearest neighbor rule and are independent of the joint distribution, underlying sample points. The model is widely used to predict student retention, performance, etc. [12].

Furthermore, the k-NN model is a way for grouping samples based on isolate training illustrations in the quality margin. K-NN model clusters sample into a trial experiment and a test case. For the category of the trial case, the K nearest (in Euclidean distance) study selected samples are present, and the majority selections set the grouping with ties altered at random. If ties exist in the  $k^{\text{th}}$  Euclidean distance, all the objects are accounted for in the analysis.

Assumptions in k-NN: k-NN presumes that the sample is in a trait in the Euclidian space. Furthermore, the data sections are contained in a cadent space. The sample could be a unit vector or probably even a scalar vector.

Every test case data has a group of vectors and category identification correlated to each vector. In this instance, it could assume either positive or negative for the respective classes. Nonetheless, KNN can perform significantly well with an inconsistent number of groups.

Taking in this instance one number "k", this k value determines the number of neighbors (assuming neighbors are selected considering the Euclidean distance) influence the grouping. This normally is an odd number if and only if the total groups are 2. When k is 1, in such a case the algorithm is referred to as the nearest neighbor algorithm [29].

#### Illustration

Take  $x_i$  to be original data having  $p$  features ( $x_{i1}, x_{i2} \dots x_{ip}$ ,  $n$  being the sum of original samples ( $i = 1, 2, 3 \dots n$ )) let  $p$  be the sum of features ( $j = 1, 2, \dots, p$ ). The Euclidean space contained in the data  $x_i$  and  $x_l$  ( $l = 1, 2 \dots n$ ) can be written as:

$$d(x_i, x_l) = (x_{i1} - x_{l1})^2 + (x_{i2} - x_{l2})^2 + \dots + (x_{ip} - x_{lp})^2$$

A graphic representation of the KNN model is explained by the Voronoi tessellation whereby neighboring points closest to each data set are defined as:

$$R_i = \{x \in R_p : d(x, x_i) \leq d(x, x_m), \forall i \neq m\}$$

Where  $R_i$  is the Voronoi cell for data  $x_i$ , and  $x$  represents all possible value contained in tessellation cell  $R_i$ .

### 2.1.2 Classification Tree

A classification tree points out regulations for splitting data into cohorts. The initial requirement divides the whole data set into some sections; yet again another requirement may be included in a sub-section, various rules to various sections, comprising a second set of data sections. Hence, therefore, a section could either be split or left intact to form a final cohort. The

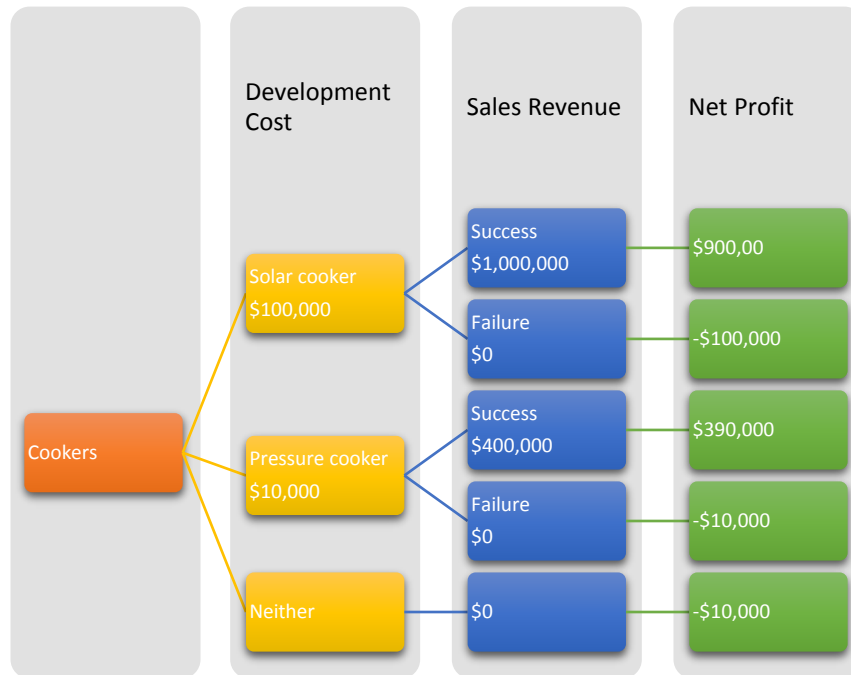


tree shows the initial split into sections as branches resulting from a base and following sections as branches resulting from parts on older branches. The leaves of the tree are the ultimate cohorts, the nodes that are not split. Due to degenerate explanation, trees are usually constructed upside down, emulating an organizational chart. For this model to be significant, the figures in a leaf should coincide regarding another target measure, such that the model represents the dissociation of a mixture of data into specified cohorts [30].

### **Illustration**

**Product Outcome.** In order to consume some short-term excess output capacity at its Arizona manufacture, Distinct Items Industry is weighing a brief production spell for either of two new items, a solar cooker or a pressure cooker. The demand for each of the items is determined if the items efficiently be manufactured. Nonetheless, there is a probability there will be some challenges in the manufacture process.

Income of \$1,000,000 would be reported from disposing of the solar cooker, and income of \$400,000 would be recorded from disposing of the pressure cooker. These figures are the total of manufacturing cost but are exclusive of development cost. If development goes south for an item, then no sales will be recorded, and the development cost will not be incurred. This cost would rise to \$100,000 for the solar cooker and \$10,000 for the pressure cooker.



**Figure 0.1**

Classification tree of product outcome for solar cooker and pressure cooker

### 2.1.3 Random Forest

This model is a combination of decision trees that give out a prediction value, survival applies to this instance. Every decision tree is modeled by employing a random subsection of the test sample. After testing the model, each test row is run through it to obtain a prediction in form of an output. This model requires floats to be used as input variables and as such conversion of all the strings is necessary and there should be no missing data. Therefore, Random forests refers to a model of the entire system of random decision trees that are essential in predictive modeling for regression, classification and analyses, that function by forming an array of classification trees at test time and releasing the group that appears most frequently of the groups or average forecast (regression) of the particular trees.

In regression situations, Random Forests are constructed by forming simple trees, with each of them able to produce a numerical response figure. The independent variable is randomly picked from the entire distribution and for all objects [28]. Formula for Mean square error of a Random Forest is as follows:

$$\text{mean error} = (\text{observed} - \text{tree response})^2$$

The expectations of the model coefficients are assumed to be the mean of the predictions of the objects:

$$\text{Random forest predictions} = \frac{1}{K} \sum K^{th} \text{ tree response}$$

Where  $k$  = runs over the particular objects in the forest.

The test algorithm for random forests utilizes the standard system of bootstrap aggregating, or attaché, to random trees.

Let a test data  $X = (x_1, \dots, x_n)$  with responses  $Y = (y_1, \dots, y_n)$  bagging several times ( $z$  times) chooses a random simple variable with replacement of the test data and plots items to the data:

Whereby:  $Z = (1, \dots, z)$

Data, with replacement,  $n$  test samples from  $X, Y$ ; labeled  $X_z Y_z$ .

Test a classification or regression tree  $f_b$  on  $X_z Y_z$ .

What follows, predictions for unnoticed object  $x'$  are conducted by summing the predictions from all the single regression objects on  $x'$ :

$$f = \frac{1}{Z} \sum f_z(x')$$

### 2.1.4 Binomial GLM

Presented a situation with  $n$  independent values thereafter summing all the  $Y$  's, we obtain a binomial distribution with:

$$mean = n\pi$$

$$Variance = n\pi(1 - \pi)$$

In this case, our attention is on parameter  $\pi$ .

There exist several models of the parameter  $\pi$  that usually differ with respect to their corresponding explanatory variables: for instance  $(x_1, x_2, \dots, x_p)$ . So as to illustrate that  $\pi$  fluctuates with respect to the independent variables, the following arise:

$$\pi(x) = P(Y = 1 | x_1, x_2, \dots, x_p)$$

Since  $Y_i \sim \text{Binomial}(n_i, p_i)$  and we intend to predict the model proportions  $Y_i = n_i$  this yields

$$E(Y_i = n_i) = p_i \text{ With } var(Y_i = n_i) = \frac{1}{n_i p_i (p_i)}$$

This yields a variance function of  $v(u_i) = u_i(1 - u_i)$

Applying natural logs after link function we obtain

$$g(u_i) = \text{logit}(u_i) = \log \frac{u_i}{1 - u_i}$$

Binomial general linear model is the most suitable when the dependent variable is categorical in nature with two probable outcomes (binary results) i.e. 1 and 0. Categorical variables could be shown by way of an indicator variable  $Y_i$  considering the values of 0 or 1, and performed by modeling a binomial distribution bearing probability  $P(Y_i = 1) = i$ . Logit transformations and logistic models mold this probability mass function as a function of a single or several explanatory variables [27].

### 2.1.5 Bayesian Neural Network

Bayesian Neural networks are nonlinear complicated modeling systems that are capable of modeling technical functions. These can be used in tackling issues of forecasting, grouping, and maintenance in a big field of specialization such as corporate world, social science, medical field, general science, and engineering.

Neural networks are employed when the actual form of the association between predictor and explanatory is not revealed. One fundamental characteristic of the neural networks is evident from the association between inputs and output through case testing. There exist three kinds of case testing in neural networks employed by various networks, supervised and unsupervised testing, with the most common one being reinforcement learning .

There exists an inferential supposition in neural network modeling whereby the input parameters are associated with the output by a non-discrete and differential function. In testing the network, the objective is to imitate the behavior of the equation under probe.

A commensurable sophisticated three-course network with one concealed course, by use of a hyperbolic tangent, functions in the nature an equation and is capable of mimicking any function of the kind [26].

The neural network model is at this moment a non-complex flexible function that can be modeled to whichever data set by changing the criterions or weights. It is normal for the function to be the total of many hyperbolic tangent functions.

The dependent variable is written as a linear summation of all their functions,  $h_i$ , multiplied by the weights  $W_i$  and the bias  $\theta$  :

$$y = \sum W_i h_i + \theta$$

In the event of a hyperbolic tangent formulation the inclusive equations for an object  $i$  in the concealed course is represented by:

$$h_i = \tanh\left(\sum W_{ij} X_i + \theta_i\right)$$

With weights  $W_{ij}$  and biases  $\theta_i$ . So as to analyze the weights, independent variables are linearly normalized by introducing natural logs within the range  $\pm 0.5$  applying the normalization equation,  $x_j = \frac{x - x_{min}}{x_{max} - x_{min}} - 0.5$ , where  $x$  represents the value of the independent variable with  $x_j$  being the resulting normalized number.

Testing the network is obtained by changing the coefficients to tally the equations of the sample by applying 'backpropagation' slope declension minimization techniques, to optimize an objective function. With effective minimization due diligence has to be observed to prevent over fitting and suitable to local minima.

This is obtained by evaluating and minimizing an objective function:

$$m(w) = \beta Ed + \alpha Ew$$

That encompasses an error term ( $Ed$ ) to analyze how better the model is and regularization term ( $Ew$ ) to contain significant weights, for instance:

$$m(w) = \beta \left( \frac{1}{2} \sum (t_i - y_i)^2 \right) + \alpha \left( \frac{1}{2} \sum w_i^2 \right)$$

Where  $\beta$  and  $\alpha$  are sophisticated parameters that majorly affect the technicality of the model,  $t_i$  and  $y_i$  being the estimates and assimilate output figures for a single example predictor from the case testing sample  $x_i$ .

Cross corroboration further shields against over modeling. Dividing the data into testing and case training cohorts enable different sub fittings to be analyzed by their productivity in

estimating the concealed course. An amalgamation of the selected  $n$  fittings as grouped are enjoined to comprise a committee fitting. The forecast being the mean of the predictions of the multi-fittings and the shift in mean deviations of predictions involving the variance of the committee's forecasted distribution together with the mean of the variances:

$$var\delta^2 = \frac{1}{n} \sum \delta_y^{l^2} + \frac{1}{n} \sum (y_i - y)^2$$

With  $n$  as the value of multi-fittings. The errors for the various groupings can be estimated and the one with the minimum calculated forecast error utilized.

Once tested the network incorporates synergies between the predictors due to the nonlinearity of the objective function. The quality of the synergies are kept in the weights, nonetheless, as opposed to multiple regression they are complicated and cannot be directly interpreted. Therefore, trends are determined by making predictions from the fit model. Predictions are followed by a signal of the uncertainty which relies on the confidence of the modeled data for the prediction, as well as the level of adequacy obtained from the sample which is presumed be bearing a Gaussian distribution [26].

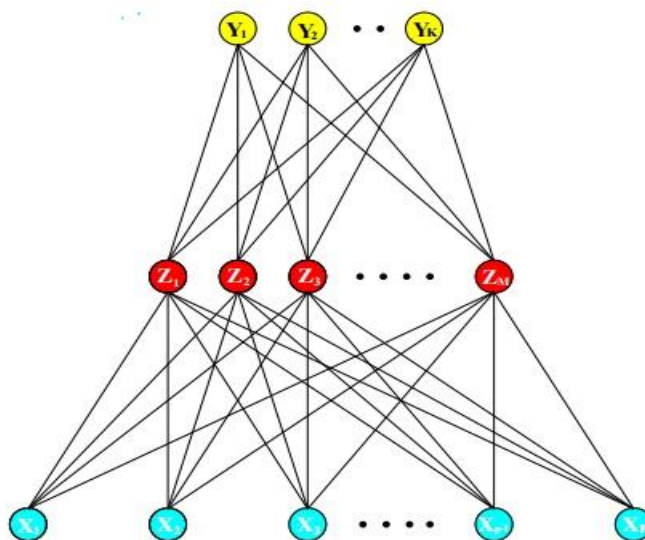
### **2.1.6 Artificial Neural Networks**

The artificial neural networks, especially those used in the field of physics, make up tools that are largely used in sciences of management and the methods of sophisticated and advanced forecast. Artificial neural networks are still much solicited and receive interest for their principle that is easy to assimilate. Even though the concept of ANN is taken from nature and biology, we still have to make assumptions about the variables that we use to build it.

Artificial neural networks are forecasting methods that are based on simple mathematical models of the brain. They allow complex nonlinear relationships between the response variable

and its predictors. A neural network can be thought of as a network of “neurons” organized in layers: the predictors (or inputs) forming the bottom layer, the forecasts (or outputs) forming the top layer, with intermediate layers containing “hidden neurons”. Algorithms help ANNs to form themselves. They do this using relationships between variables and by considering the history of data in a way similar to how the human brain operates.

Artificial neural networks, as a class of models, have independently been developed in two separate fields – artificial intelligence and statistics [14]. Hastie et al. (2009) defines the neural networks method as a two-stage regression nonlinear model (or classification), which represents the relationships of the chosen variables through a network diagram. The model is usually composed of multiple layers, and depends on a large number of puts, that are usually unknown.



**Figure 0.2**

Schematic diagram of a hidden layer in feed-forward neural network [14]



Numerous studies have utilized the neural network based models to study student retention and performance [15]. Alkhasawneh and Hobson (2011) used a feed-forward backpropagation network to construct a model used to predict science and engineering students' retention. The models classified students into three risk groups: at-risk, intermediate and advanced. The model delivered a 70.1% accuracy prediction rate [16].

## **Chapter 3: Research Methodology**

### **3.1 Data Used in the Research**

The data used in this research were obtained through the Admissions Office at SCSU. This data includes the following categorical data which were used to identify the input variables for the models. The following subsections use example variables from the set of variables available, to better explain the categorization.

#### **3.1.1 Readiness for a College Education**

The first category considered was the likelihood of a student to retain in school relative to the applicant's level of readiness when applying for college. For example, the variable `AdmitDaysBeforeTerm` can be used as a parameter to measure this likelihood. It would entail the number of days starting from the day the student receives admission to college, to the academic year start date. It is possible to assume that the earlier the student gets admission, the earlier s/he would have started the application process. And this would help us conclude that the student's decision to get a college education was well thought out.

#### **3.1.2 Academic Capacity**

The variable `ACE_Student`, which would mean a student who is provisionally admitted and needs more preparation for college studies. This can be used to determine the academic capacity of students. It is more likely that students who are not provisionally admitted will perform better in their college education. This would lead to academic satisfaction, a factor that is directly linked to student retention at any stage.

### **3.1.3 Financial Stability**

There are a few variables that indicate financial stability of the student. Apart from the variables related to Federal student aid or scholarships provided by the college there are other variables such as, MilesToSCSU, that is, the number of miles to SCSU from the applicant's home. If a student thinks that s/he is too far from school, in which case the student must rent an apartment closer to the school or seek university housing, this could impose an additional financial stress on the student, and could eventually lead to the student dropping out from college.

### **3.1.4 Other Variables**

These are variables that do not represent any of the above mentioned categories but are still important for the models. Some examples are as follows.

**ClosestToSCSU:** This parameter determines if the closest college to the applicant's home is SCSU or not. This could later have an effect if the student decides to transfer to a closer school. In that case, SCSU fails to retain the student since the student has another school in closer proximity than SCSU.

**FirstGenStudent:** Being a first generation college student could have a negative effect on continuing college studies. Reasons for this could be lack of guidance or motivation from family.

**FAFSA number:** This is the number that the applicants mark SCSU in the precedence order of the FAFSA list. If this number is low, it is likely that the student will try to transfer to a school which s/he prefers more.

**Age:** The student's age could be a determining factor when deciding if s/he wants to continue studying. It is highly likely that the student would have commitments and

responsibilities other than her/his education, the older in age the student is. This could be another reason why adult students drop out from college.

### **3.2 Data Cleaning Process**

In this section, the steps that were taken to clean data is explained. This includes identifying and handling outliers, handling missing values, etc.

First the data was converted into CSV format files to have three different CSVs containing the data of admission information, term results information, courses and grades from DataMart tables. All three of these files are linked through the student IDs. After data input files were prepared, data cleaning was done in the following steps:

1. Data type conversion (character to numeric, character to factor)
2. Aggregated data to student ID level
3. Identified outliers and removed them appropriately
4. Cleaned/ removed missing values
5. Merged dataset into one dataset

Admission information and term results information datasets were compiled together for preprocessing while the third dataset courses and grades from DataMart table were treated separately. Then all datasets were merged at the end of the process.

### **3.3 Data Type Conversion**

Having read the csv files, as the first step of the data preparation process, all the character type data were converted in to numeric data. The R statements used to carry out these actions were as follows.

Read data were assigned to two variables, df1 and df2, for further processing (as seen below).

```
df1 <- read.csv("../data/Adminssion InfoCSV.csv", sep=",", dec = ".", stringsAsFactors = F)

df2 <- read.csv("../data/Term Results dataCSV.csv", sep = ",", dec=":", stringsAsFactors = F)
```

Next the data in df1 and df2 were scanned to check for character type data and were converted to numeric type. Upon finishing the conversion, the datasets were saved in R data files to use when needed.

```
for(i in 2:45){
  if(class(df1[[i]])=="character"){
    df1[[i]] <- as.numeric(gsub(",", ".", df1[[i]]))
  }
}

for(i in c(2:6,8:27)){
  if(class(df2[[i]])=="character"){
    df2[[i]] <- as.numeric(gsub(",", ".", df2[[i]]))
  }
}

saveRDS(df1, "../data/df1.Rds")

saveRDS(df2, "../data/df2.Rds")
```

Converting character variables to factors were done after merging the two data sets together. The following R scripts were used for that purpose.

```
## Convert character variables to factors

varClasses <- sapply(names(df), function(x) {class(df[[x]])})

charVars <- names(varClasses)[varClasses=="character"]

for(i in charVars){df[[i]] <- factor(df[[i])}
```

### 3.4 Aggregate Data to Student ID Level

Next a single data set was created by combining two datasets in df1 and df2 using student ID. This was done by performing a left join on df1 and df2 by student ID column of the dataset. R package “dplyr” was used for data manipulation.

```
require(dplyr)

df1$in1 <- 1

df2$in2 <- 1

df <- left_join(df1, df2, by = "StudentId") %>%

  filter(in1==1 & in2 == 1 & T1_Enrolled == 1) %>%

  select(c(StudentId:Gender, T1_TermGPA, T2_TermGPA,

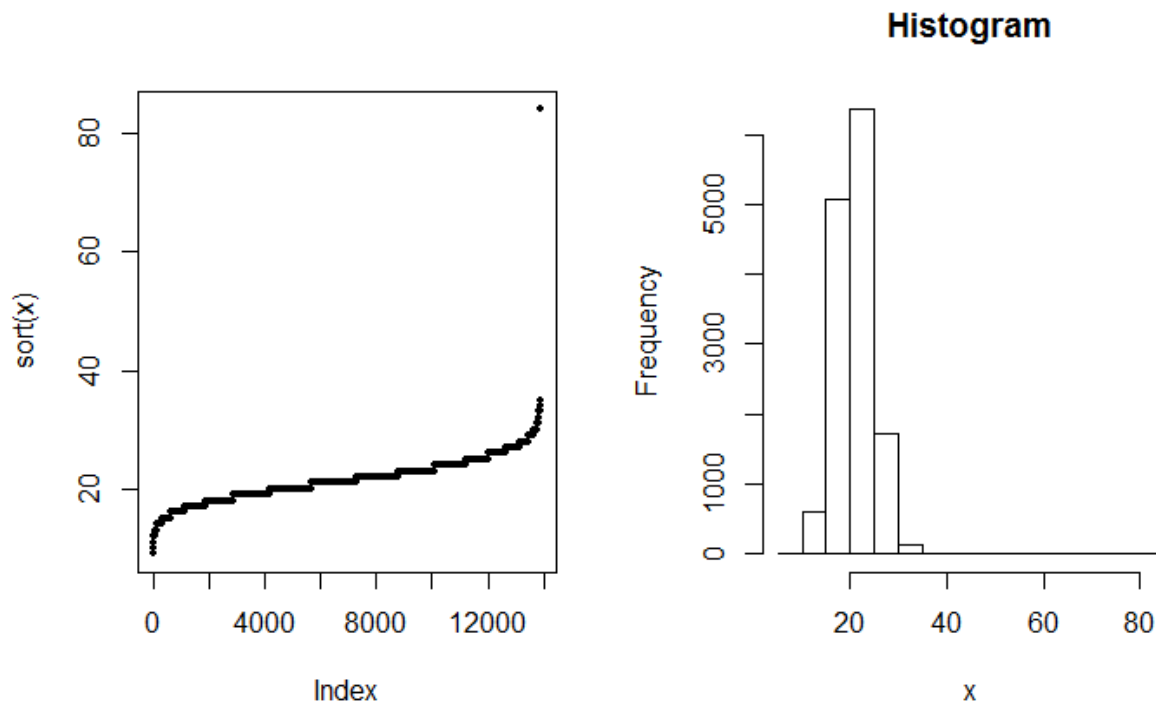
           T1_TermLocalCreditsAttempted, T2_TermLocalCreditsAttempted,

           T1_TermLocalCreditsEarned, T2_TermLocalCreditsEarned, T3_Enrolled))
```

### 3.5 Removing Outliers

After aggregating the two dataset, outliers were identified in the current dataset. The following plots and histograms are used to demonstrate the existence of outliers for each of the listed variables. Plots use a sorted list of data as the Y-axis and histograms use the frequency of data points as Y-axis.

### 3.5.1 ACT Composite Scores



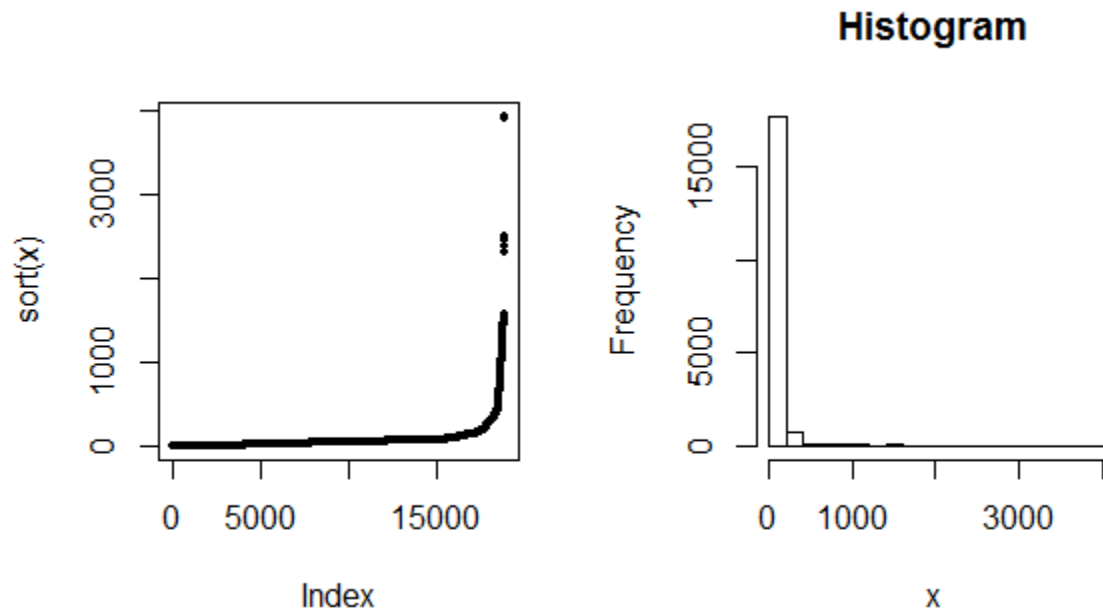
**Figure 0.1**

Histogram and data plot for the ACT\_Composite scores variable

There were a few outliers identified for variable ACT\_Composite scores, as shown in Figure 3.1. The outliers of the ACT\_Composite scores were replaced by the maximum value of the said dataset excluding all the outliers. The maximum value without the outliers was identified as 40. The R statement for replacing outliers with the value 40 is as follows.

```
df$ACT_Composite[df$ACT_Composite>40]<-40
```

### 3.5.2 MilesToSCSU



**Figure 0.2**

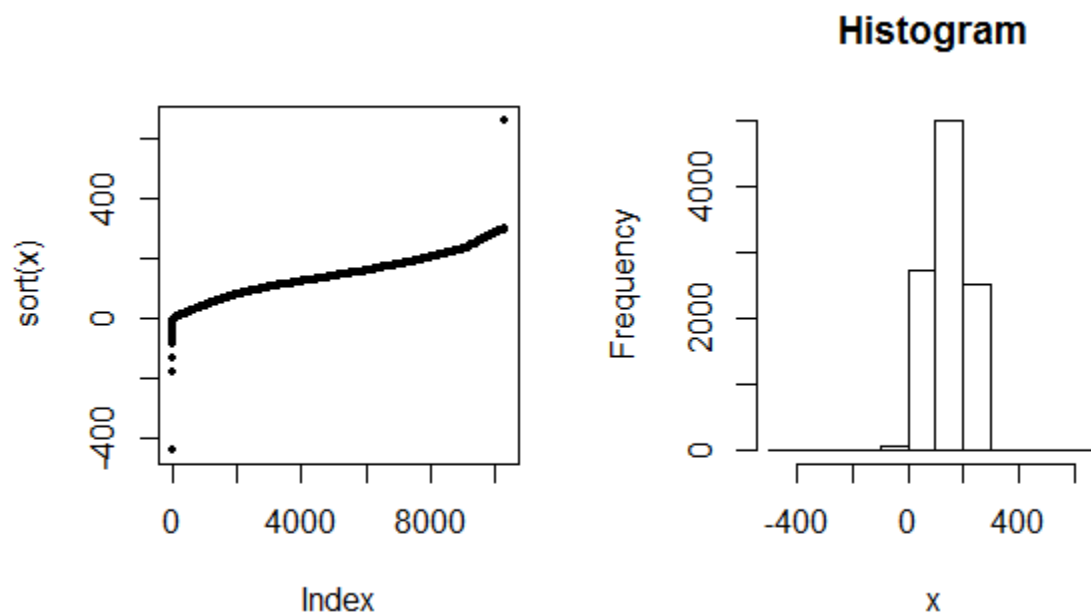
Histogram and data plot for the MilesToSCSU variable

The outliers for MilesToSCSU, as shown in Figure 3.2, were rectified using a log function on the variable. Before the log function was calculated, all the data points were increased by one to avoid  $\log(0)$  resulting in NA values after the calculation. The R statement used is as follows.

```
df$MilesToSCSU <- log(df$MilesToSCSU+1)
```



### 3.5.3 HousingAppDaysBeforeTerm



**Figure 0.3**

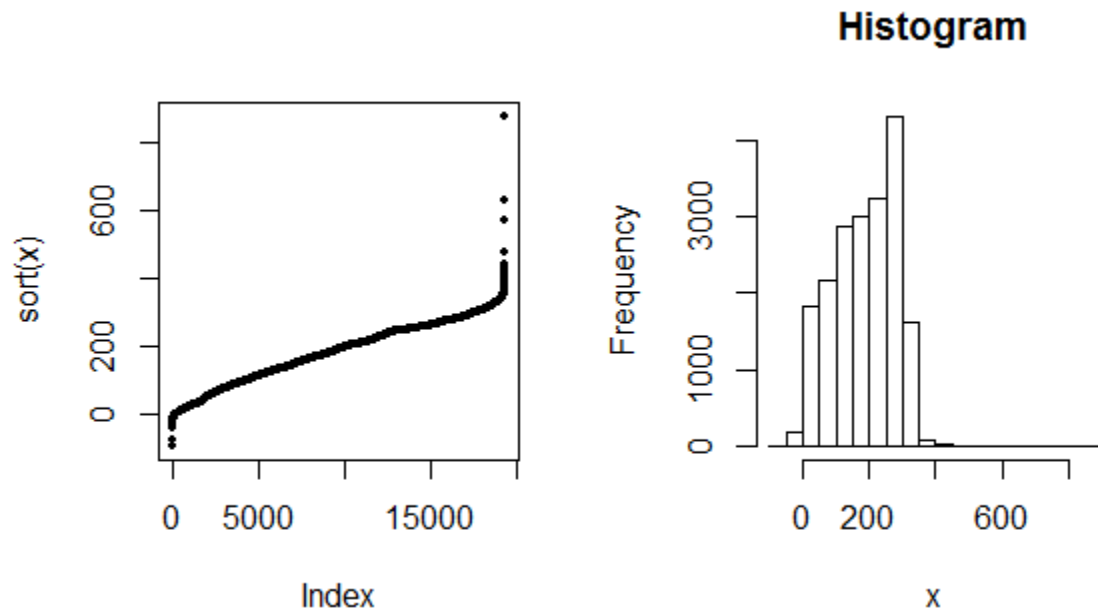
Histogram and data plot for the HousingAppDaysBeforeTerm variable

The HousingAppDaysBeforeTerm variable had outliers both from positive and negative sides (see Figure 3.3). Three hundred (300) was identified as the maximum value without the outlier values and -100 was identified as the minimum. All the positive outliers were replaced with 300 and all the negative outliers were replaced with -100.

```
df$HousingAppDaysBeforeTerm[df$HousingAppDaysBeforeTerm>300] <- 300
```

```
df$HousingAppDaysBeforeTerm[df$HousingAppDaysBeforeTerm < -100] <- -100
```

### 3.5.4 AdmitDaysBeforeTerm



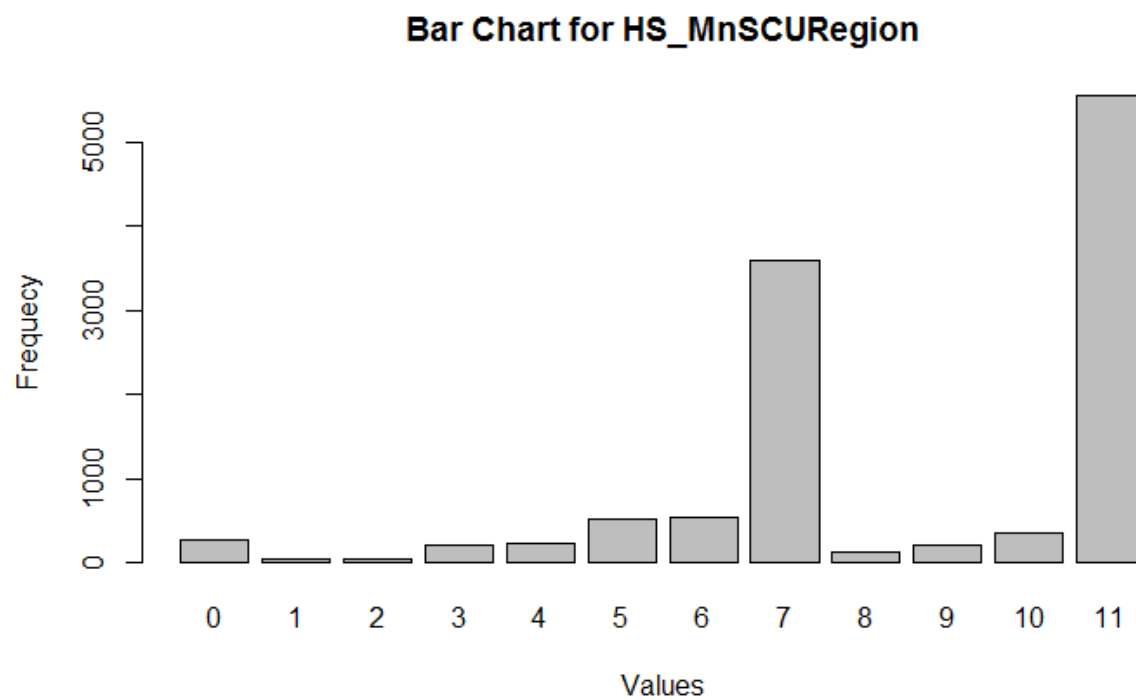
**Figure 0.4**

Histogram and the data plot of the AdmitDaysBeforeTerm variable

All the values above 500 days were identified as outliers, as shown in Figure 3.4, and were replaced with a value of 500 days.

```
df$AdmitDaysBeforeTerm[df$AdmitDaysBeforeTerm>500] <- 500
```

### 3.5.5 HS\_MnSCURegion

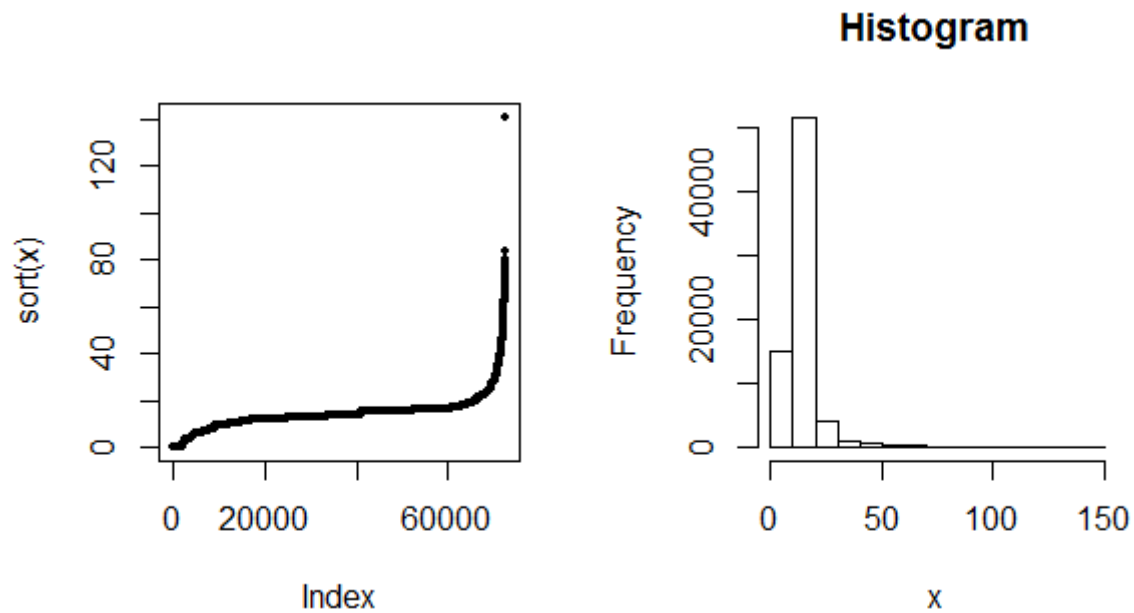


**Figure 0.5**

Bar chart for HS\_MnSCURegion variable

HS\_MnSCURegion has a value of 99 for students who have a high school region outside of Minnesota (see Figure 3.5). Because this is a categorical variable, value 99 is not an outlier. For this reason these values were not replaced in the distribution.

### 3.5.6 T1\_TermCumulativeLocalCreditsEarned



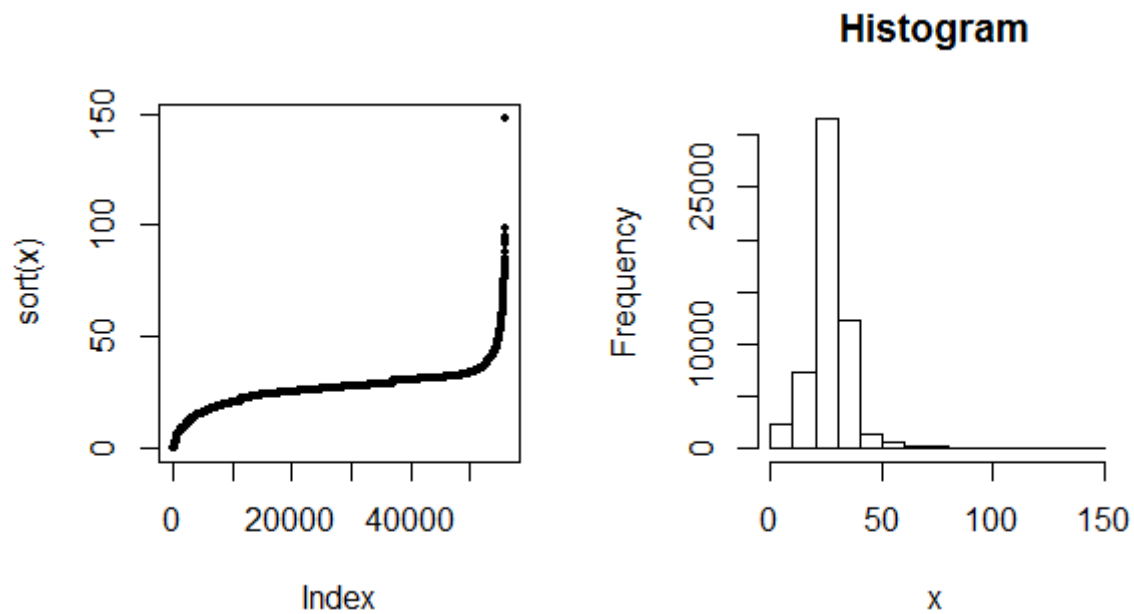
**Figure 0.6**

Histogram and data plot of the T1\_TermCumulativeLocalCreditsEarned variable

The outliers, as shown in Figure 3.6, which were identified for this variable were replaced by the maximum value of the dataset when taken without the outliers. The said value was identified as 84.

```
dd$T1_TermCumulativeLocalCreditsEarned[dd$T1_TermCumulativeLocalCreditsEarned
==141] <- 84
```

### 3.5.7 T2\_TermCumulativeLocalCreditsEarned



**Figure 0.7**

Histogram and data plot of the T2\_TermCumulativeLocalCreditsEarned variable

The outliers which were identified for this variable (see Figure 3.7) were replaced by the maximum value of the dataset when taken without the outliers. This was identified as 99.

```
dd$T2_TermCumulativeLocalCreditsEarned[dd$T2_TermCumulativeLocalCreditsEarned
>99] <- 99
```

### 3.6 Cleaning Missing Values

This section will discuss how the missing values were handled in the dataset. Following table (see Table 3.1) displays the number of missing values for each variable.

**Table 0.1**

Number of missing values for each variable in the dataset

Variable	Number of missing values
ACT_Composite	5456
HS_GPA_4Scale	4179
HSPct	5653
QPP	5
EnglTotal	3229
ClosestToSCSU	464
MilesToSCSU	464
HousingAppDaysBeforeTerm	9044
FAFSADaysBeforeTerm	2690
FAFSADayOfYear	2690
FAFSA_Nbr	2690
TotalCRHR_TRSF	8312
TransferGPA	8484
TransferQP	8312
HS_MnSCURegion	502
T1_TermGPA	299
T2_TermGPA	2502
T2_TermLocalCreditsAttempted	2249
T2_TermLocalCreditsEarned	2249

The following user specified R function was used to replace the missing values for each variable.

```
replaceMissing <- function(x, newvalue){
  x[is.na(x)] <- newvalue
  return(x)
}
```

Rather than removing the variables with missing values, those were replaced by a suitable value. For each variable which had over 5% missing values, which is approximately over 500 missing values, a missing value indicator variable was introduced. This is to minimize the bias which may introduce by imputing the missing values in the distribution. These indicator variables are Boolean variables, which have False (0) where value was missing, and True (1) for others.

### 3.6.1 QPP

There were only 5 missing values present for the QPP variable. The missing value of this variable was replaced with the mean value of the QPP dataset. Following table (see Table 3.2) displays the summary statistics of the QPP variable before handling the missing values.

**Table 0.2**

Summary statistics of the QPP variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
24.87	62.44	71.52	72.93	83.81	122.3	5

The use of `replaceMissing` R function to replace the missing values for the QPP variable is shown below.

```
df$QPP <- replaceMissing(df$QPP, mean(df$QPP, na.rm=T))
```

### 3.6.2 T1\_TermGPA

Having observed the T1\_TermGPA variable, it was evident that the variable had 299 missing values and that out of these records 85% of them were not retained. These missing values had a minimum impact on retention. Therefore the missing values were replaced with a value of 0. Following table (see Table 3.3) shows the summary of statistics.

**Table 0.3**

Summary statistics of the T1\_TermGPA variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0	2.09	2.84	2.625	3.38	4	299

Following is the R statement that was used to replace the missing values.

```
df$T1_TermGPA <- replaceMissing(df$T1_TermGPA, 0)
```

### 3.6.3 ClosestToSCSU

It was observed the ClosestToSCSU variable had 464 missing values. Since this is a Boolean variable all the missing values were considered as not reported and substituted with a zero. Below is the R statement which was used for the replacement.

```
df$ClosestToSCSU <- replaceMissing(df$ClosestToSCSU, 0)
```



### 3.6.4 MilesToSCSU

The MilesToSCSU variable had 464 missing values. After considering the retained records out of these missing value records, the mean value of the dataset was substituted for the missing values. Following table (see Table 3.4) shows summary statistics of the MilesToSCSU variable after applying the log function to eliminate outliers.

**Table 0.4**

Summary statistics of the log (MilesToSCSU) variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0	2.835	3.875	3.449	4.249	8.275	464

Below is the R statement used for replacing the missing values of the adjusted MilesToSCSU variable.

```
df$MilesToSCSU <- replaceMissing(df$MilesToSCSU, mean(df$MilesToSCSU,
na.rm=T))
```

### 3.6.5 FAFSADaysBeforeTerm

The FAFSADaysBeforeTerm variable had 2690 missing values. Having looked at the effects the records with missing values in the FAFSADaysBeforeTerm variable have had on retention, it can be determined that the percentage retained have close values (Table 3.5). While 69.5% of the time a student was retained when a value was present for FAFSADaysBeforeTerm variable, the same number was at 75.7% when the variable had a missing value.

**Table 0.5**

Retained probabilities for the FAFSADaysBeforeTerm with respect to the availability of data points

		T3_Enrolled(Retained)	
		0	1
FAFSADaysBefforeTerm	Has value	0.304763	0.695237
	Missing value	0.2423792	0.7576208

Furthermore, by looking at the boxplot of the FAFSADaysBeforeTerm vs T3\_Enrolled (Figure 3.8), it can be concluded that the FAFSADaysBeforeTerm variable has very low impact on the retention. Considering these facts, the missing values were replaced by the mean value and a missing value indicator variable was introduce since this variable had more than 5% missing values to it.

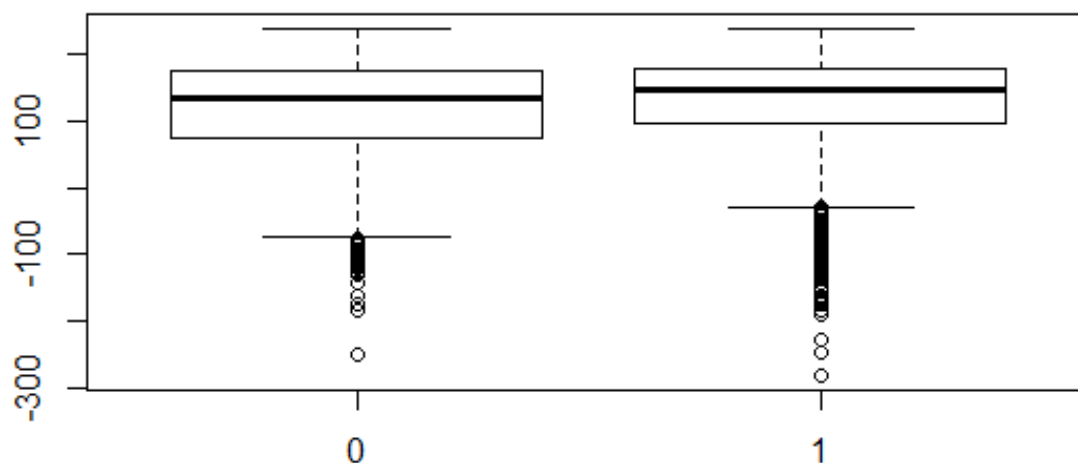


Figure 0.8

Boxplot of the FAFSADaysBeforeTerm vs T3\_Enrolled

Following table (see Table 3.6) shows the summary statistics of the FAFSADaysBeforeTerm variable. Below is the R statement used for replacing missing values.

```
df$FAFSADaysBeforeTerm <- replaceMissing(df$FAFSADaysBeforeTerm,
mean(df$FAFSADaysBeforeTerm, na.rm=T)))
```

**Table 0.6**

Summary statistics of the FAFSADaysBeforeTerm variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
-281	89	145	130.1	178	239	2690

### 3.6.6 FAFSADayOfYear

The FAFSADayOfYear variable had a same number of missing values as the FAFSADaysBeforeTerm variable. The number of missing values were 2690. The behavior and the impact on the retention of this variable is almost the same as the FAFSADaysBeforeTerm variable. Because of this, the same approach which was taken with the FAFSADaysBeforeTerm variable was used with this variable as well. Following table (see Table 3.7) shows you the summary statistics of the FAFSADayOfYear variable.

**Table 0.7**

Summary statistics of the FAFSADayOfYear variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
1	58	91	105.7	147	365	2690

The following statement was used to replace the missing values of this variable with a zero.

```
df$FAFSADayOfYear <- replaceMissing(df$FAFSADayOfYear, mean(df$FAFSADayOfYear,
na.rm=T)))
```

### 3.6.7 FAFSA\_Nbr

This variable also had the same number of missing values as the other two FAFSA variables and behaved in the same way. But as this variable is a categorical variable, missing values were replaced with a value of 0. Following table (see Table 3.8) shows the summary statistics of the variable.

**Table 0.8**

Frequency table of the FAFSA\_Nbr variable

Value	0	1	2	3	4	5	6	7	8	9
Freq.	1708	8435	919	286	138	75	41	23	10	5

The following R statement was used to replace the missing values of this variable.

```
df$FAFSA_Nbr <- replaceMissing(df$FAFSA_Nbr, 0)
```

### 3.6.8 EnglTotal

The EnglTotal variable had 3229 missing values. Whenever there is no value for this variable, -1 is assigned to that data point. Because this variable had a high number of missing values a missing value indicator variable was introduced while replacing the missing value with the mean of the distribution. Following table (see Table 3.9) shows the summary statistics for the EnglTotal.

**Table 0.9**

Summary statistics of the EnglTotal variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0	4	4	3.974	4	10	3229

The following R statement is used to replace the missing values of this variable with mean.

```
df$EnglTotal <- replaceMissing(df$EnglTotal, mean(df$EnglTotal, na.rm=T))
```

### 3.6.9 HS\_GPA\_4Scale

HS\_GPA\_4Scale variable had 4179 missing values and were replaced by mean value of the available dataset of the same variable. Following table (see Table 3.10) shows the summary statistics of the HS\_GPA\_4Scale variable.

**Table 0.10**

Summary statistics of the HS\_GPA\_4Scale variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
1.06	2.79	3.157	3.138	3.53	4.97	4179

Following R statement was used to replace the missing values of this variable.

```
df$HS_GPA_4Scale <- replaceMissing(df$HS_GPA_4Scale, mean(df$HS_GPA_4Scale, na.rm=T))
```

### 3.6.10 ACT\_Composite

There were 5456 missing values within the dataset of ACT\_Composite variable. Missing values were replaced by the mean value of the available dataset. Following table (see Table 3.11) shows the summary statistics of the ACT\_Composite variable.

**Table 0.11**

Summary statistics of the ACT\_Composite variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
9	19	21	21.43	24	40	5456

Following R statement was used to replace the missing values with mean.

```
df$ACT_Composite <- replaceMissing(df$ACT_Composite, mean(df$ACT_Composite,
na.rm=T))
```

### 3.6.11 HSPct

This variable had 5653 missing values within its dataset. Missing values were replaced with the mean value of the HSPct distribution. Table 3.12 shows the summary statistics of the HSPct variable.

**Table 0.12**

Summary statistics of the HSPct variable

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0	39.23	56.67	56.16	74.42	100	5653

Following R statement was used to replace the missing values.

```
df$HSPct <- replaceMissing(df$HSPct, mean(df$HSPct, na.rm=T))
```

### 3.6.12 TotalCRHR\_TRSF

This variable had 8312 missing values. Those values were handle by assigning them with the mean value of the variable. Table 3.13 shows the summary statistics of the TotalCRHR\_TRSF variable.

**Table 0.13**

Summary statistics of the TotalCRHR\_TRSF variable

Min.	1st Qu.	Median	3rd Qu.	Max.	NA's
0	14	32	60	250.3	8312

R statement for replacing those missing values is below.

```
df$TotalCRHR_TRSF <- replaceMissing(df$TotalCRHR_TRSF, mean(df$TotalCRHR_TRSF  
, na.rm=T)))
```

### 3.6.13 HS\_MnSCURegion

This variable had 502 missing values. Whenever an HS\_MnSCURegion is not specified the value was set to 0. Table 3.14 shows the summary statistics of the HS\_MnSCURegion variable.

**Table 0.14**

Frequency table of the HS\_MnSCURegion variable

Value	0	1	2	3	4	5	6	7	8	9	10	11
Freq.	267	423	40	201	230	516	542	3585	119	200	345	555

Following R statement was used to replace the missing values.

```
df$HS_MnSCURegion <- replaceMissing(df$HS_MnSCURegion,0)
```

### 3.6.14 Other Variables

All the missing values of TransferQP, TransferGPA, HousingAppDaysBeforeTerm, T2\_TermGPA, T2\_TermLocalCreditsAttempted, T2\_TermLocalCreditsEarned variables were replaced with the mean. Each of these variables had over 5% missing values, therefore a missing value indicator variable was introduced for each variable.

After replacing all the missing data, the following function was used to verify if all the missing data were replaced correctly and if the missing values in the dataset were no longer existent.

```
tmp_missing1 <- apply(dd,2,function(x){sum(is.na(x))})
```

If all the missing values were handled correctly, the tmp\_missing1 R variable should have a value of zero after this function was executed.



## Merging Dataset into One Dataset

The datasets were merged together using student ID as the linking column after cleaning the datasets. The R script which was used to perform the merge is as follows.

```
df$in1 <- 1
ds$in2 <- 1
d <- left_join(df, ds, by="StudentId") %>% filter(!is.na(in1) & !is.na(in2)) %>%
%
select(-c(43,72))
```

Once the dataset was merged, the T3\_Enrolled column was moved to the last position, to set as the target variable. Following is the R script executed for this purpose.

```
tmp <- select(d, StudentId, T3_Enrolled)
d <- select(d, -T3_Enrolled)
d$T3_Enrolled <- tmp$T3_Enrolled
d$T3_Enrolled <- factor(d$T3_Enrolled == 1)
```

At the end of the data cleaning process, the dataset was saved as an R data file so that it is available to use anytime afterwards. This same data file was used to train the predictive models in the next stage.

## Chapter 4: Building Models

This section discusses training the models using the data that were cleaned during the previous stage.

Once the dataset was cleaned, there were 68 variables excluding Student ID and T3\_Enrolled variables. Which means there were 68 covariates that could be used as predictors with T3\_Enrolled being the target variable. The distribution of the target variable is shown in the Table 4.1.

**Table 0.1**

Distribution of T3\_Enrolled variable

T3_Enrolled	Frequency	Rate
TRUE	10968	0.7067
FALSE	4551	0.2933

### 4.1 Confusion Matrix

Confusion matrix is used to measure the performance of a predictive model. These are often used in classification models. A confusion matrix is composed of true positives, true negatives, false positives, and false negatives, which are statistics calculated using predicted values and actual values.

**Table 4.2**

Example Confusion Matrix

n=1000		Prediction	
		FALSE	TRUE
Actual	FALSE	175	35
	TRUE	115	675

True positives: The case where the predicted value and the actual value is true, is a true positive. True positives represents the number of these cases in a population. In Table 5.1, 675 is the number of true positives out of a population of 1000.

True Negatives: The case where the predicted value and the actual value is false, is a true negative. True negatives represents the number of these cases in a population. In Table 5.1, 175 is the number of true negatives out of a population of 1000.

False Positives (type I error): The case where the actual false value is predicted as true, is a false positive. False positives represents the number of these cases in a population. In Table 5.1, 35 is the number of false positives out of a population of 1000.

False Negatives (type II error): The case where the actual true value is predicted as false, is a false negative. False negatives represents the number of these cases in a population. In Table 5.1, 115 is the number of false negatives out of a population of 1000.

#### **4.1.1 Accuracy**

Accuracy is how probable it is on average for the prediction of the model to be correct. This is calculated as the proportion of the number of correctly classified cases to the total number of cases. Correctly classified cases are the total of true positives and true negatives. The following equation represents the calculation of accuracy,  $n$  being the total number of cases.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{n}$$

As an example, the accuracy calculated using the confusion matrix shown in Table 5.1 would be  $(175+675)/1000$ , which is equal to 0.85.

### 4.1.2 Sensitivity

Sensitivity is how probable it is to classify a case as true when it is actually true. This is calculated as the proportion of correctly classified true cases within actual true cases. The following equation represents the calculation of sensitivity.

$$\text{Sensitivity} = \frac{\text{True Positives}}{\text{Actual Trues}}$$

As an example, sensitivity of the model represented with confusion matrix shown in Table 5.1 is  $675/(115+675)$ , which is 0.8544.

### 4.1.3 Specificity

Specificity is how probable it is to classify a case as false when it is actually false. This is calculated as the proportion of correctly classified false cases within actual false cases. The following equation represents the calculation of specificity.

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{Actual Falses}}$$

As an example, specificity of the model represented with confusion matrix shown in Table 5.1 is  $175/(175+35)$ , which is 0.8334.

### 4.1.4 Positive Predictive Value

Positive predictive value is how probable it is for a case to be actually true when the model predicts it to be true. In the context of this thesis problem, the probability would be how likely it is that a student will retain, when the model identifies the student as a student who will retain. This is calculated as the proportion of correctly classified true cases within predicted true cases. The following equation represents the calculation of positive predictive value.

$$\text{Pos. Pred. Value} = \frac{\text{True Positives}}{\text{Predicted Trues}}$$

As an example, the positive predictive value of the model represented with confusion matrix shown in Table 5.1 is  $675/(675+35)$ , which is 0.9507.

#### 4.1.5 Negative Predictive Value

Negative predictive value is how probable it is for a case to be actually false when the model predicted it to be false. In the context of this thesis problem, it is how likely it would be for a student to dropout, when the model identifies the student as a student who will dropout. This is calculated as the proportion of correctly classified false cases within predicted false cases. The following equation represents the calculation of negative predictive value.

$$\text{Neg. Pred. Value} = \frac{\text{True Negatives}}{\text{Predicted Falses}}$$

As an example, the negative predictive value of the model represented with confusion matrix shown in Table 5.1 is  $175/(175+115)$ , which is 0.6034.

## 4.2 Preprocessing

Sixty six point seven percent (66.7%) of the available data were used to train the selected model. The other 33.3% of the data were used as the test dataset to verify the validity of the final trained model.

Choosing a model was done by comparing the number of models for their accuracy, sensitivity, specificity, positive predictive value, and negative predictive value. In the comparison stage, the models were built using the same sample dataset. For this purpose, a randomly selected dataset of 3000 records (20% of the population) were used.

```
## Take a smaller sample of original data for model comparison
set.seed(2234)
```

```
inSmp <- sample(1:nrow(d), 3000, replace = F)
s <- d[inSmp, -1] #remove ID
```

Out of the sample dataset one third of data were used as the test dataset to test the built models and the rest were used to train the models.

```
# train and test set
```

```
inTrain <- createDataPartition(y=s$T3_Enrolled, p=.667, list=F)
trn <- s[inTrain,]
tst <- s[-inTrain,]
```

### 4.3 Identifying Near-Zero Variance Variables

In some situations, the data have predictors that only have a single unique value (i.e., a “zero-variance predictor”). For many models (excluding tree-based models), this may cause the model to crash or the fit to be unstable. Similarly, predictors might have only a handful of unique values that occur with very low frequencies. These predictors may become zero-variance predictors when the data are split into cross-validation/bootstrap sub-samples or when a few samples may have an undue influence on the model. These “near-zero-variance” predictors may need to be identified and eliminated prior to modeling [24].

```
# remove zero variance columns and ids
```

```
nsv <- nearZeroVar(trn, saveMetrics=F)
trn <- trn[, -nsv]
tst <- tst[, -nsv]
trn$Cohort <- factor(trn$Cohort)
tst$Cohort <- factor(tst$Cohort)
```

After removing the near zero variance variables, there were 64 variables left as predictors. Variables CohortTermEnrolled, IntlFlag, Top10Flag, and TotalCRHR\_TRSF were identified as near zero variance variables.

#### 4.4 Principal Component Analysis

In some cases, the use of principal component analysis (PCA) is needed to transform the data to a smaller sub-space where the new variables are uncorrelated with one another. Using a number of correlated predictors may lead to over fitted models. PCA transformation eliminates this problem as it results in uncorrelated variables which will improve the results of the predictions.

Principal Component Analysis is carried out usually to achieve the following:

1. Extract the most important information from the data table.
2. Compress or reduce the size of the data set by only keeping important information.
3. Simplify the description of the data set.
4. Analyze the structure of the observations and the variables.

When Principal Component Analysis is carried out, a new set of variables is computed. These new variables, which are called Principal Components are attained as linear combinations of the original variables. (Principal Component Analysis Hervé Abdi · Lynne J. Williams)

After removing the near-zero variance variables from the data set, a PCA was performed to compress the data set.

```
# reduce n of columns with pca
preProc <- preProcess(trn[, -c(1, ncol(trn))], method="pca", thresh = .95 )
trnPre <- predict(preProc, newdata = trn)
tstPre <- predict(preProc, newdata = tst)
```

The above R code was run to compress the data set using PCA. The compressed set of variables was stored in preProc R variable while the trnPre variable held the compressed sample data set for training models while tstPre held the same for testing.

Note that the threshold of 95% was used for PCA, which means that 95% of information out of the original data were held. The structure of the Principal Components were stored in variable preProc so that the same structure could be applied to both the training and the test sample (and, furthermore, to the total dataset after our initial model testing). The trnPre R variable held compressed sample data set for training models and tstPre held the same for testing. After the Principal Code Analysis, the number of variables were reduced from 64 to 35.

#### **4.5 Training Models**

Once the preprocessing of data was done, each of the models were built to use in the comparison stage. The models which were built in this phase were KNN (K Nearest Neighbor), Classification Tree, Random Forest, Binomial GLM, Neural Network, and Bayesian Neural Network. After building each model, accuracy, sensitivity, specificity, positive prediction value, and negative prediction value was calculated using the confusion matrix (refer comparing models phase for description of these terms).

Building models were done in three different phases. First, the models were build using PCA compressed data, then using the entire variable set. These first two phases were done to compare and identify which model using which set of variables had the best statistics and to test the hypothesis that was made before, that is if the Principal Component Analysis improve the results of prediction.



The third phase was to build the identified best model with the entire training data set, and to test it for accuracy and other related statistics.

#### 4.5.1 K Nearest Neighbor

The following R code is used to train KNN model using the PCA data structure.

```
mod1knn <- train(T3_Enrolled ~ ., data=trnPre, method="knn")
```

Confusion matrix of the trained model is shown below (see Table 4.2)

**Table 0.3**

Confusion Matrix of KNN using PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	148	29
	TRUE	140	681

The KNN model built with PCA data structure had an accuracy of 0.8307 and a positive prediction value of 0.8295. The following table (see Table 4.3) shows the calculated values from the confusion matrix.

**Table 0.4**

Computed statistics based on actual values and predicted values of KNN (PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>KNN</b>	0.8307	0.9592	0.5139	0.8295	0.8362

Following R code is used to train the KNN model using the original set of variables.

```
mod2knn <- train(T3_Enrolled ~ ., data=trn, method="knn")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.4)

**Table 0.5**

Confusion Matrix of KNN using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	50	46
	TRUE	238	664

The KNN model built with the original set of variables had an accuracy of 0.7154 and a positive prediction value of 0.7361. The following table (see Table 4.5) shows the calculated values from the confusion matrix.

**Table 0.6**

Computed statistics based on actual values and predicted values of KNN (original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>KNN</b>	0.7154	0.9352	0.1736	0.7361	0.5208

#### 4.5.2 Classification Tree

The following R code is used to train Classification Tree model using the compressed data set.

```
mod3tree <- train(T3_Enrolled ~ ., data=trnPre, method="rpart")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.6).

**Table 0.7**

Confusion Matrix of Classification Tree using PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	157	69
	TRUE	131	641

The Classification Tree model built with PCA data structure had an accuracy of 0.7996 and a positive prediction value of 0.8303. The following table (see Table 4.7) shows the calculated values from the confusion matrix.

**Table 0.8**

Computed statistics based on actual values and predicted values of Classification Tree  
(PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Classification Tree</b>	0.7996	0.9028	0.5451	0.8303	0.6947

The following R code is used to train Classification Tree model using the original set of variables.

```
mod3treeAll <- train(T3_Enrolled ~ ., data=trn, method="rpart")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.8)

**Table 0.9**

Confusion Matrix of Classification Tree using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	155	31
	TRUE	133	679

The Classification Tree model built with original set of variables had an accuracy of 0.8357 and a positive prediction value of 0.9362. The following table (see Table 4.9) shows the calculated values from the confusion matrix.

**Table 0.10**

Computed statistics based on actual values and predicted values of Classification Tree  
(original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Classification Tree</b>	0.8357	0.9563	0.5382	0.9362	0.8334

### 4.5.3 Random Forest

The following R code is used to train Random Forest model using the compressed data set.

```
mod4rf <- train(T3_Enrolled ~ ., data=trnPre, method="rf")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.10).

**Table 0.11**

Confusion Matrix of Random Forest using the PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	177	35
	TRUE	111	675

The Random Forest model built with PCA data structure had an accuracy of 0.8537 and a positive prediction value of 0.8588. The following table (see Table 4.11) shows the calculated values from the confusion matrix.

**Table 0.12**

Computed statistics based on actual values and predicted values of Random Forest (PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Random Forest</b>	0.8537	0.9507	0.6146	0.8588	0.8349

The following R code is used to train Random Forest model using the original set of variables.

```
mod4rfA11 <- train(T3_Enrolled ~ ., data=trn, method="rf")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.12).

**Table 0.13**

Confusion Matrix of Random Forest using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	173	39
	TRUE	115	671

The Random Forest model built with original set of variables had an accuracy of 0.8457 and a positive prediction value of 0.8537. The following table (see Table 4.13) shows the calculated values from the confusion matrix.

**Table 0.14**

Computed statistics based on actual values and predicted values of Random Forest (original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Random Forest</b>	0.8457	0.9451	0.6007	0.8537	0.816

#### 4.5.4 Binomial GLM

The following R code is used to train Binomial GLM model using the compressed data set.

```
mod5glm <- train(T3_Enrolled ~ ., data=trnPre, method="glm",
family="binomial")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.14).

**Table 0.15**

Confusion Matrix of Binomial GLM using PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	165	42
	TRUE	123	668

The Binomial GLM model built with PCA data structure had an accuracy of 0.8347 and a positive prediction value of 0.8445. The following table (see Table 4.15) shows the calculated values from the confusion matrix.

**Table 0.16**

Computed statistics based on actual values and predicted values of Binomial GLM (PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Binomial GLM</b>	0.8347	0.9408	0.5729	0.8445	0.7971

The following R code is used to train Binomial GLM model using the original set of variables.

```
mod5glmAll <- train(T3_Enrolled ~ ., data=trn, method="glm",
family="binomial")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.16).

**Table 0.17**

Confusion Matrix of Binomial GLM using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	169	46
	TRUE	119	664

The Binomial GLM model built with PCA data structure had an accuracy of 0.8347 and a positive prediction value of 0.848. The following table (see Table 4.17) shows the calculated values from the confusion matrix.

**Table 0.18**

Computed statistics based on actual values and predicted values of Binomial GLM (original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Binomial GLM</b>	0.8347	0.9352	0.5868	0.848	0.786

#### 4.5.5 Neural Network

The following R code is used to train Neural Network model using compressed data set.

```
mod6nnet <- train(T3_Enrolled ~ ., data=trnPre, method="nnet")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.18).



**Table 0.19**

Confusion Matrix of Neural Network using PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	176	37
	TRUE	112	673

The Neural Network model built with PCA data structure had an accuracy of 0.8507 and a positive prediction value of 0.8573. The following table (see Table 4.19) shows the calculated values from the confusion matrix.

**Table 0.20**

Computed statistics based on actual values and predicted values of Neural Network  
(PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Neural Network</b>	0.8507	0.9479	0.6111	0.8573	0.8263

The following R code is used to train Neural Network model using the original set of variables.

```
mod6nnetAll <- train(T3_Enrolled ~ ., data=trn, method="nnet")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.20)

**Table 0.21**

Confusion Matrix of Neural Network using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	120	50
	TRUE	168	660

The Neural Network model built with original set of variables had an accuracy of 0.7816 and a positive prediction value of 0.7971. The following table (see Table 4.21) shows the calculated values from the confusion matrix.

**Table 0.22**

Computed statistics based on actual values and predicted values of Neural Network (original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>Neural Network</b>	0.7816	0.9296	0.4167	0.7971	0.7059

#### 4.5.6 Bayesian Neural Network (BNN)

The following R code is used to train BNN model using compressed data set. One difference of training a BNN in R than other models is for BNN all the inputs should be numerical. Hence, before training the model the categorical variables were converted in to numerical variables.

```
trnBnn <- cbind(trnPre[, -(1:7)],
               dummy(trnPre$Cohort),
```

```

dummy(trnPre$School),
dummy(trnPre$College),
dummy(trnPre$Gender))
tstBnn <- cbind(tstPre[,-(1:7)],
dummy(tstPre$Cohort),
dummy(tstPre$School),
dummy(tstPre$College),
dummy(tstPre$Gender))
trnBnn$T3_Enrolled <- as.numeric(trnPre$T3_Enrolled) - 1
tstBnn$T3_Enrolled <- as.numeric(tstPre$T3_Enrolled) - 1

mod7bnn <- train(T3_Enrolled ~ ., data=trnBnn, method="brnn")

```

Confusion matrix of the trained model is shown in the following table (see Table 4.22)

**Table 0.23**

Confusion Matrix of BNN using PCA data structure

		Reference	
		FALSE	TRUE
Prediction	FALSE	175	29
	TRUE	113	681

The BNN model built with PCA data structure had an accuracy of 0.8577 and a positive prediction value of 0.8577. The following table (see Table 4.23) shows the calculated values from the confusion matrix.

**Table 0.24**

Computed statistics based on actual values and predicted values of BNN (PCA data structure)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>BNN</b>	0.8577	0.9592	0.6076	0.8577	0.8578

The following R code is used to train BNN model using the original set of variables.

```
mod7bnn2 <- train(T3_Enrolled ~ ., data=trnBnn2, method="brnn")
```

Confusion matrix of the trained model is shown in the following table (see Table 4.24).

**Table 0.25**

Confusion Matrix of BNN using original set of variables

		Reference	
		FALSE	TRUE
Prediction	FALSE	171	30
	TRUE	117	680

The BNN model built with original set of variables had an accuracy of 0.8507 and a positive prediction value of 0.8555. The following table (see Table 4.25) shows the calculated values from the confusion matrix.

**Table 0.26**

Computed statistics based on actual values and predicted values of BNN (original set of variables)

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>BNN</b>	0.8527	0.9577	0.5938	0.8507	0.8532

## Chapter 5: Comparing Models and Experimental Results

The results obtained during the model building phase were used in this stage to determine two things. First, to determine if the hypotheses made about using Principal Component Analysis will result in better predictive models, and second, to select the best model by looking at the statistics of the trained models.

### 5.1 Measures Used to Compare Models and Calculated Statistics

A few measures were used to compare the models. Namely, accuracy, sensitivity, specificity, positive predictive value, and negative predictive value.

**Table 0.1**

Calculated statistics of predictions of models using principal component analysis structure

	Accuracy	Sensitivity	Specificity	Pos.pred.val	Neg.pred.val
<b>KNN</b>	0.8307	0.9592	0.5139	0.8295	0.8362
<b>Classification Tree</b>	0.7996	0.9028	0.5451	0.8303	0.6947
<b>Random Forest</b>	0.8537	0.9507	0.6146	0.8588	0.8349
<b>Binomial GLM</b>	0.8347	0.9408	0.5729	0.8445	0.7971
<b>Neural Network</b>	0.8507	0.9479	0.6111	0.8573	0.8263
<b>BNN</b>	0.8577	0.9592	0.6076	0.8577	0.8578

**Table 0.2**

Calculated statistics of predictions of models using original set of variables

	<b>Accuracy</b>	<b>Sensitivity</b>	<b>Specificity</b>	<b>Pos.pred.val</b>	<b>Neg.pred.val</b>
<b>KNN</b>	0.7154	0.9352	0.1736	0.7361	0.5208
<b>Classification Tree</b>	0.8357	0.9563	0.5382	0.8362	0.8333
<b>Random Forest</b>	0.8457	0.9451	0.6007	0.8537	0.816
<b>Binomial GLM</b>	0.8347	0.9352	0.5868	0.848	0.786
<b>Neural Network</b>	0.7816	0.9296	0.4167	0.7971	0.7059
<b>BNN</b>	0.8577	0.9592	0.6076	0.8577	0.8578

**Table 0.3**

Performance deference when using PCA structure over original set of variables

	<b>Accuracy</b>	<b>Sensitivity</b>	<b>Specificity</b>	<b>Pos.pred.val</b>	<b>Neg.pred.val</b>
<b>KNN</b>	0.1153	0.024	0.3403	0.0934	0.3154
<b>Classification Tree</b>	-0.0361	-0.0535	0.0069	-0.0059	-0.1386
<b>Random Forest</b>	0.008	0.0056	0.0139	0.0051	0.0189
<b>Binomial GLM</b>	0	0.0056	-0.0139	-0.0035	0.0111
<b>Neural Network</b>	0.0691	0.0183	0.1944	0.0602	0.1204
<b>BNN</b>	0.005	0.0015	0.0138	0.007	0.0046

KNN algorithm had an accuracy improvement of 0.1153 when the PCA structure was used to train the model. Particularly, the specificity of KNN when using PCA was improved substantially. Improvement of specificity was 0.3403. That is a 34% increase when PCA was used. Sensitivity was also improved by 0.024.

Classification Tree did not show an overall performance improvement when the PCA structure was used. However, specificity was 0.0069 higher when PCA was used whereas sensitivity was 0.0535 higher when the original set of variables was used. Overall there was a

0.0361 or 3.61% improvement of accuracy when the original set of variables was used to train this model.

Random Forest indicated higher performance in all measures when the PCA structure was used for training the model. It had a 0.0056 higher sensitivity value and a 0.0139 improvement of the specificity value. The accuracy improvement was at 0.008 when the PCA structure was used. Even though it did not show drastic improvements when the PCA structure was used, considering that all measures had higher values than using the original set of variables, it is safe to determine that Random Forest had higher performance when the PCA structure was used.

Binomial GLM did not have any difference in accuracy value in either cases. Even though sensitivity was 0.0056 higher when PCA was used, specificity was reduced by 0.0139. Overall, this model did not show any significant difference when the PCA structure was used over the original set of variables.

Similar to Random Forest, Neural Networks showed improvement in all measures. When the PCA structure was used, the improvements in accuracy, sensitivity and specificity were 0.6910, 0.0183, and 0.1944 respectively. When the statistics are considered Neural Networks performed better when the PCA structure was used to train the model.

The BNN model also behaved in a similar way to the Neural Network model. It had higher performance when the PCA structure was used. Sensitivity and specificity improved by 0.0015 and 0.0138 respectively. Accuracy was improved by 0.005. Even though the improvements are less significant, all the measures had slightly better values when the PCA structure was used to train BNN.



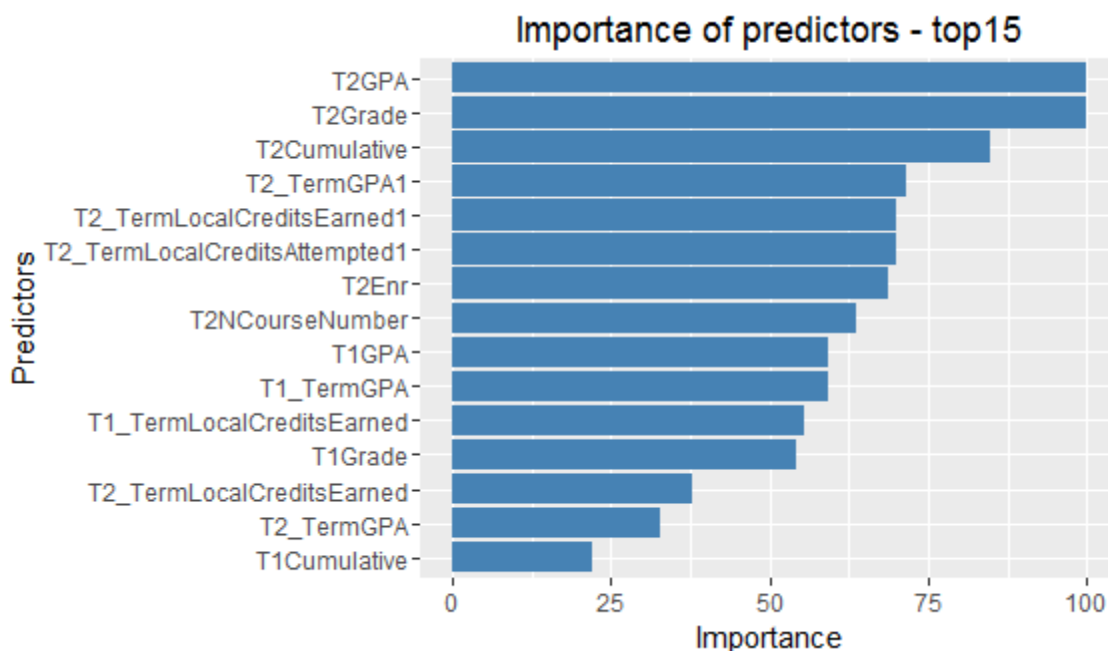
## 5.2 Top Predictors and Retention Probabilities

After comparing the models, the two models with the best performance were selected out of the compared models. These two models were retrained using data of 11640 students, which is 75% of data in the entire population. Then, by using the trained models, the most significant factors contributed in predicting the student retention were identified.

The importance for the model was calculated using the R function ‘varImp’ in package caret. This function uses the separate variables of importance that each predictor will have in most classification models. These importance variables have values ranging from 0-100 based on how closely these variables are tied to the model performance. For example, in the Random Forest model the R package description explains that the calculation of the importance can be done in two ways: using Partial Least Squares (PLS) and Recursive Partitioning (RP). In the case of PLS, the importance measure is based on weighted sums of the absolute regression coefficients. The weights are a function of the reduction of the sums of squares across the number of PLS components and are computed separately for each outcome. Therefore, the contribution of the coefficients are weighted proportionally to the reduction in the sums of squares. In the case of Recursive Partitioning, the reduction in the loss function (e.g. mean squared error) attributed to each variable at each split is tabulated and the sum is returned. Also, since there may be candidate variables that are important but are not used in a split, the top competing variables are also tabulated at each split [31].

### 5.2.1 Top 15 Predictors for BNN Model

For BNN, academic factors of the student's sophomore year were impacting the prediction than the other factors that were used in training the model. Figure 5.1 shows the top 15 predictors for BNN in the order of importance.



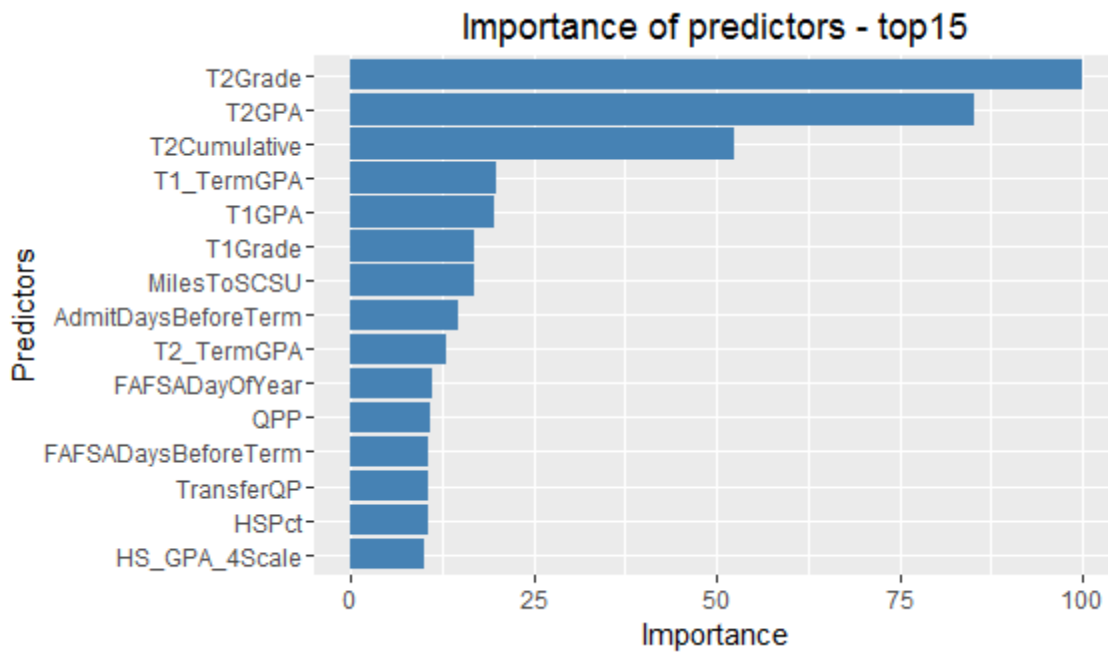
**Figure 0.1**

Top 15 predictors for BNN

### 5.2.2 Top 15 Predictors for Random Forest Model

For Random Forest, a combination of academic factors of the student's sophomore year and high school, as well as some of the factors which were introduced to measure readiness for the college education and financial stability were included in the top 15 predictors. Even though there were a good blend of factors as the top 15 predictors for Random Forest, academic factors of student's second semester of the sophomore year had significantly high impact than the other

factors in top 15. Figure 5.2 shows the top 15 predictors for Random Forest in the order of importance.



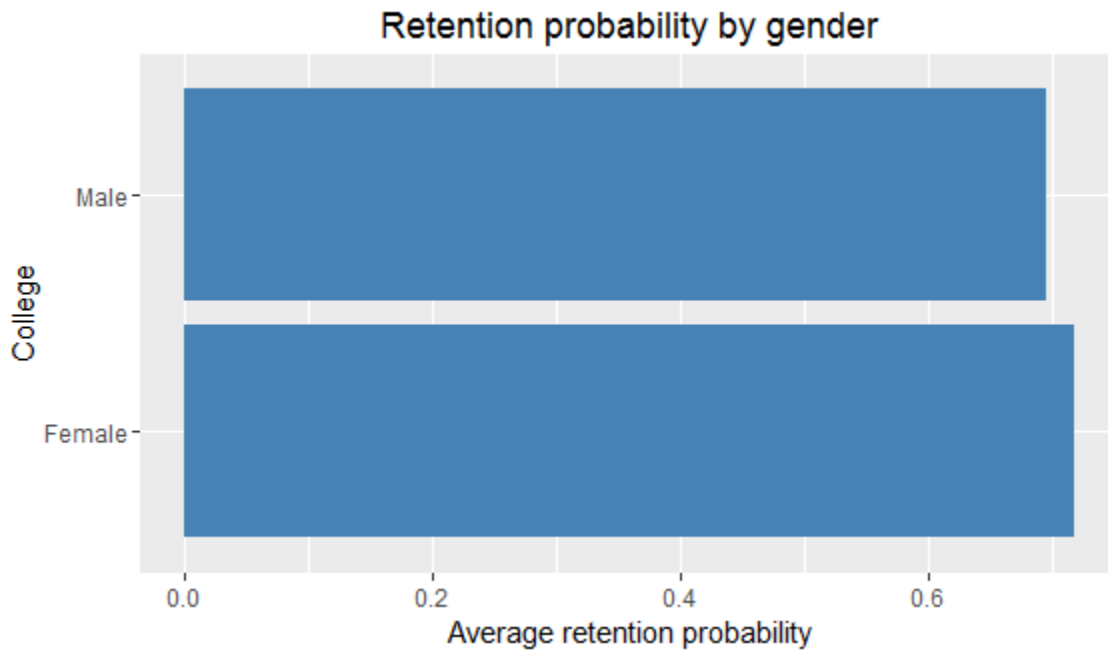
**Figure 0.2**

Top 15 predictors for Random Forest

### 5.2.3 Retention Probabilities by Gender

Even though the statistics suggest that females has a higher probability of retaining after sophomore year, there was no substantial difference in the retention probability by gender.

Figure 5.3 shows the retention probability by gender.

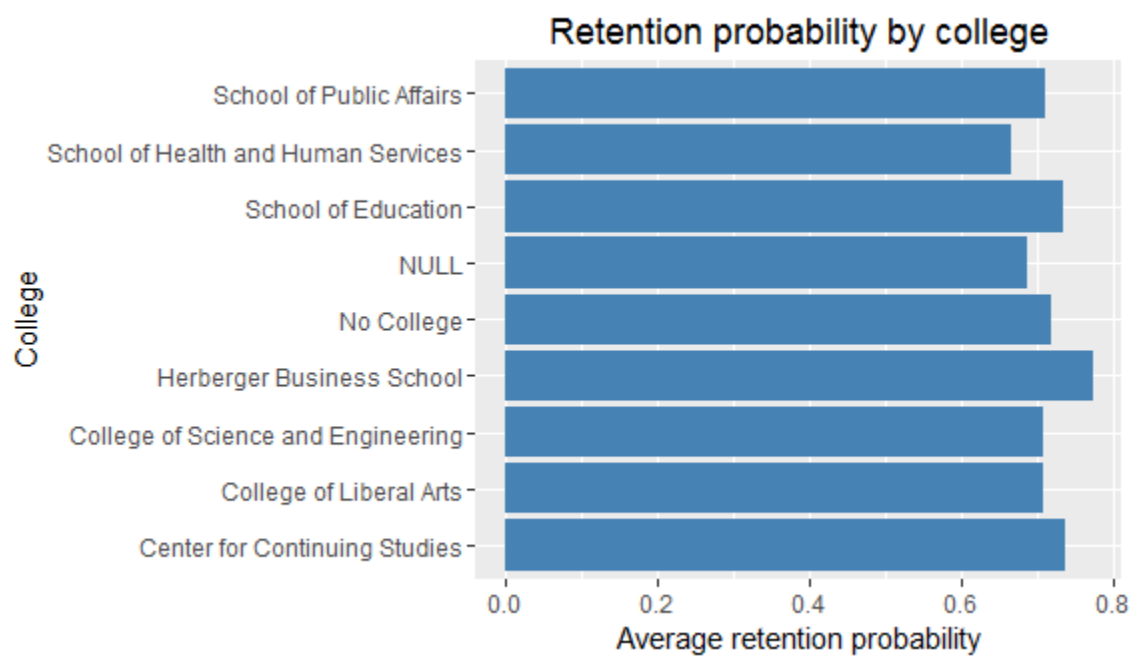


**Figure 0.3**

Retention probability by gender

#### 5.2.4 Retention Probability by College

Majority of the colleges had retention probabilities over 70% while Herberger Business School had the highest probability which was close to 78%. Figure 5.4 shows the average retention probabilities by college. The students who were indecisive of the college at the time of admission were categorized as “No College” and the students with missing values for college details were categorized as “NULL”.



**Figure 0.4**

Retention probability by college

## Chapter 6: Discussion

By comparing the statistics of models trained with PCA structure with the same statistics of the models trained with the original set of variables, it can be concluded that there are no clear cut results to confirm that the hypothesis that was made about the principal component analysis is always true. That is that PCA yields better results than the results obtained by using the original set of variables. However, when the accuracy value is compared it is evident that in most cases, models built using the PCA structure yielded better results. But, in some cases, the models that were built using the original set of variables were better. Namely, KNN had the most significant difference using the PCA structure, while Random Forest, Neural Network, and BNN had a slight advantage when the PCA structure was used. The Classification Tree model had a considerable advantage when the original set of variables was used over the PCA trained model. The Binomial GLM performed equally in either cases when considering the accuracy of the trained model.

Apart from the Classification Tree model, all the other cases had improvements of sensitivity when the PCA structure was used in training models. Majority of the models showed improvements in specificity when the PCA structure was used. Namely, KNN, Classification Tree, Random Forest, Neural Networks and BNN had better specificity values with PCA while Binomial GLM was the only model which had a drop in the same category.

Out of the six models that were used, apart from the Classification Tree and the Binomial GLM, all the other models showed improvements in all areas when the PCA structure was used. The overall average of accuracy of all models was approximately 84%. KNN showed the lowest

accuracy at 83.07%, while BNN showed the highest accuracy at 85.87% when the PCA structure was used.

An analysis of the results reveal that Random Forest and BNN had the highest accuracy values and that they were very close to each other. Both models had the highest accuracy when the PCA structure was used. It can be determined that either of these models can be used with PCA data structure to predict retention at St Cloud State University with a very high accuracy. Furthermore, it can be concluded that the use of data mining and an algorithmic approach to predict retention can yield results with high accuracy.

## References

- [1] D. W. Knight, L. E. Carlson, and J. F. Sullivan, "Staying in engineering: Impact of a hands-on, team-based, first-year projects course on student retention," in *American Society for Engineering Education Annual Conference & Exposition. Session 3553*, 2003.
- [2] J. Richardson and J. Dantzler, "Effect of a freshman engineering program on retention and academic performance," in *Frontiers in Education Conference Proceedings, ASEE/IEEE*, 2002, pp. S2C-16-S2C-22.
- [3] Education—US News & World Report. "Freshman Retention Rate". [Online}/ Available: <http://colleges.usnews.rankingsandreviews.com/best-colleges/rankings/national-universities/freshmen-least-most-likely-return>.
- [4] V. Tinto, "Research and practice of student retention: What next?", *Journal of College Student Retention: Research, Theory and Practice*, vol. 8, no. 1, pp. 1-19, 2006.
- [5] R. Alkhasawneh and R. H. Hargraves, "Developing a hybrid model to predict student first year retention in STEM disciplines using machine learning techniques," *Journal of STEM Education: Innovations and Research*, vol. 15, no. 3, pp. 35-42, 2014.
- [6] S. Herzog, "Estimating student retention and degree-completion time: decision trees and neural networks vis-à-vis regression," *New Directions for Institutional Research*, vol. 131, pp. 17-33, 2006.
- [7] V. Tinto, "Dropout from higher education: A theoretical synthesis of recent research," *Review of Educational Research*, no. 1, p. 89, 1975.
- [8] A. W. Astin, *Assessment for Excellence: The Philosophy and Practice of Assessment and Evaluation in Higher Education*. Phoenix, AZ: Oryx Press, 1993.
- [9] M. Bogard, T. Helbig, G. Huff, and C. James, "A comparison of empirical models for predicting student retention," *White paper*. Office of Institutional Research, Western Kentucky University. 2011.
- [10] A. Cabrera, A. Nora, and N. Castaneda, "College persistence: Structural equations modeling test of an integrated model of student retention," *The Journal of Higher Education*, vol. 64, no. 2, pp. 123-139, 1993.
- [11] S. K. Yadav, B. Bharadwaj, and S. Pal, "Mining education data to predict student's retention: A comparative study," arXiv preprint arXiv:1203.2987, 2012.



- [12] D. Kabakchieva, "Student performance prediction by using data mining classification algorithms," *International Journal of Computer Science and Management Research*, vol. 1, no. 4, pp. 686-690, 2012.
- [13] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21-27, 1967.
- [14] Hasti, Tibshirani, and Friedman, *Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2<sup>nd</sup> ed. Springer-Verlag, 2009.
- [15] J. Arifovic and R. Gencay, "Using genetic algorithms to select architecture of a feedforward artificial neural network," *Physica A: Statistical Mechanics and Its Applications*, vol. 289, no. 3, pp. 574-594, 2001.
- [16] R. Alkhasawneh and R. Hobson, "Modeling student retention in science and engineering disciplines using neural networks," in *Global Engineering Education Conference (EDUCON), 2011 IEEE*, 2011, pp. 660-663.
- [17] J. Yang, J. Ma, M. Berryman, and P. Perez, "A structure optimization algorithm of neural networks for large-scale data sets," in *2014 International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2014, pp. 956-961.
- [18] J. L. Brown, "Developing a freshman orientation survey to improve student retention within a college," *College Student Journal*, vol. 46, no. 4, pp. 834-851, 2012.
- [19] CollegeBoard Advocacy, "How colleges organize themselves to increase student persistence: four-year institutions," [Online], Available: <https://professionals.collegeboard.com/profdownload/college-retention.pdf>, [April 2009].
- [20] Student Financial Aid Services, Inc., "What is FAFSA?," {Online}, Available: <http://www.fafsa.com/understanding-fafsa/what-is-fafsa>.
- [21] J. Luan, and A. Serban, *Knowledge Management: Building a Competitive Advantage in Higher Education*. New Directions for Institutional Research, 113. San Francisco: JosseyBass, 2002.
- [22] M. Skahill, "The role of social support network in college persistence among freshmen students," *Journal of College Student Retention*, vol. 4, no. 1, pp. 39-52, 2002.
- [23] U.S. Department of Education, "Federal Pell Grant program," [Online], Available: <http://www2.ed.gov/programs/fpg/index.html>.

- [24] J.-H. Kim, “Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap,” *Computational Statistics and Data Analysis*, vol. 53, no. 11, pp. 3735–3745. doi:10.1016/j.csda.2009.04.009
- [25] H. Abdi, and L. J. Williams, “Principal component analysis,” *Wiley Interdisciplinary Reviews: Computational Statistics*, 2010.
- [26] A. Gelman, J. B. Carlin, S. Hal, and D. B. Rubin, *Bayesian Data Analysis*. Washington DC: Chapman and Hall/CRC, 2003.
- [27] D. Kahneman, P. Slovic, and A. Tversky, *Judgement Under Uncertainty: Heuristics and Biases*. Cambridge. University of Cambridge Press, 2001.
- [28] R. Peck, and J. Devore, *Statistics: The Exploration and Analysis of Data*. Boston, MA: Richard Straton, 2012.
- [29] S. Lohr, *Sampling Design and Analysis*. Boston: Arizona State University, 2010.
- [30] T. Lung Lai, and H. Xing, *Statistical Models and Methods for Financial Markets*. Stanford: Springer, 2008.
- [31] M. Kuhn, “Calculation of variable importance for regression and classification models,” inside-r.org, caret package, version 6.0-70.

### Additional Reading

A. C. Noonan, M. Lundy, A. R. Smith Jr., and B. P. Livingston, "A successful model for improving student retention in physical therapist education programs: a case report," *Journal of Physical Therapy Education*, vol. 26, no. 2, pp. 74-80, 2012.

D. Johnson, T. Wasserman, N. Yildirim, and B. Yonai, "Examining the effects of stress and campus climate on the persistence of students of color and white students: An application of bean and eaton's psychological model of retention," *Research in Higher Education*, vol. 55, no. 1, pp. 75-100, 2014. doi:10.1007/s11162-013-9304-9

E. N. Shelton, "A model of nursing student retention," *International Journal of Nursing Education Scholarship*, vol. 9, no. 1, pp. 1-16, 2012. doi:10.1515/1548-923X.2334

G. Hartsuff, G. "The science of retention – freshman retention," [Online], Available: [http://www.americancollegiaterowing.com/Files/Coaching\\_Manual /THE%20SCIENCE %20OF%20RETENTION-Freshman.pdf](http://www.americancollegiaterowing.com/Files/Coaching_Manual /THE%20SCIENCE %20OF%20RETENTION-Freshman.pdf).

J. A. Morrow and M. E. Ackermann, "Intention to persist and retention of first-year students: the importance of motivation and sense of belonging," *College Student Journal*, vol. 46, no. 3, pp. 483-491, 2012.

K. M. Soria and M. J. Stebleton, "First-generation students' academic engagement and retention," *Teaching In Higher Education*, vol. 17, no. 6, pp. 673-685, 2012. doi:10.1080/13562517.2012.666735

L. A. Schreiner, "Linking student satisfaction and retention, [Online], Available: <http://www.uwstout.edu/admin/provost/upload/LinkingStudentSatis0809.pdf> [2009].

L. D. Singell and G. R. Waddell, "Modeling retention at a large public university: Can at-risk students be identified early enough to treat?", *Research in Higher Education*, vol. 51, no. 6, pp. 546-572, 2010. doi:10.1007/s11162-010-9170-7




M. McElveen and A. Rossow, "Relationship of intramural participation to GPA and retention in first-time-in-college students," *Recreational Sports Journal*, vol. 38, no. 1, pp. 50-54, 2014.

National Student Clearinghouse Research Center, "Snapshot report persistence-retention," [Online], Available: <http://nscresearchcenter.org/wp-content/uploads/SnapshotReport14-PersistenceRetention-.pdf> [spring 2014].

S. Kampf, and E. J. Teske, "Collegiate recreation participation and retention," *Recreational Sports Journal*, vol. 37, no. 2, pp. 85-96, 2013.

## Appendix

### Institutional Review Board (IRB) approval for the use of data involving human subjects.

	<b>Institutional Review Board (IRB)</b>	
<b>OFFICE OF RESEARCH AND SPONSORED PROGRAMS</b> ST. CLOUD STATE UNIVERSITY.	<b>Administrative Services 210</b> Website: <a href="http://stcloudstate.edu/osp">stcloudstate.edu/osp</a> Email: <a href="mailto:osp@stcloudstate.edu">osp@stcloudstate.edu</a> Phone: 320-308-4932	
<b>Name:</b> Hasith Disanayake <b>Address:</b> 1815 15th Ave. SE St. Cloud, MN 56304 USA <b>Email:</b> diha1201@stcloudstate.edu		<b>IRB PROTOCOL DETERMINATION: Exempt Review</b>
<b>Project Title:</b> Predictive Modeling: Predicting Student Retention at St. Cloud State University <b>Advisor:</b> Omar Al-Azzam		
The Institutional Review Board has reviewed your protocol to conduct research involving human subjects. Your project has been: <b>APPROVED</b>		
Please note the following important information concerning IRB projects:		
- The principal investigator assumes the responsibilities for the protection of participants in this project. Any adverse events must be reported to the IRB as soon as possible (ex. research related injuries, harmful outcomes, significant withdrawal of subject population, etc.).		
- For expedited or full board review, the principal investigator must submit a Continuing Review/Final Report form in advance of the expiration date indicated on this letter to report conclusion of the research or request an extension.		
-Exempt review only requires the submission of a Continuing Review/Final Report form in advance of the expiration date indicated in this letter if an extension of time is needed.		
- Approved consent forms display the official IRB stamp which documents approval and expiration dates. If a renewal is requested and approved, new consent forms will be officially stamped and reflect the new approval and expiration dates.		
- The principal investigator must seek approval for any changes to the study (ex. research design, consent process, survey/interview instruments, funding source, etc.). The IRB reserves the right to review the research at any time.		
Good luck on your research. If we can be of further assistance, please contact the Office of Research and Sponsored Programs at 320-308-4932 or email <a href="mailto:lidonnay@stcloudstate.edu">lidonnay@stcloudstate.edu</a> . Use the SCSU IRB number listed on any forms submitted which relate to this project, or on any correspondence with the IRB.		
<b>Institutional Review Board:</b>  Linda Donnay IRB Administrator Office of Research and Sponsored Programs	<b>St. Cloud State University:</b>  Marilyn Hart Interim Associate Provost for Research Dean of Graduate Studies	
OFFICE USE ONLY		
SCSU IRB# 1527 - 1901 1st Year Approval Date: 12/17/2015 1st Year Expiration Date: 12/16/2018	Type: Exempt Review 2nd Year Approval Date: 2nd Year Expiration Date:	Today's Date: 12/18/2015 3rd Year Approval Date: 3rd Year Expiration Date:

### Description of the Variables Used

Variable Name	Description
<b>StudentId</b>	Student ID
<b>Cohort</b>	Cohort of the student (freshman, transfer student etc.)
<b>CohortTerm</b>	Semester that the student got admission to SCSU (Summer, fall, or spring)
<b>CohortTermEnrolled</b>	A flag to indicate if the student enrolled for the semester they got admission to
<b>AdmitDaysBeforeTerm</b>	Number of days after the student got admission and before the semester start date
<b>AdmitCalYear</b>	Calendar year which the student got admission to SCSU
<b>AdmitTermBeginDate</b>	Start date of the admitted semester
<b>SOC</b>	A flag to indicate a student of color
<b>ACE</b>	A flag to indicate if a student is provisionally admitted
<b>ACT_Composite</b>	ACT score of the student
<b>ACT1</b>	Indicator variable for ACT_Composite variable
<b>HS_GPA_4Scale</b>	High school GPA in a scale of 4
<b>HSGPA1</b>	Indicator variable for HS_GPA_4Scale variable
<b>HSPct</b>	High school percentage
<b>HSPct1</b>	Indicator variable for HSPct variable
<b>QPP</b>	Quality points predicted of the student
<b>EnglTotal</b>	Number of years the student learnt English
<b>EnglTotal1</b>	Indicator variable for EnglTotal variable
<b>ClosestToSCSU</b>	Student is within a predefined range from SCSU
<b>ClosestToSCSU1</b>	Indicator variable for ClosestToSCSU variable
<b>ScholarGrid</b>	A flag to indicate if the student got a scholarship or not
<b>MilesToSCSU</b>	The distance in miles from student's home to SCSU
<b>MilesToSCSU1</b>	Indicator variable for MilesToSCSU variable
<b>FirstGen</b>	A flag to indicate if the student is a first generation student
<b>PellElig</b>	A flag to indicate if the student is Pell grant eligible
<b>HonorsFlag</b>	A flag to indicate if the student is an honors student
<b>IntlFlag</b>	A flag to indicate if the student is an international student

<b>Top10Flag</b>	A flag to indicate if the student is within the top 10 of the class
<b>Top25Flag</b>	A flag to indicate if the student is within the top 25 of the class
<b>MidRangeFlag</b>	A flag to indicate if the student is around the mid-range of the class
<b>HousingAppDaysBeforeTerm</b>	If student applied for housing, how many before the start date did s/he apply for housing
<b>HousingAppDaysBeforeTerm1</b>	Indicator variable for HousingAppDaysBeforeTerm variable
<b>HousingFlag</b>	A flag to indicate if the student applied for housing
<b>FAFSADaysBeforeTerm</b>	How many days before the term was the FAFSA form submitted
<b>FAFSADayOfYear</b>	Number of days from January 1st to the date
<b>FAFSASFlag</b>	A flag to indicate if the student applied for FAFSA
<b>FAFSA_Nbr</b>	Order of interest on FAFSA form, the number was SCSU was ranked at
<b>FAFSASFirstPosFlag</b>	A flag to indicate if SCSU is the first on student's FAFSA list
<b>TotalCRHR_TRSF</b>	Number of transferred credits
<b>TransferGPA</b>	GPA of a transfer student
<b>TransferGPA1</b>	Indicator variable for TransferGPA variable
<b>TransferQP</b>	Quality points of a transfer student
<b>TransferQP1</b>	Indicator variable for TransferQP variable
<b>Major</b>	Intendent major at SCSU
<b>Program</b>	Name of program the student is in
<b>School</b>	Name of school the student is in
<b>College</b>	Name of college within the school the student is in
<b>Department</b>	Name of department within the college the student is in
<b>NEF</b>	A flag to indicate if the student is a freshman
<b>NET</b>	A flag which indicates if the student is a transfer student
<b>HS_MnSCURegion</b>	High school MnSCU region out of the 13 regions
<b>Age</b>	Age of the student
<b>Gender</b>	Gender of the student
<b>T1_TermGPA</b>	GPA of the student in term one
<b>T2_TermGPA</b>	GPA of the student in term two
<b>T2_TermGPA1</b>	Indicator variable for T2_TermGPA variable

<b>T1_TermLocalCreditsAttempted</b>	Number of credits attempted during term one
<b>T2_TermLocalCreditsAttempted</b>	Number of credits attempted during term two
<b>T2_TermLocalCreditsAttempted1</b>	Indicator variable for T2_TermLocalCreditsAttempted variable
<b>T1_TermLocalCreditsEarned</b>	Number of credits earned during term one
<b>T2_TermLocalCreditsEarned</b>	Number of credits earned during term two
<b>T2_TermLocalCreditsEarned1</b>	Indicator variable for T2_TermLocalCreditsEarned variable
<b>T1_Enrolled</b>	A flag to indicate if the student is enrolled for the 1st term
<b>T2_Enrolled</b>	A flag to indicate if the student is enrolled for the 2nd term
<b>T3_Enrolled</b>	A flag to indicate if the student is enrolled for the 3rd term
<b>T1Cumulative</b>	The cumulative credits of the student at the end of term 1
<b>T2Cumulative</b>	The cumulative credits of the student at the end of term 2
<b>T1Grade</b>	Mean of the number grades of the student for semester 1
<b>T2Grade</b>	Mean of the number grades of the student for semester 2

## R Code for Training Models

### k-Nearest Neighbor with PCA structure

```
# knn with compressed data

t0 <- Sys.time()
if(!file.exists("./result/mod1knn.rds")){
  mod1knn <- train(T3_Enrolled ~ ., data=trnPre, method="knn")
  saveRDS(mod1knn, "./result/mod1knn.rds")
} else {
  mod1knn <- readRDS("./result/mod1knn.Rds")
}
res1knn <- confusionMatrix(predict(mod1knn, newdata=tstPre, type="raw"), tst$
T3_Enrolled)
print(Sys.time()-t0)
```

### k-Nearest Neighbor with original set of variables

```
# knn with all data

t0 <- Sys.time()
if(!file.exists("./result/mod2knn.rds")){
  mod2knn <- train(T3_Enrolled ~ ., data=trn, method="knn")
  saveRDS(mod2knn, "./result/mod2knn.rds")
} else {
  mod2knn <- readRDS("./result/mod2knn.rds")
}
res2knn <- confusionMatrix(predict(mod2knn, newdata=tst, type="raw"), tst$T3_Enrolled)

print(Sys.time()-t0)
```

### Classification Tree with PCA structure

```
# use compressed data - tree

t0 <- Sys.time()
if(!file.exists("./result/mod3tree.rds")){
  mod3tree <- train(T3_Enrolled ~ ., data=trnPre, method="rpart")
  saveRDS(mod3tree, "./result/mod3tree.rds")
} else {
  mod3tree <- readRDS("./result/mod3tree.rds")
}
res3tree <- confusionMatrix(predict(mod3tree, newdata=tstPre, type="raw"), ts
t$T3_Enrolled)
print(Sys.time()-t0)
```

### Classification Tree with original set of variables

```
# use ALL data - tree

t0 <- Sys.time()
if(!file.exists("./result/mod3treeAll.rds")){
  mod3treeAll <- train(T3_Enrolled ~ ., data=trn, method="rpart")
  saveRDS(mod3treeAll, "./result/mod3treeAll.rds")
} else {
```



```

    mod3treeAll <- readRDS("./result/mod3treeAll.rds")
  }
  res3treeAll <- confusionMatrix(predict(mod3treeAll, newdata=tst, type="raw"),
    tst$T3_Enrolled)
  print(Sys.time()-t0)

```

### Random Forest with PCA structure

```

t0 <- Sys.time()
if(!file.exists("./result/mod4rf.rds")){
  mod4rf <- train(T3_Enrolled ~ ., data=trnPre, method="rf")
  saveRDS(mod4rf, "./result/mod4rf.rds")
} else {
  mod4rf <- readRDS("./result/mod4rf.rds")
}
res4rf <- confusionMatrix(predict(mod4rf, newdata=tstPre, type="raw"), tst$T3_Enrolled)
print(Sys.time()-t0)

```

### Random Forest with original set of variables

```

# use ALL data - random forest

t0 <- Sys.time()
if(!file.exists("./result/mod4rfAll.rds")){
  mod4rfAll <- train(T3_Enrolled ~ ., data=trn, method="rf")
  saveRDS(mod4rfAll, "./result/mod4rfAll.rds")
} else {
  mod4rfAll <- readRDS("./result/mod4rfAll.rds")
}
res4rfAll <- confusionMatrix(predict(mod4rfAll, newdata=tst, type="raw"), tst$T3_Enrolled)
print(Sys.time()-t0)

```

### Binomial GLM with PCA structure

```

# GLM with compressed data

t0 <- Sys.time()
if(!file.exists("./result/mod5glm.rds")){
  mod5glm <- train(T3_Enrolled ~ ., data=trnPre, method="glm", family="bino

```

```

mial")
  saveRDS(mod5glm, "./result/mod5glm.rds")
} else {
  mod5glm <- readRDS("./result/mod5glm.rds")
}
res5glm <- confusionMatrix(predict(mod5glm, newdata=tstPre, type="raw"), tst$
T3_Enrolled)
print(Sys.time()-t0)

```

## Binomial GLM with original set of variables

```

# GLM with all data

t0 <- Sys.time()
if(!file.exists("./result/mod5glmAll.rds")){
  mod5glmAll <- train(T3_Enrolled ~ ., data=trn, method="glm", family="bino
mial")
  saveRDS(mod5glmAll, "./result/mod5glmAll.rds")
} else {
  mod5glmAll <- readRDS("./result/mod5glmAll.rds")
}
res5glmAll <- confusionMatrix(predict(mod5glmAll, newdata=tst, type="raw"), t
st$T3_Enrolled)
print(Sys.time()-t0)

```

## Neural Networks with PCA structure

```

# nnet with compressed data

t0 <- Sys.time()
if(!file.exists("./result/mod6nnet.rds")){
  mod6nnet <- train(T3_Enrolled ~ ., data=trnPre, method="nnet")
  saveRDS(mod6nnet, "./result/mod6nnet.rds")
} else {
  mod6nnet <- readRDS("./result/mod6nnet.rds")
}
res6nnet <- confusionMatrix(predict(mod6nnet, newdata=tstPre, type="raw"), ts
t$T3_Enrolled)
print(Sys.time()-t0)

```

## Neural Networks with original set of variables

```
# nnet with all data
t0 <- Sys.time()
if(!file.exists("./result/mod6nnetAll.rds")){
  mod6nnetAll <- train(T3_Enrolled ~ ., data=trn, method="nnet")
  saveRDS(mod6nnetAll, "./result/mod6nnetAll.rds")
} else {
  mod6nnetAll <- readRDS("./result/mod6nnetAll.rds")
}
res6nnetAll <- confusionMatrix(predict(mod6nnetAll, newdata=tst, type="raw"),
tst$T3_Enrolled)
print(Sys.time()-t0)
```

## Bayesian Networks with PCA structure

```
# BNN structure

### Dummy vars

trnBnn <- cbind(trnPre[,-(1:7)],
               dummy(trnPre$Cohort),
               dummy(trnPre$School),
               dummy(trnPre$College),
               dummy(trnPre$Gender))
tstBnn <- cbind(tstPre[,-(1:7)],
               dummy(tstPre$Cohort),
               dummy(tstPre$School),
               dummy(tstPre$College),
               dummy(tstPre$Gender))

trnBnn$T3_Enrolled <- as.numeric(trnPre$T3_Enrolled) - 1
tstBnn$T3_Enrolled <- as.numeric(tstPre$T3_Enrolled) - 1

names(trnBnn) <- make.names(names(trnBnn))
names(trnBnn)[2:ncol(trnBnn)] <- paste0(names(trnBnn)[2:ncol(trnBnn)], 2:ncol
(trnBnn))

t0 <- Sys.time()
if(!file.exists("./result/mod7bnn.rds")){
  mod7bnn <- train(T3_Enrolled ~ ., data=trnBnn, method="brnn")
  saveRDS(mod7bnn, "./result/mod7bnn.rds")
}
```

```

} else {
  mod7bnn <- readRDS("./result/mod7bnn.rds")
}

names(tstBnn) <- names(trnBnn)
ps <- predict(mod7bnn, newdata=tstBnn)
psc <- ps > .5
res7bnn <- confusionMatrix(psc, tstBnn$T3_Enrolled==1)
print(Sys.time()-t0)

```

### Bayesian Network with original set of variables

```

trnBnn2 <- cbind(T3_Enrolled = trn[,64],
                trn[, -c(1,32:36,41,64)],
                dummy(trnPre$Cohort),
                dummy(trnPre$School),
                dummy(trnPre$College),
                dummy(trnPre$Gender))
tstBnn2 <- cbind(T3_Enrolled = tst[,64],
                tst[, -c(1,32:36,41,64)],
                dummy(tstPre$Cohort),
                dummy(tstPre$School),
                dummy(tstPre$College),
                dummy(tstPre$Gender))

trnBnn2$T3_Enrolled <- as.numeric(trnPre$T3_Enrolled) - 1
tstBnn2$T3_Enrolled <- as.numeric(tstPre$T3_Enrolled) - 1

names(trnBnn2) <- make.names(names(trnBnn2))
names(trnBnn2)[2:ncol(trnBnn2)] <- paste0(names(trnBnn2)[2:ncol(trnBnn2)], 2:
ncol(trnBnn2))

#### Run the model

t0 <- Sys.time()
if(!file.exists("./result/mod7bnn2.rds")){
  mod7bnn2 <- train(T3_Enrolled ~ ., data=trnBnn2, method="brnn")
  saveRDS(mod7bnn2, "./result/mod7bnn2.rds")
} else {
  mod7bnn2 <- readRDS("./result/mod7bnn2.rds")
}

names(tstBnn2) <- names(trnBnn2)

```

```
ps <- predict(mod7bnn2, newdata=tstBnn2)
psc <- ps > .5
res7bnn2 <- confusionMatrix(psc, tstBnn2$T3_Enrolled==1)
print(Sys.time()-t0)
```