Students Dropout Prediction Model in Higher Education Institutions Using Machine Learning Algorithms

Luke Philip Ogweno, Hong Shi, and Divya Sharma

Towards a Students' Dropout Prediction Model in Higher Education Institutions Using Machine Learning Algorithms

Abstract

Student dropout is considered the most complex and significnt issue in the education system. This problem causes economic, social, academic, political, and financial damage to the main agents of education i.e., from the student to the governmental and promotional agencies for effective and efficient strategies to minimize the indexex of school dropout, so that the measures adopted can have a positive effect on the problem. To successfully reduce student dropout, it is imperative to understand what the underlying determinants of dropout rates are and which students are at risk of dropping out. Therefore, early identification of potential dropout students is very imperative. Here we analyzed educational data and generated predictive models for student dropout using neural networks (NN), decision trees (DT), naive Bayes (NB), support vector machine (SVM) and random forest (RF) algorithms to identify student characteristics which distinguish potential dropouts from graduates.

1 Introduction

Academic success in higher education is vital for jobs, social justice, and economic growth. Dropout represents the most problematic issue that higher education institutions must address to improve their success. There is no universally accepted definition of dropout. The proportion of students who dropout varies between different studies depending on how dropout is defined, the data source, and the calculation methods (Behr et al. 2020)

Frequently, dropout is analyzed in the research literature based on the timing of the dropout (early vs. late) (Kehm, Larsen, and Sommersel 2019). Due to differences in reporting, it is not possible to compare dropout rates across institutions (Atchley, Wingenbach, and Akers 2013). In this work, we define dropouts from a micro-perspective, where field and institution changes are considered dropouts independently of the timing these occur. This approach leads to much higher dropout rates than the macro-perspective, which considers only students who leave the higher education system without a degree.

Namoun and Alshanqiti (Namoun and Alshanqiti 2020) performed an exhaustive search that found 62 papers published in peer-reviewed journals between 2010 and 2020, which present intelligent models to predict student performance. Additionally, in recent years, early prediction of student outcomes has attracted increasing research interest [(Saa, Al-Emran, and Shaalan 2019),(Akçapınar, Altun, and Aşkar 2019),(Daud et al. 2017), and (Martins et al. 2021)]. However, despite the research interest and the considerable amount of data that the universities generate, there is a need to collect more and better administrative data, including dropout and transfer reasons (Kehm, Larsen, and Sommersel 2019).

This project uses a dataset created from a higher education institution (acquired from several disjoint databases) related to students enrolled in different undergraduate degrees, such as agronomy, design, education, nursing, journalism, management, social service, and technologies. The dataset includes information known at the time of student enrollment (academic path, demographics, and macroeconomics and socioeconomic factors) and the students' academic performance at the end of the first and second semesters. The data are used to build classification models to predict student dropout and academic success. The problem is formulated as a three-category classification task (dropout, enrolled, and graduate) at the end of the normal duration of the course. These classification models are part of a Learning Analytic tool that includes predictive analyses which provide information to the tutoring team at our higher education institution with an estimate of the risk of dropout and failure. With this information, the tutoring team provides more accurate help to students.

The dataset contained 4424 records with 35 attributes, where each record represents an individual student and can be used for benchmarking the performance of different algorithms for solving the same type of problem and for training in the machine learning area.

In addition to this introduction section, the rest of the descriptor is organized as follows. Section 2 provides the description of the dataset. Section 3 presents the methodology that was followed and also presents a brief exploratory data analysis. Section 4 presents the conclusions, which are followed by references.

2. Data Description

The dataset is from a higher education institution (acquired from several disjoint databases) related to students enrolled between the academic years 2008/2009 (after the application of the Bologna Process to higher education in Europe) to 2018/2019 in different undergraduate degrees, such as agronomy, design, education, nursing, journalism, management, social service, and technologies. The dataset includes information known at the time of student enrollment (academic path, demographics, and macroeconomics and socioeconomic factors) and the students' academic performance at the end of the first and second semesters. The data are used to build classification models to predict student dropout and academic success.The problem is formulated as a three-category classification task (dropout, enrolled, and graduate) at the end of the normal duration of the course.

The dataset is available as a comma-separated values (CSV) file encoded as UTF8 and consists of 4424 records with 35 attributes and contains no missing values.

Table 1 describes each attribute used in the dataset grouped by class: demographic, socioeconomic, macroeconomic, academic data at enrollment, and academic data at the end of the first and second semesters. The details of the dataset the descriptions of possible values for the attributes can be obtained from Appendix 1 in (Realinho et al. 2022) which contains more detailed information about the dataset.

```r
knitr::opts_chunk$set(echo = FALSE)

library(knitr)
library(tidyverse)
library(kableExtra)
```

Attributes used grouped by class of attribute is shown in Table 1 below

Table 1: Attributes used grouped by class of attribute

Demographic data	Socioeconomic data	Macroeconomic data	Academic data at enrollment	Academic data at the end of 1st semester	Academic data at the end of 2nd semester	Target
Marital status	Mother's qualification	Unemployment rate	Application mode	Curricular units 1st sem (credited)	Curricular units 2nd sem (credited)	Target
Nationality	Father's qualification	Inflation rate	Application order	Curricular units 1st sem (enrolled)	Curricular units 2nd sem (enrolled)	Target
Displaced	Mother's occupation	GDP	Course	Curricular units 1st sem (evaluations)	Curricular units 2nd sem (approved)	Target
Gender	Father's occupation	Unemployment rate	Daytime/evening attendance	Curricular units 1st sem (approved)	Curricular units 2nd sem (grade)	Target
Age at enrollment	Educational special needs	Inflation rate	Previous qualification	Curricular units 1st sem (grade)	Curricular units 2nd sem (without evaluations)	Target
International	Debtor	GDP	Application mode	Curricular units 1st sem (without evaluations)	Curricular units 2nd sem (credited)	Target
Marital status	Tuition fees up to date	Unemployment rate	Application order	Curricular units 1st sem (credited)	Curricular units 2nd sem (enrolled)	Target
Nationality	Scholarship holder	Inflation rate	Course	Curricular units 1st sem (enrolled)	Curricular units 2nd sem (approved)	Target

2 Literature review

2.1 Related works

(S. A. Kumar et al. 2011) applied various data mining techniques on educational data. From the their results, it is clear that classification techniques can be applied on educational data for predicting the student's outcome and improve their results. The authors presents a paper entitled " Efficiency of decision trees in predicting students academic performance". They used the C4.5 Decision tree algorithm. They compare the predicted and actual results, indicating a significant improvement in results as the prediction helped identify weak and good students and help them to score better marks. They also compared the model with the ID3 Decision Tree algorithm and prove that the developed model is better in terms of efficiency and time taken to build the decision tree

(M. Kumar, Singh, and Handa 2017) researched on "Literature Survey on Educational Dropout Prediction", and analyzed different contributions of students dropout prediction in India between 2009 and 2016. The objective of this analysis was to find the existing gaps in predicting educational dropout and find the missing attributes if any in the Educational Data Mining , which may further contribute for better prediction. They stressed four kinds of studies in Educational Data Mining: Classification, Clustering, Prediction, and Association Rule mining. The machine learning classifiers found in the literature are varied. They noted that the most used ML are Support Vector Machine, Decision Tree algorithms, Artificial Neural Networks, Logistic Regression, Naïve Bayes, Random Forest, and others. The data variables used to implement the models are diversified, e.g., grade in high school, secondary school, and other related education, Gender, Family structure, Parents' Qualification, Parents' Occupation, Required for Household work, Addictions (Alcohol, Smoke, Pills, Solvents, Drugs, etc.), Basic facility in the education institution different for boys and girls, Poor Teaching methodology adopted, Got married .

(Bayer et al. 2012) presents a paper on "Predicting drop-out from social behavior of students". The authors predicted whether a bachelor student will drop out from university or not. The paper focuses on predicting drop-outs and school failures when student data has been enriched with data derived from students social behavior. These data describe social dependencies gathered from e-mail and discussion board conversations, among other sources. They worked with the data of Applied Informatics bachelor students from Masaryk University and predicted students' studies and activities via email or discussion with other students. They found students who communicate with students having good grades can successfully graduate with a higher probability than students with similar performance but not communicating with successful students. In this case, J48 decision tree learner, IB1 lazy learner, PART rule learner, SMO support vector machines have been used.

(Aman et al. 2019) presented "A Predictive Model for Predicting Students Academic Performance". The paper focuses on identifying key features, influencing students' performance, and then develop an accurate predication model for prediction of their performance, prior to taking admission in an intended program or deciding to continue for higher classes and semesters in the same program or to quit the program at this stage. In this study, first, a subjective method is used for identification of academic and socio-economic features to develop the prediction model and then a decision tree-based algorithm, Logistic Model Trees (LMT), is adopted to learn the intrinsic relationship between the identified features and students' academic grades. They predicted the students performance and compared the efficiency of two classifiers, Decision Tree and Bayesian Networks, using the WEKA tool. They used two different groups of students of undergraduate and postgraduate level. The performance of the Decision Tree was 3-12% more accurate than Bayesian networks. This research was helpful in identifying the weak students for guiding and selecting good students for scholarships.

(Adhatrao et al. 2013) used merit for examination marks,gender, and marks scored in Science, Technology and Mathematics in the examination of Grade 12 to predict student performance. A class label was retained with the expected result, either "Pass" or "Fail". As such, attributes included distinct values where there was a description of different groups to predict better outcomes. If the merit scored was 120 and above, the merit rating had a "good" value and merit was graded as "bad" if less than 120. This dataset was derived from a university database containing 123 documents.

(Aulck et al. 2016) analyzed a large, heterogeneous dataset from the University of Washington s Information school. The data included demographic information, school exit information and records from the university. They focused on cohorts over a defined period in a population of 69 116 students. Those who did not complete their studies were marked as dropouts. They applied three machine learning algorithms (regularized LR, KNN and RF) to the datasets to predict a dropout. The strongest individual predictors of student retention were the Grade Point Average (GPA) in Mathematics, English, Chemistry and Psychology classes. Regularized LR provided the strongest predictions for the dataset.

(Bergin et al. 2015) reported: "Identifying struggling students at an early stage was not easy as introductory programming modules often have a high student to lecture ratio (100:1 or greater) and early assessment may not be a reliable indicator of overall performance". The factors included, background information, perceived comfort level factors at the start of the module and motivation and use of learning strategies. Some of the background factors include among others previous academic experience for example mathematics, science and language. They used six different types of algorithms for evaluation i.e., Logistic Regression, K-Nearest Neighbor, Backpropagation, C4.5, Naïve Bayes and SVM using Sequential Minimal Optimization (SMO). Three measurement techniques e.g., overall classifier accuracy, precision and recall were employed in their study. Naïve Bayes produced the highest result among these algorithms in the study.

3 Methods

The methods used for the analysis are the Random Forest models (RF), the neural network model (NN), Support Vector Machine (SVM), Naive Bayes(NB), and decision tree (DT) algorithms. First we select variables to include in the model using random forest permutation. Then we select balanced dataset of the "Target" class. The Target class has three multilevels i.e., "Dropout", "Enrolled", and "Graduate".

Naive Bayes Classifier (NB) is a conditional probability model based in Bayes' theorem. Given a n-dimensional feature vector $(x_1, \dot{;} x_n)$ and a classification class C, the algorithm computes $p(C|x_1, \ldots, x_n)$ using the Bayes' theorem. In practice, independence between features is assumed. Combining this with a decision rule, the classification $\hat{y}$ for a feature vector $(x_1, \dot{;} x_n)$ is done as follows:

$\hat{y} = \text{argmax}\, j \in f1; 2gp(C_j) \prod_{i=1}^{n} p(x_{ij}|C_j)$

Support Vector Machine (SVM) is a classifier based on the idea of separating data using hyperplanes. More specifically, it consider a set of n±dimensional feature vectors as points in the n±dimensional real euclidean space. It supposes that each point is associated with one class (0 or 1) and solves the problem of separating the points from each class by finding the hyperplane which is at the largest distance from both points of class 0 and points of class 1. Consider the non-linear transformation $\Phi : \mathbb{R}^m \to H$ in order to represent the input vectors in a new feature space $\Phi(x) \in H$. The kernel function indicates similarity, which is obtained by scalar product between two given vectors in the transformed space $\Phi(u) \cdot \Phi(v) = K(u, v)$. The most used kernel function is:

$$\text{Gaussian} K(u, v) = \exp(-\frac{\sigma}{2}|u - v|^2)$$

Given the problem of binary classification consisting of N examples of training. Each example is indicated by a tuple (X_i, y_i) where X corresponds to the set of attributes, for example i, and the class denomination is indicated by $y_i \in 1, -1)$. The learning task with SVM can be formalized as the following constrained optimization problem:

$$\text{Max } L = \sum_{i=1}^{N} \lambda_i + \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j K(X_i, X_j) \text{ such that } \sum_{i=1}^{N} \lambda_i y_i = 0 \text{ and } \lambda_i \geq 0 \text{ for all } i$$

.

A test case Z can be classified using the equation $f(z) = \text{sign}\left(\sum_{i=1}^{n} \lambda_i y_i K(X_i, Z) + b\right)$, where λ_i is a Lagrange multiplier, b is a parameter, and K is a kernel function. SVM is a technique known to adapt well to high-dimensional data; as a limitation, it can be noted that its performance depends on the proper selection of its parameters and the kernel function

Random Forests Classifiers (RF) are an ensemble learning technique that works by constructing a multitude of Decision Trees (Quinlan 1986) and outputs the mode of the classes of the individual trees.

The ANN is composed of input layer units, hidden layer units, output layer units and connections between these layers; it is an algorithms model that simulates the neural networks of animals. The input layer unit corresponds to each variable of the input attributions, while the output layer corresponds to the variables of the category attributions. Training is a process in which the weighting of inter-layer connections is adjusted based on the training using classified data that are already known to achieve amore accurate classification of data with unknown categories. The majority of ANN are based on the multilayer feed-forward error back propagation algorithm (Wong, Bodnovich, and Selvi 1997). The main objective of this algorithm is to minimize the estimation error by calculating all the weights of the network, and systematically updating these weights to achieve the best neural network configuration.

ANN is represented as the following mathematical structure of a neuron k:

$U_k = \sum_{j=1}^{n} W_{kj} X_j \quad Y_k = f(U_k + b_k)$

where U_k represents the linear combiner, X_j are the input signals, W_{kj} are the weights for neuron k, b_k is the bias value, $f(.)$ is the activation transfer function, and Y_k is the output signal of the neuron; a detailed explanation can be found in (Lippmann 1994). Multilayer perceptron (MLP) and radial basis function (RBF) networks are widely used as supervised training methods.

Bayes' theorem is the theoretical basis of the BN, and its essence is a probability network based on probabilistic reasoning. This probability network consists of two parts, namely the directed acyclic graph and the conditional probability table. Each node in the directed acyclic graph represents a random variable, while the conditional probability table is calculated from the data set. The algorithm can be divided into the exact inference algorithm and the approximate reasoning algorithm (Plant and Holland 2011). To ensure the efficiency of the algorithm, this study adopted the relatively less complex approximate reasoning algorithm.

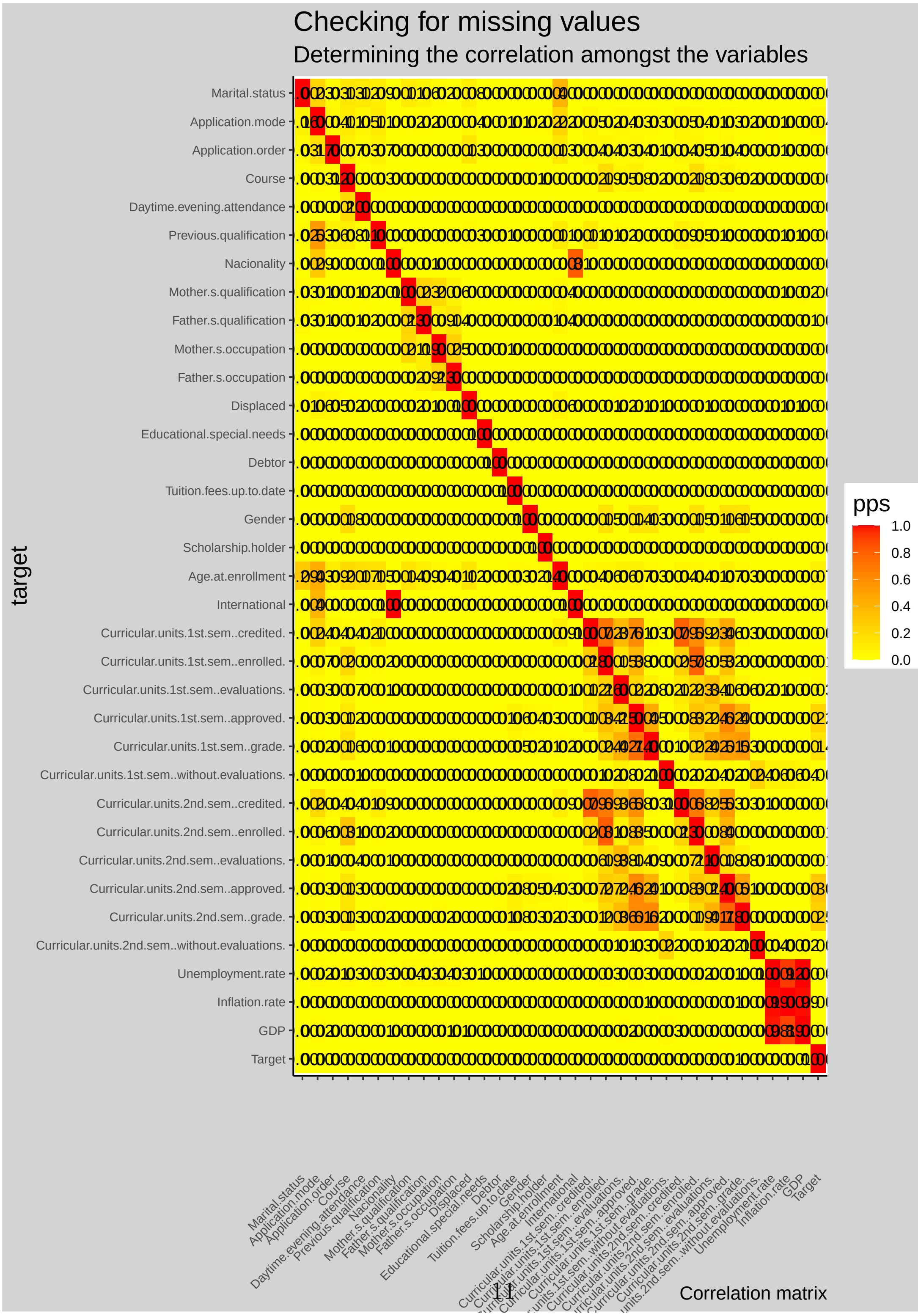
3.1 Libraries used

3.2 EDA and Feature Engineer

We performed a brief exploratory data analysis in rstudio using the predictive power score (ppsr) library, the caret library, and the ggplot2 library among the visualization tools in r.

Overview of the dataset

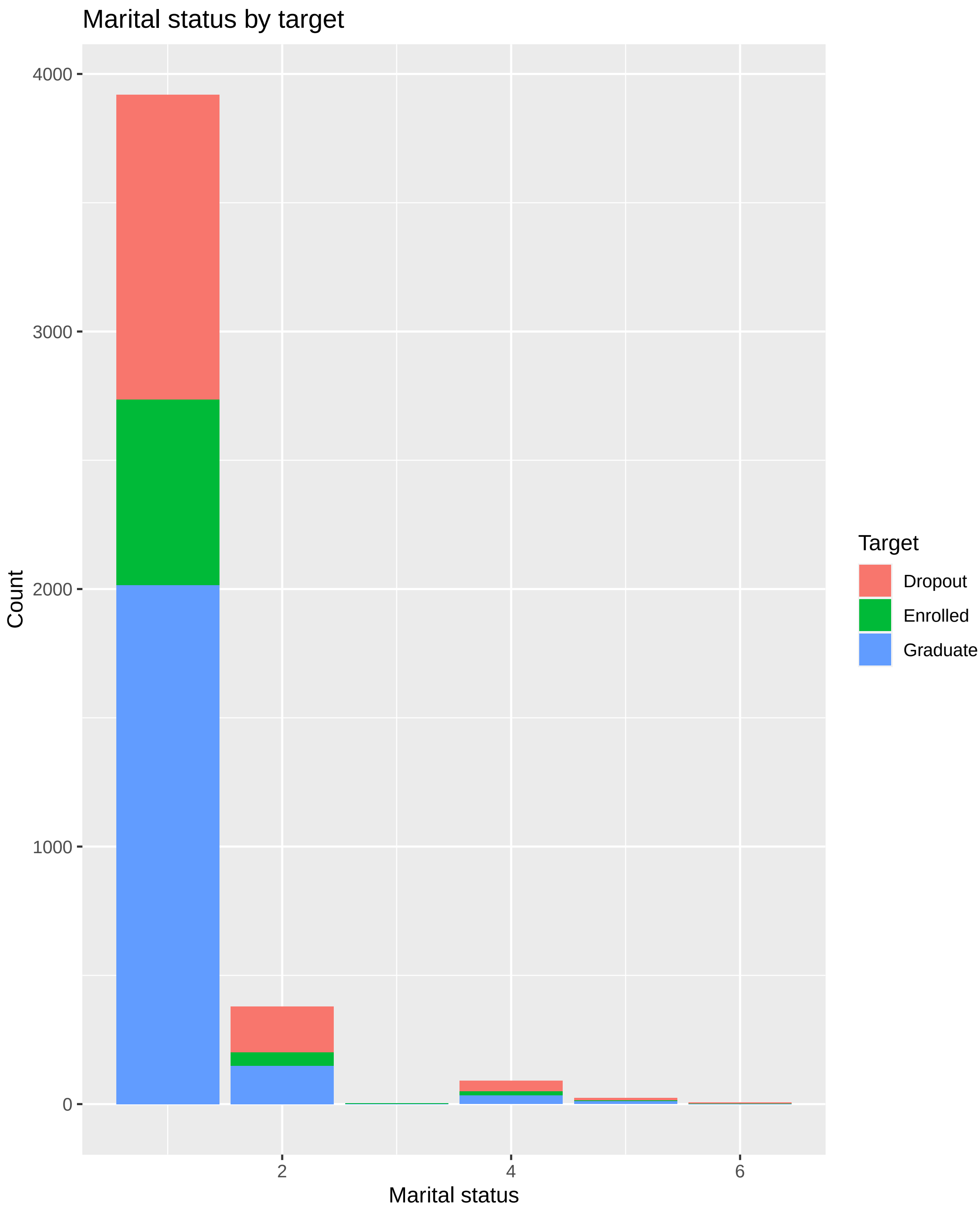
3.2.1 Correlation matrix



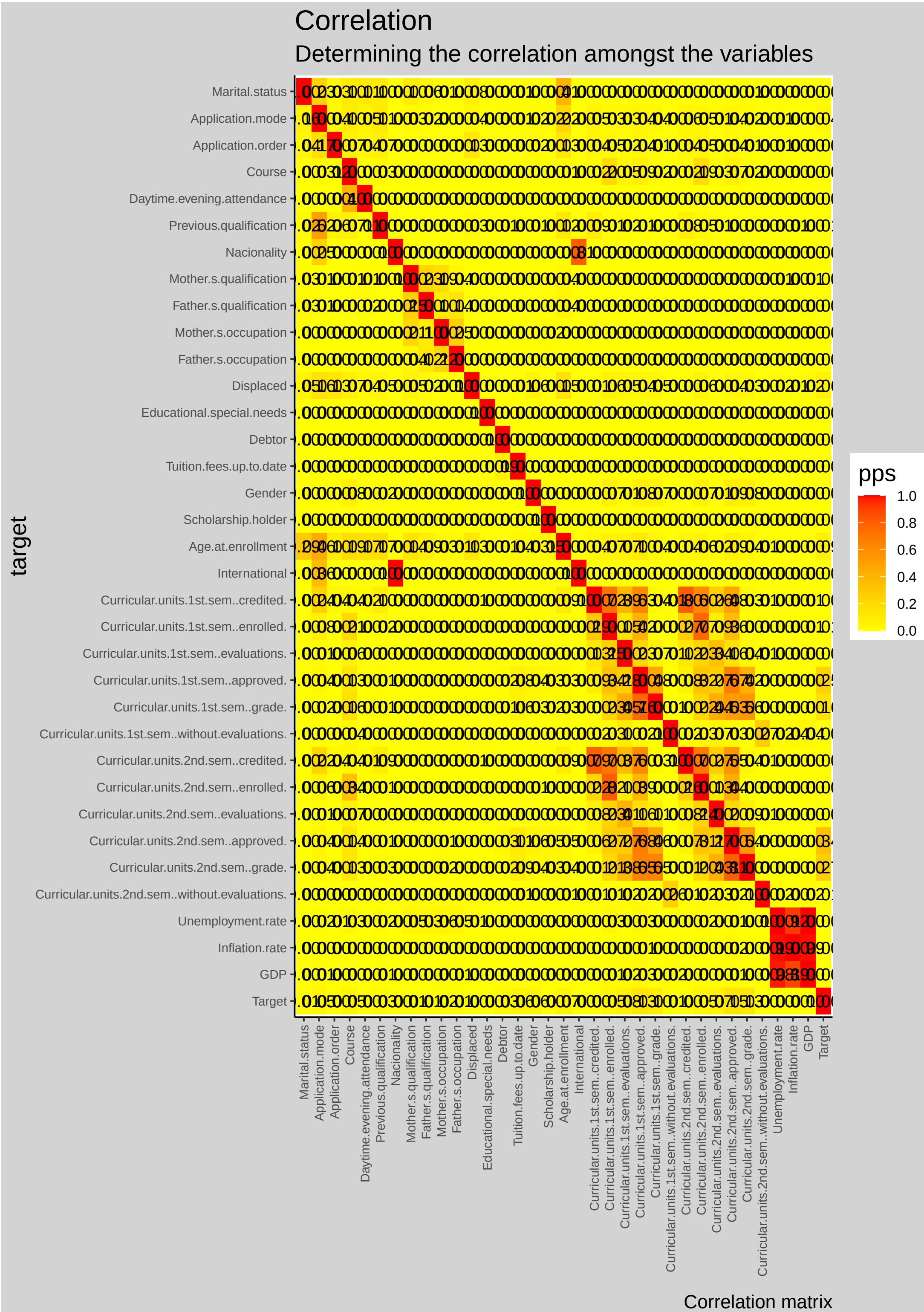
3.2.1 Data distribution and Basic statistics information

```
 Dropout Enrolled Graduate
    1421      794     2209
```

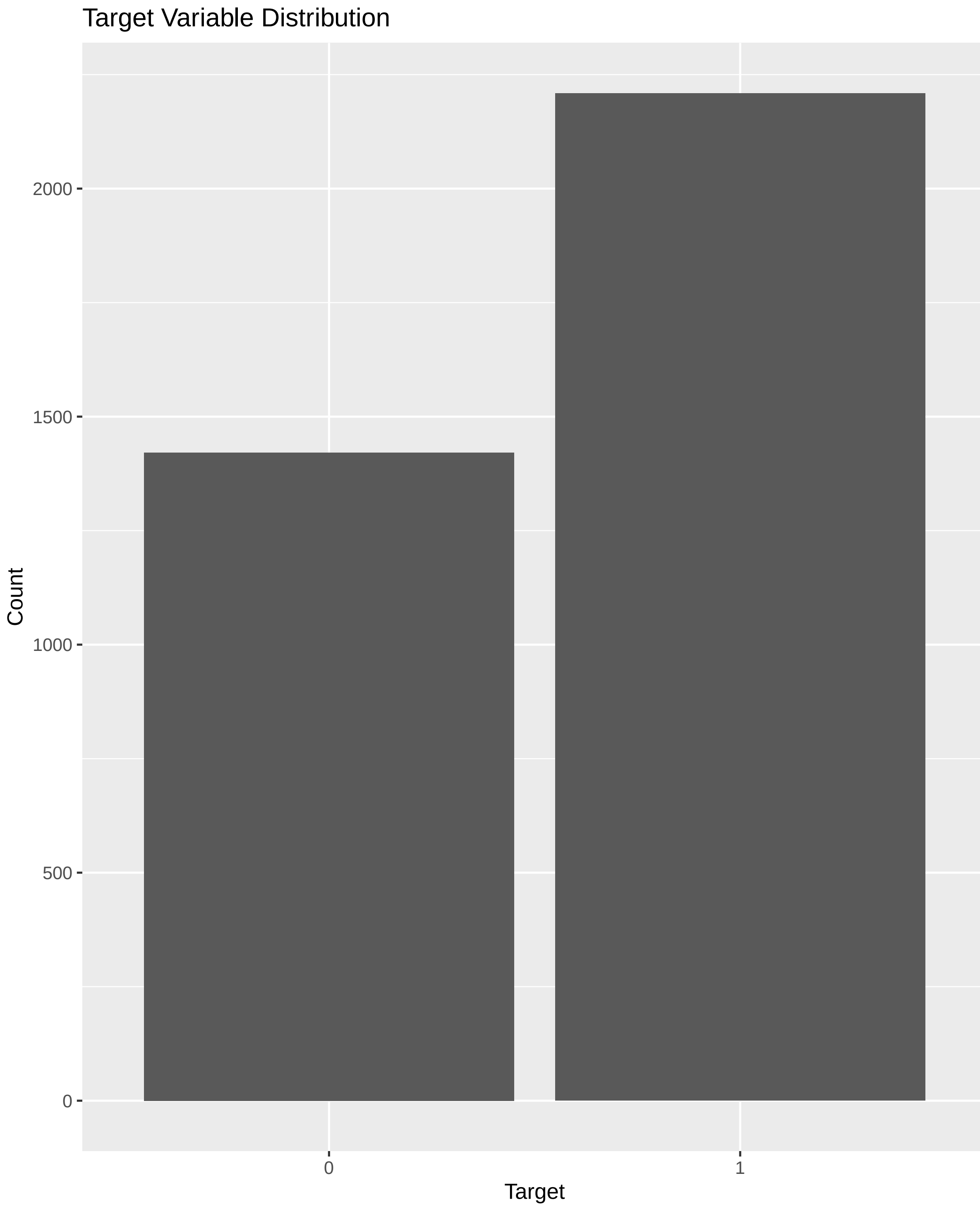
```
pdf
  2
```

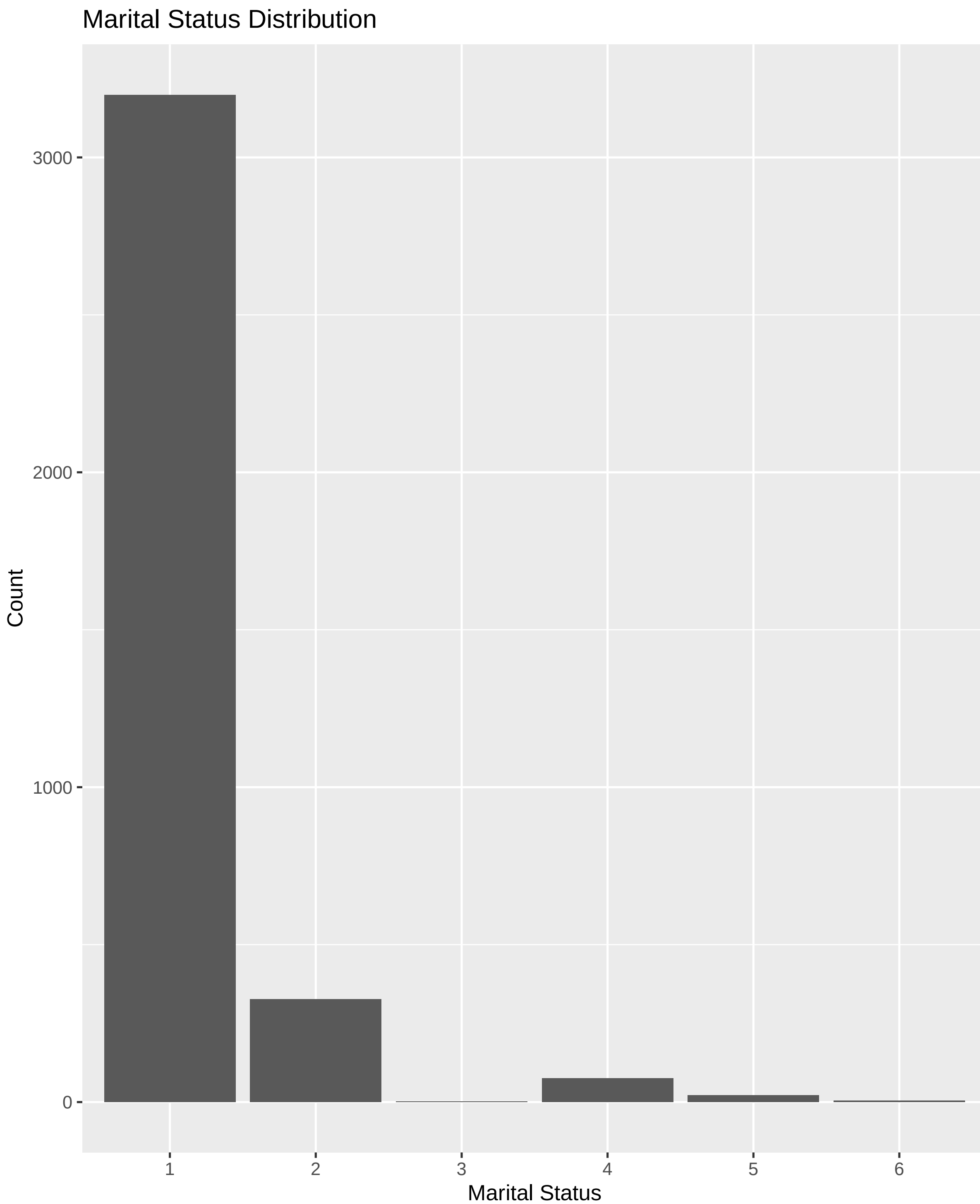


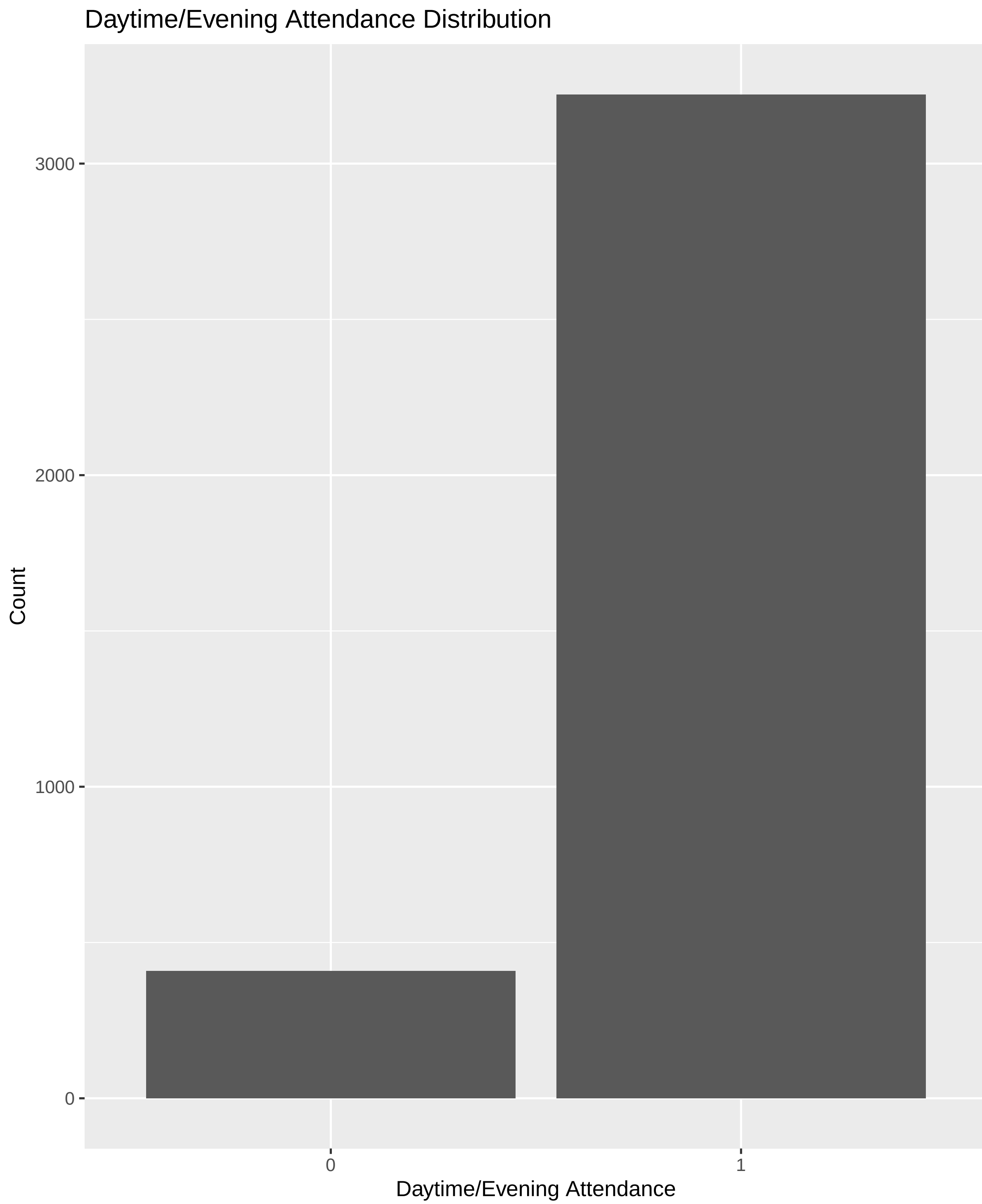
Marital status by target
4000
3000
2000
1000
0
Count
2
4
6
Marital status
Target
Dropout
Enrolled
Graduate

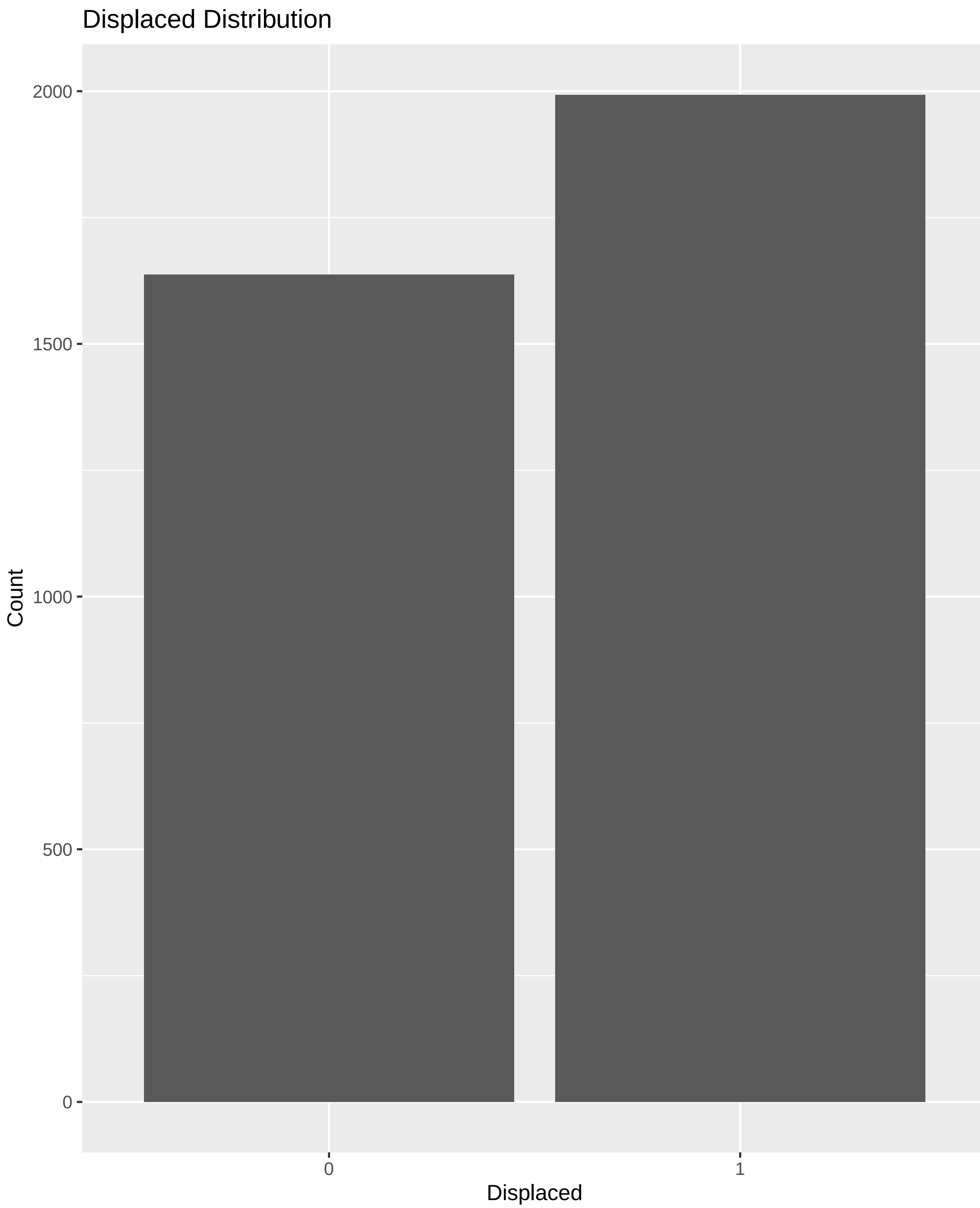

Correlation
Determining the correlation amongst the variables
target
Marital.status
Application.mode
Application.order
Course
Daytime.evening.attendance
Previous.qualification
Nacionality
Mother.s.qualification
Father.s.qualification
Mother.s.occupation
Father.s.occupation
Displaced
Educational.special.needs
Debtor
Tuition.fees.up.to.date
Gender
Scholarship.holder
Age.at.enrollment
International
Curricular.units.1st.sem..credited.
Curricular.units.1st.sem..enrolled.
Curricular.units.1st.sem..evaluations.
Curricular.units.1st.sem..approved.
Curricular.units.1st.sem..grade.
Curricular.units.1st.sem..without.evaluations.
Curricular.units.2nd.sem..credited.
Curricular.units.2nd.sem..enrolled.
Curricular.units.2nd.sem..evaluations.
Curricular.units.2nd.sem..approved.
Curricular.units.2nd.sem..grade.
Curricular.units.2nd.sem..without.evaluations.
Unemployment.rate
Inflation.rate
GDP
Target
pps
1.0
0.8
0.6
0.4
0.2
0.0
Correlation matrix

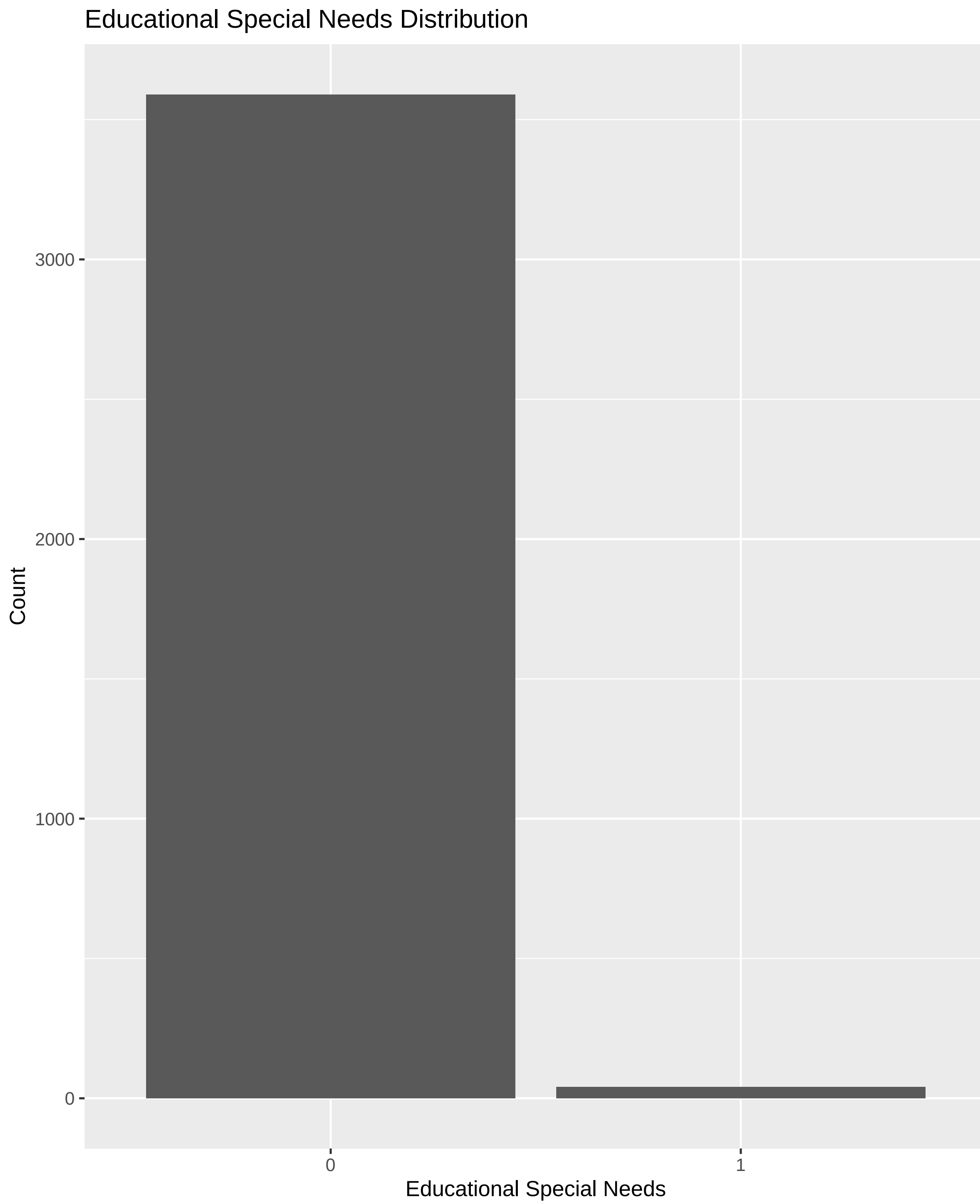
3.2.2 Descriptive Analysis

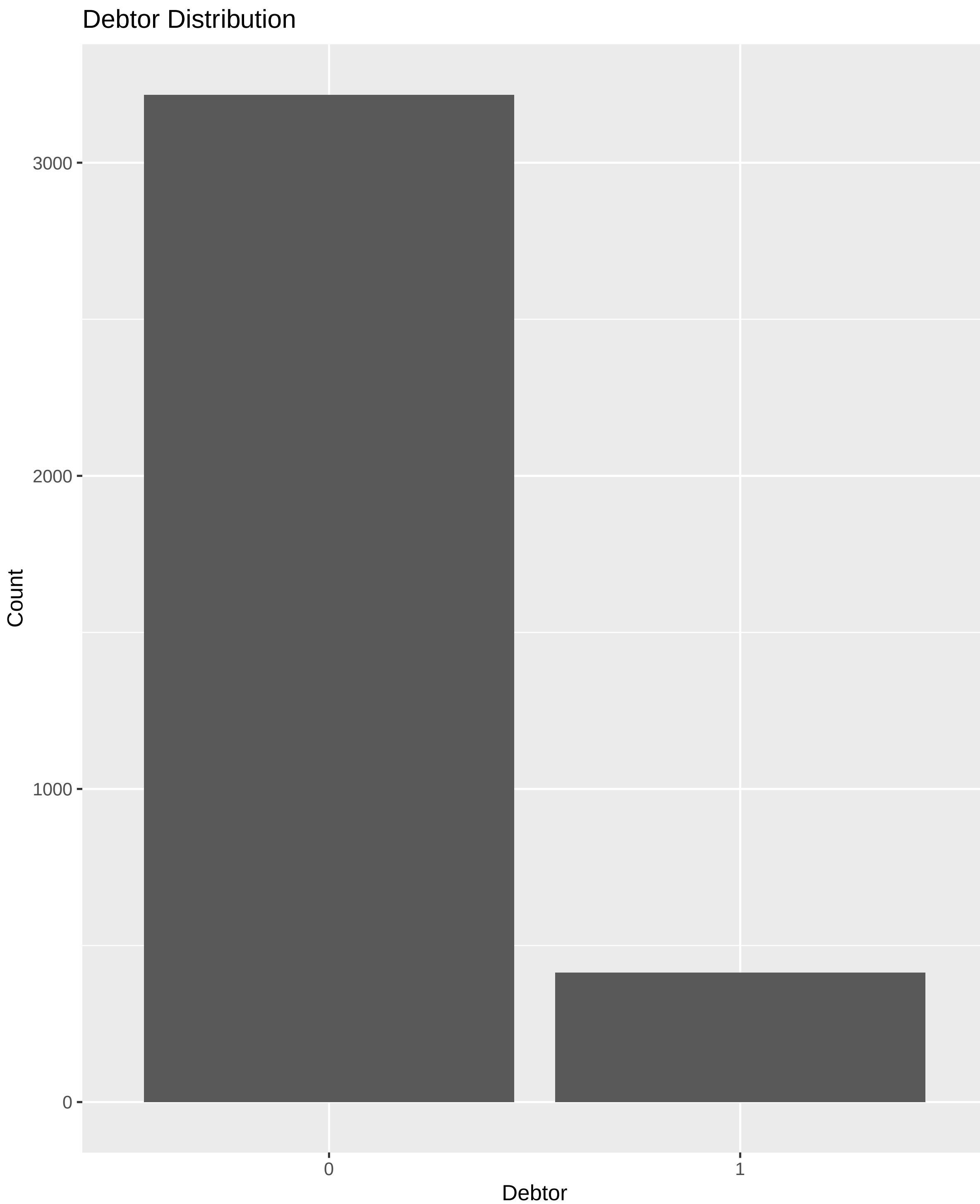


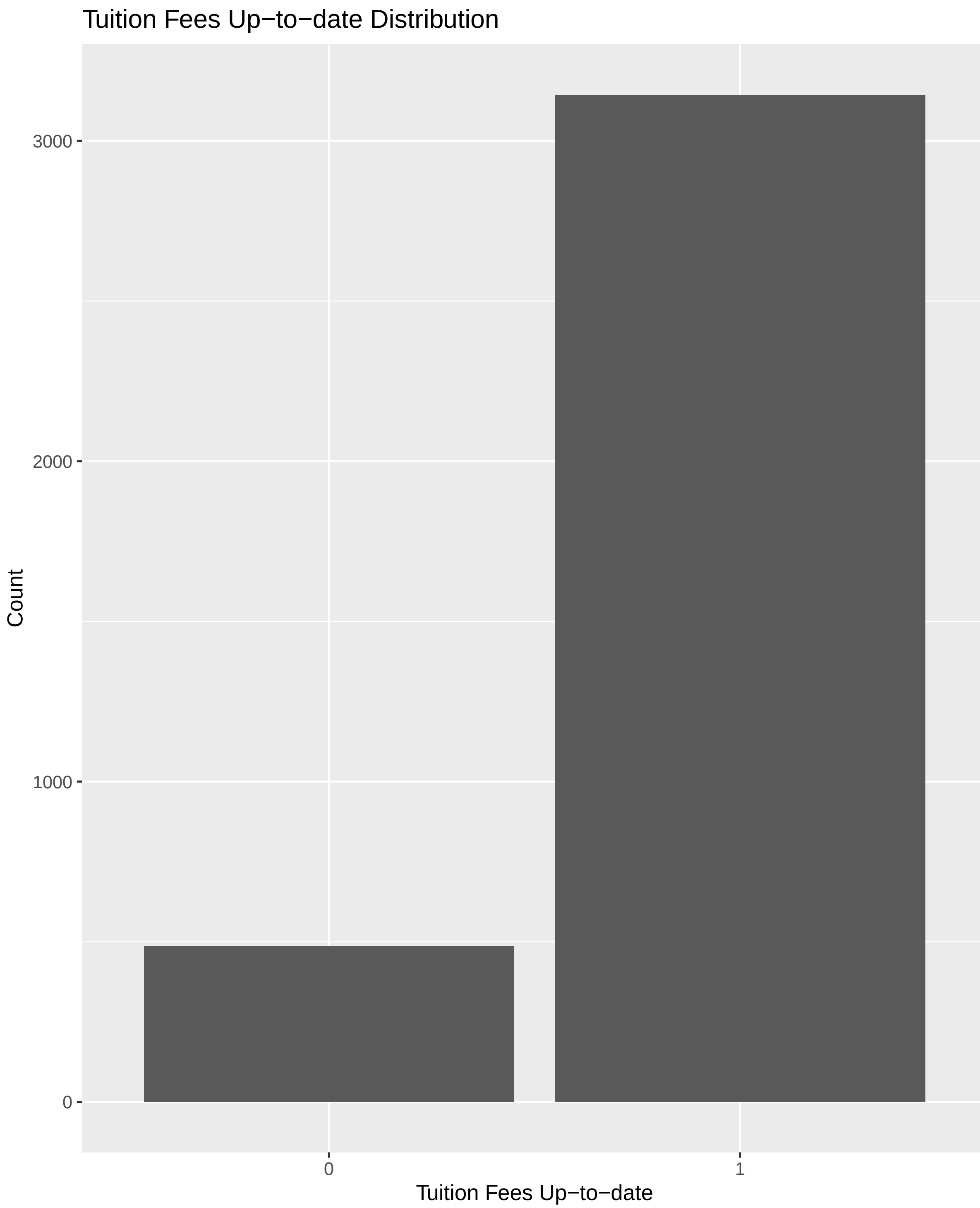

Marital Status Distribution
3000
2000
1000
0
Count
1
2
3
4
5
6
Marital Status

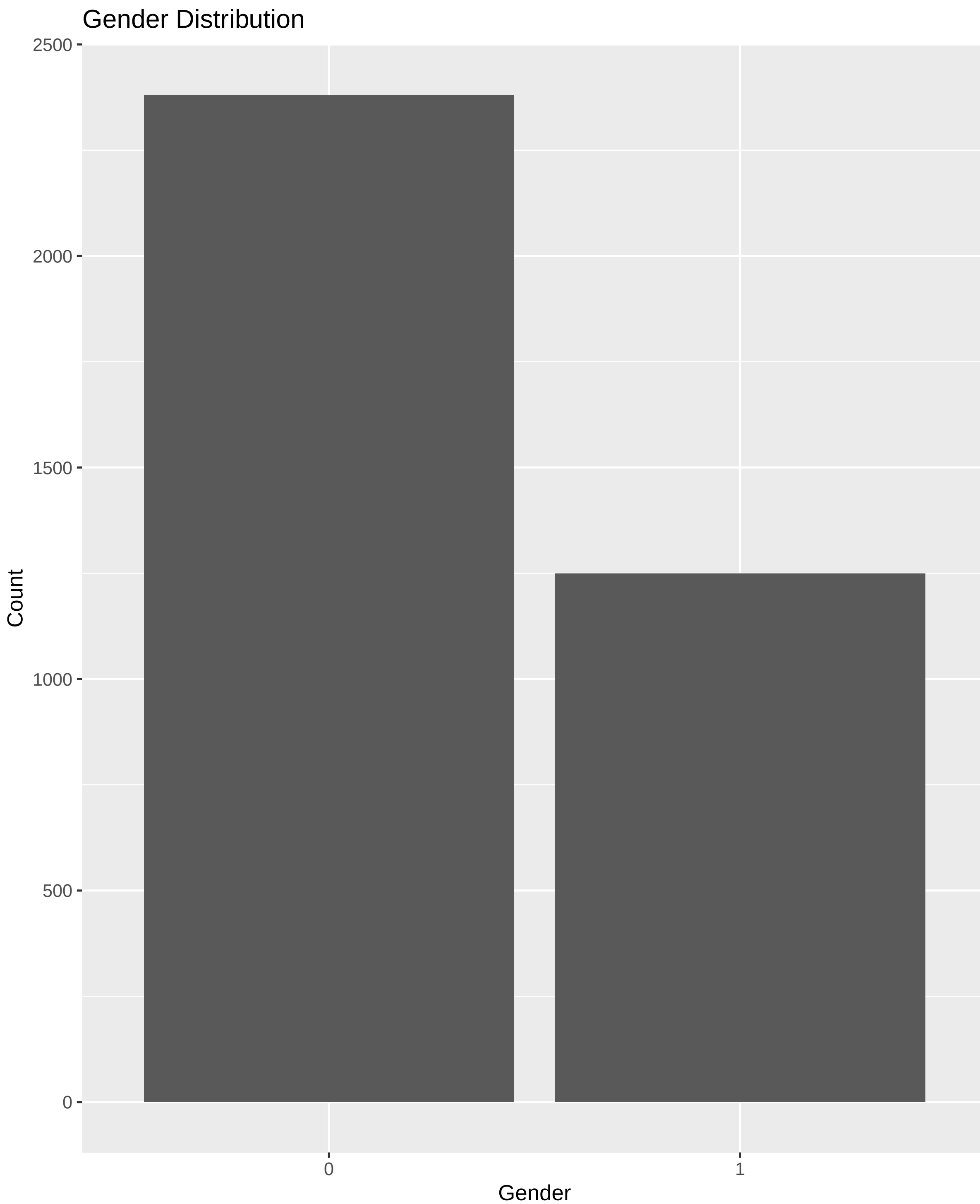

Daytime/Evening Attendance Distribution
3000
2000
1000
0
Count
0
1
Daytime/Evening Attendance

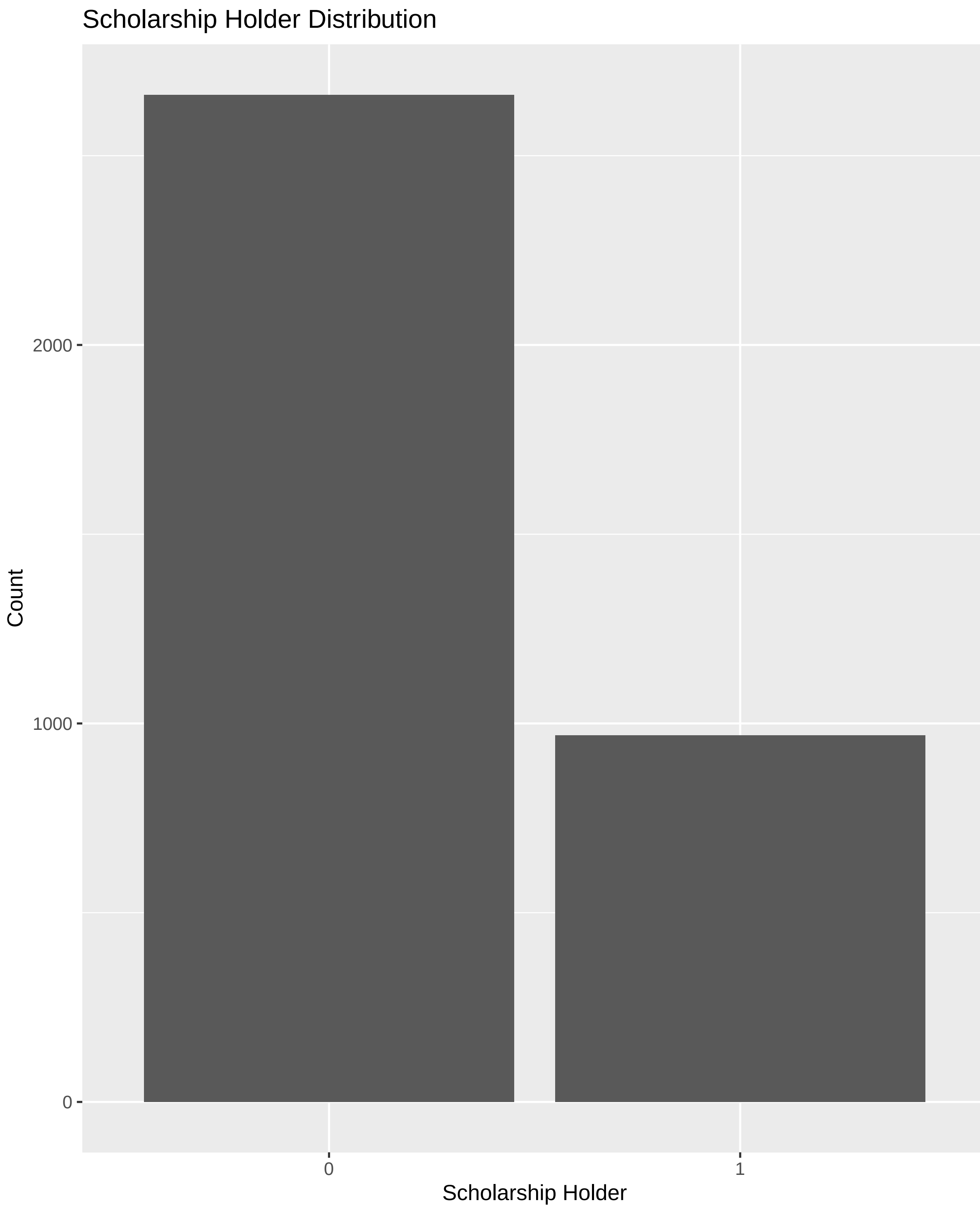

Displaced Distribution
2000
1500
1000
500
0
Count
0
1
Displaced

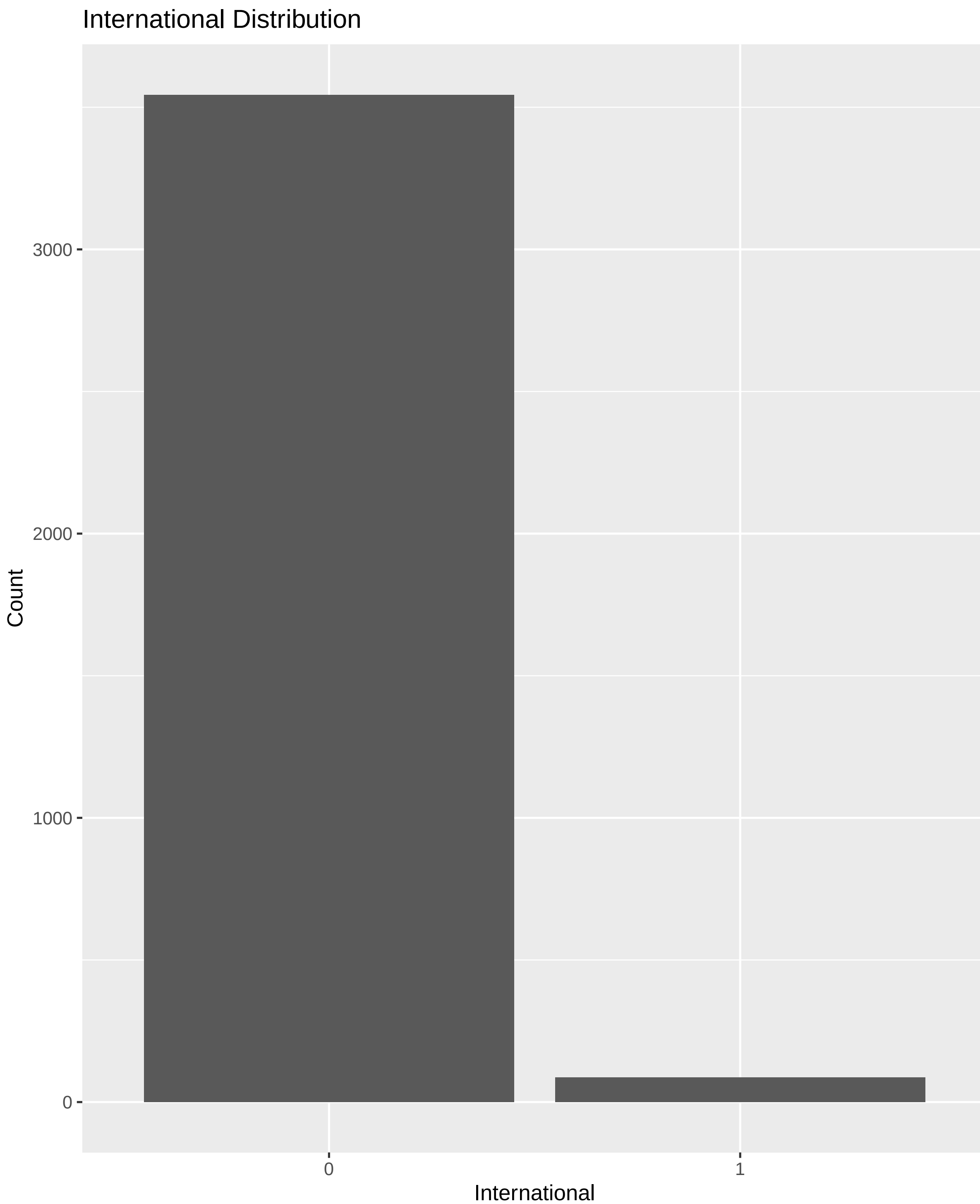

Educational Special Needs Distribution
Count
3000
2000
1000
0
0
1
Educational Special Needs


Debtor Distribution
3000
2000
1000
0
Count
0
1
Debtor


Tuition Fees Up−to−date Distribution
Count
3000
2000
1000
0
0
1
Tuition Fees Up−to−date


Gender Distribution
2500
2000
1500
1000
500
0
Count
0
1
Gender


Scholarship Holder Distribution
Count
2000
1000
0
0
1
Scholarship Holder


International Distribution
Count
3000
2000
1000
0
0
1
International

Convert data into factors

3.2.3 Subset dataset for statistical summaries

Table 2. Basic statistics information about demographic data

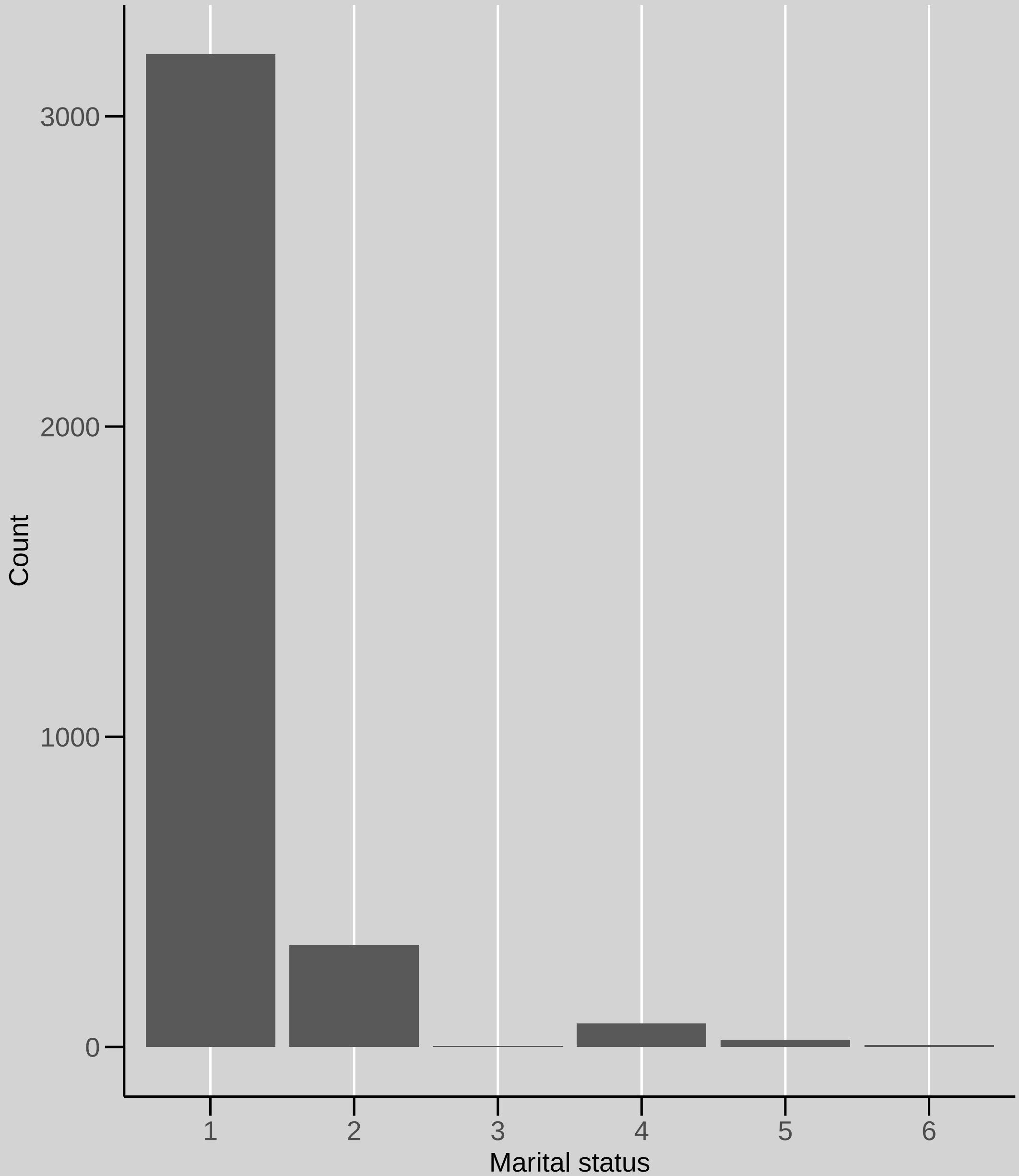


Figure 1. Distribution plot showing marital status in the demographic data.

Table 2. Basic statistics information about demographic data.

3.2.4 Data imbalance

The problem was formulated as a three-category classification task, in which there is a strong imbalance towards one of the classes (Figure 2). The majority class, Graduate, represents 50% of the records (2209 of 4424) and Dropout represents 32% of total records (1421 of 4424), while the minority class, Enrolled, represents 18% of total records (794 of 4424). This might result in a high prediction accuracy driven by the majority class at the expense of a poor performance of the minority class. Therefore, anyone using this dataset should pay attention to this problem and address it with a data-level approach or with an algorithm-level approach. At the data-level approach, a sampling technique such as the Synthetic Minority Over Sampling Technique (SMOTE) (Chawla et al. 2002) or the Adaptive Synthetic Sampling Approach (ADASYN) (Haibo He et al. 2008) or any variant thereof can be applied. At the algorithm-level approach, a machine learning algorithm that already incorporates balancing steps must be used, such as Balanced Random Forest (W. Wang, Liu, and Chan 2020) or Easy Ensemble (Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou 2009), or bagging classifiers with additional balancing, such as Exactly Balanced Bagging (Opitz and Maclin, n.d.), Roughly Balanced Bagging (Hido, Kashima, and Takahashi 2009), Over-Bagging (Opitz and Maclin, n.d.), or SMOTE-Bagging(S. Wang and Yao 2009).

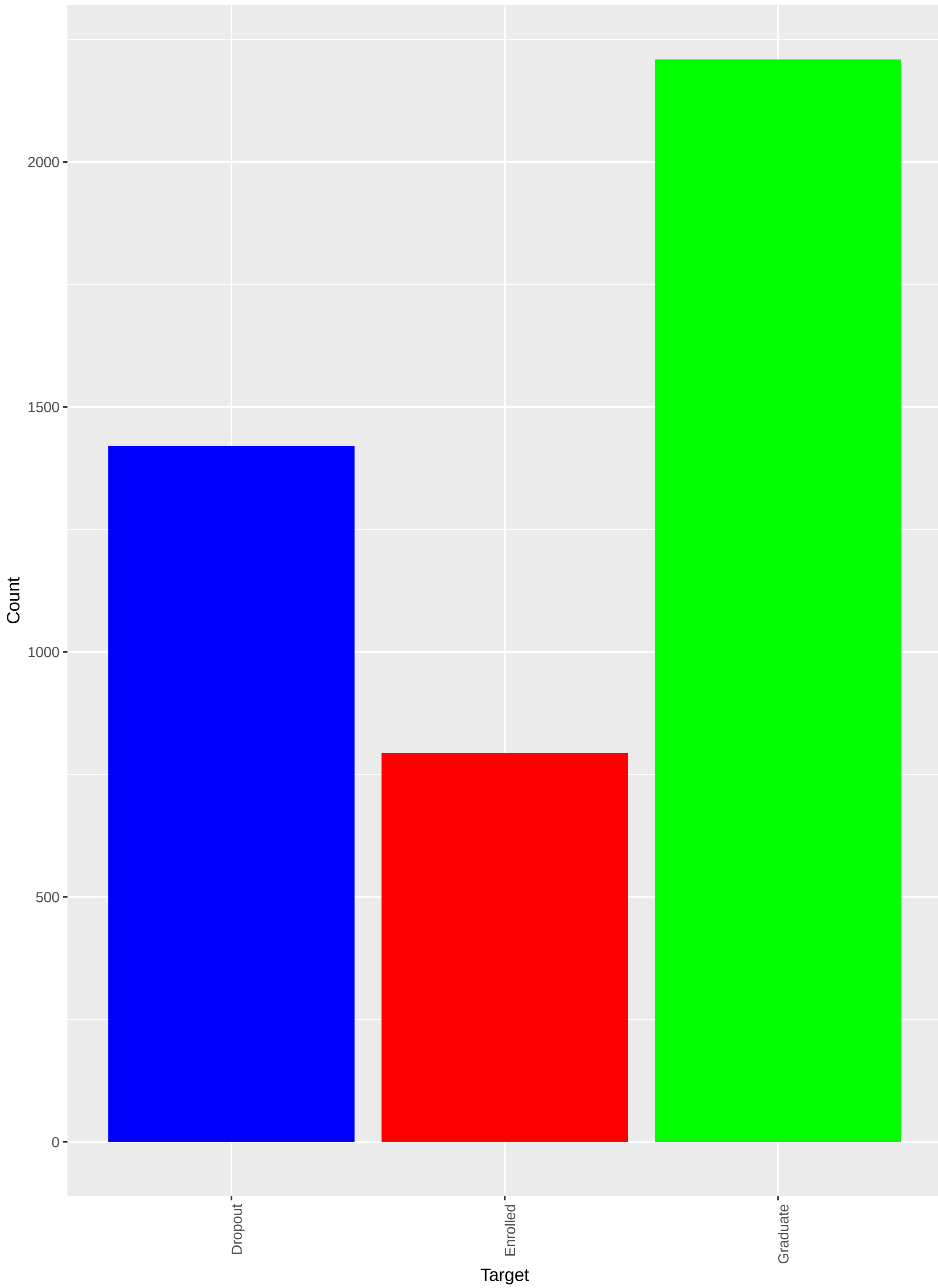


Figure 2. Distribution of student records among the three categories considered for academic success

Figure 3 shows the same imbalanced nature of data when grouping the student outcomes by course and Gender.

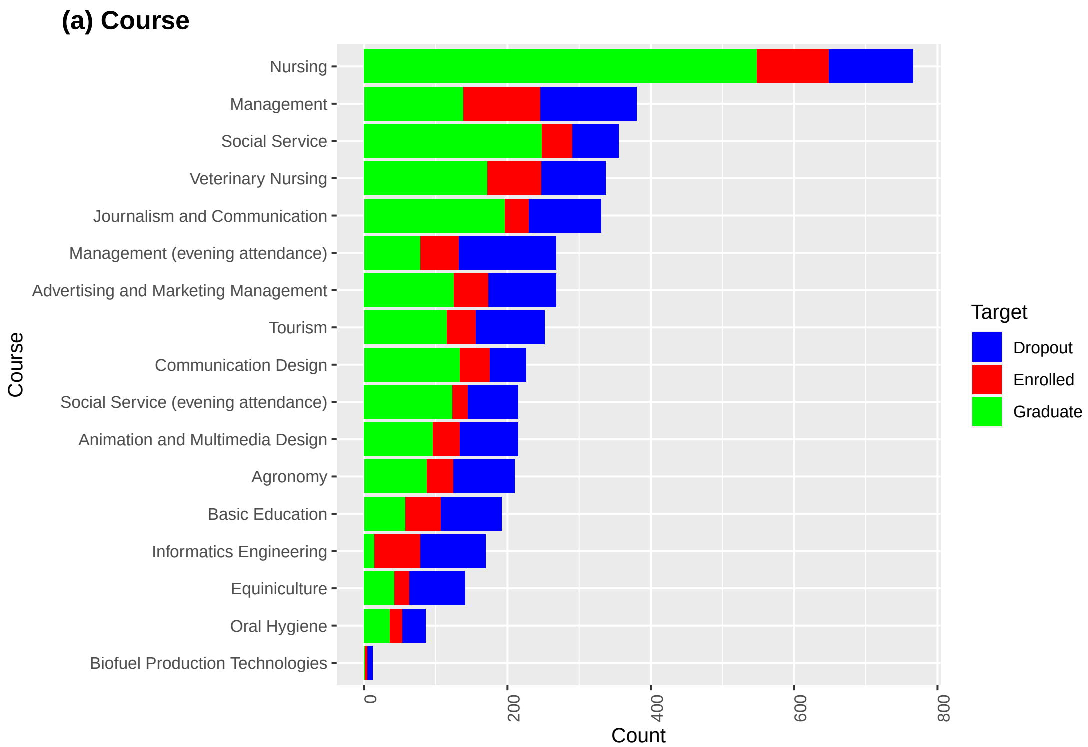


Figure 3. Student outcomes grouped by: (a) Course

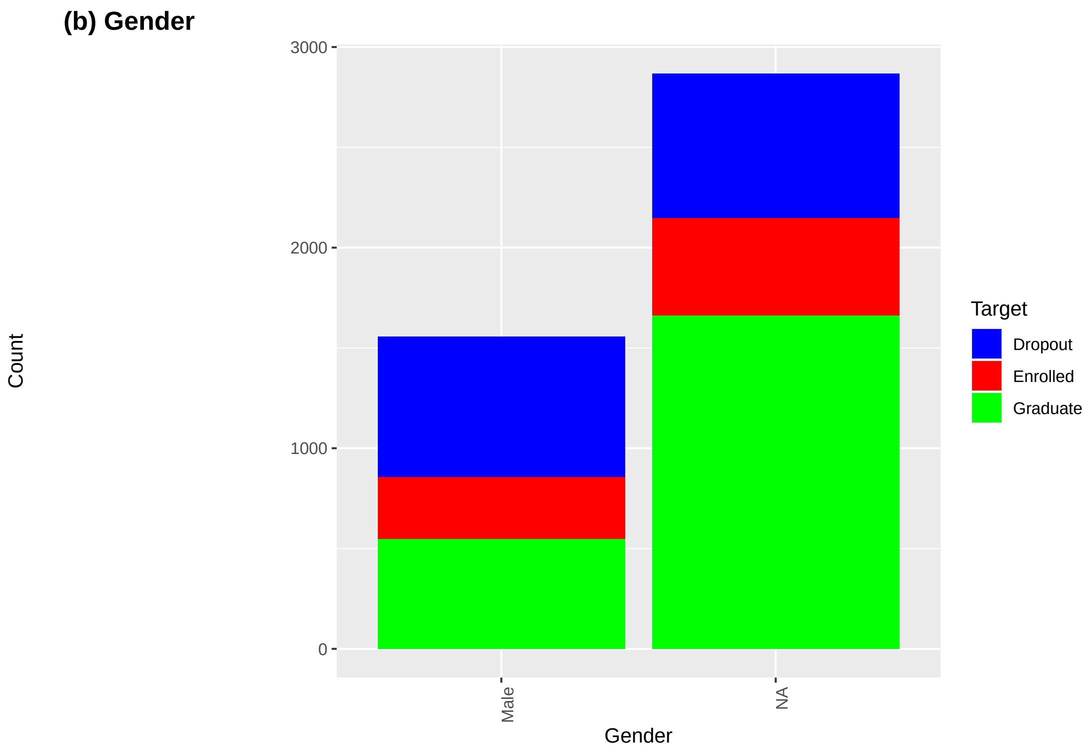


Figure 3. Student outcomes grouped by: (b) Gender

3.2.5 Feature Importance

Feature importance plays an important role in understanding the data and also in the improvement and interpretation of the machine learning models. On the other hand, useless data results in bias that messes up the final results of a machine learning problem, so feature importance is frequently used to reduce de number of features used. The most important features differ depending on the technique used to calculate the importance of each feature and also the machine learning algorithm used (Saarela and Jauhiainen 2021). One of the simplest and most used techniques to measure feature importance is Permutation Feature Importance. In this technique, feature importance is calculated by noticing the increase or decrease in error when we permute the values of a feature. If permuting the values causes a huge change in the error, it means the feature is important for our model.

We performed a test to determine the most important features considering the Permutation Feature Importance, using F1 as the error metric, which is a metric more adequate for imbalanced data, taking into account the trade-off between precision and recall. The Permutation Feature Importance was applied to some of the most interesting results reported in the literature for multiclass imbalanced classification (Spelmen and Porkodi 2018) and (Ali et al. 2019).

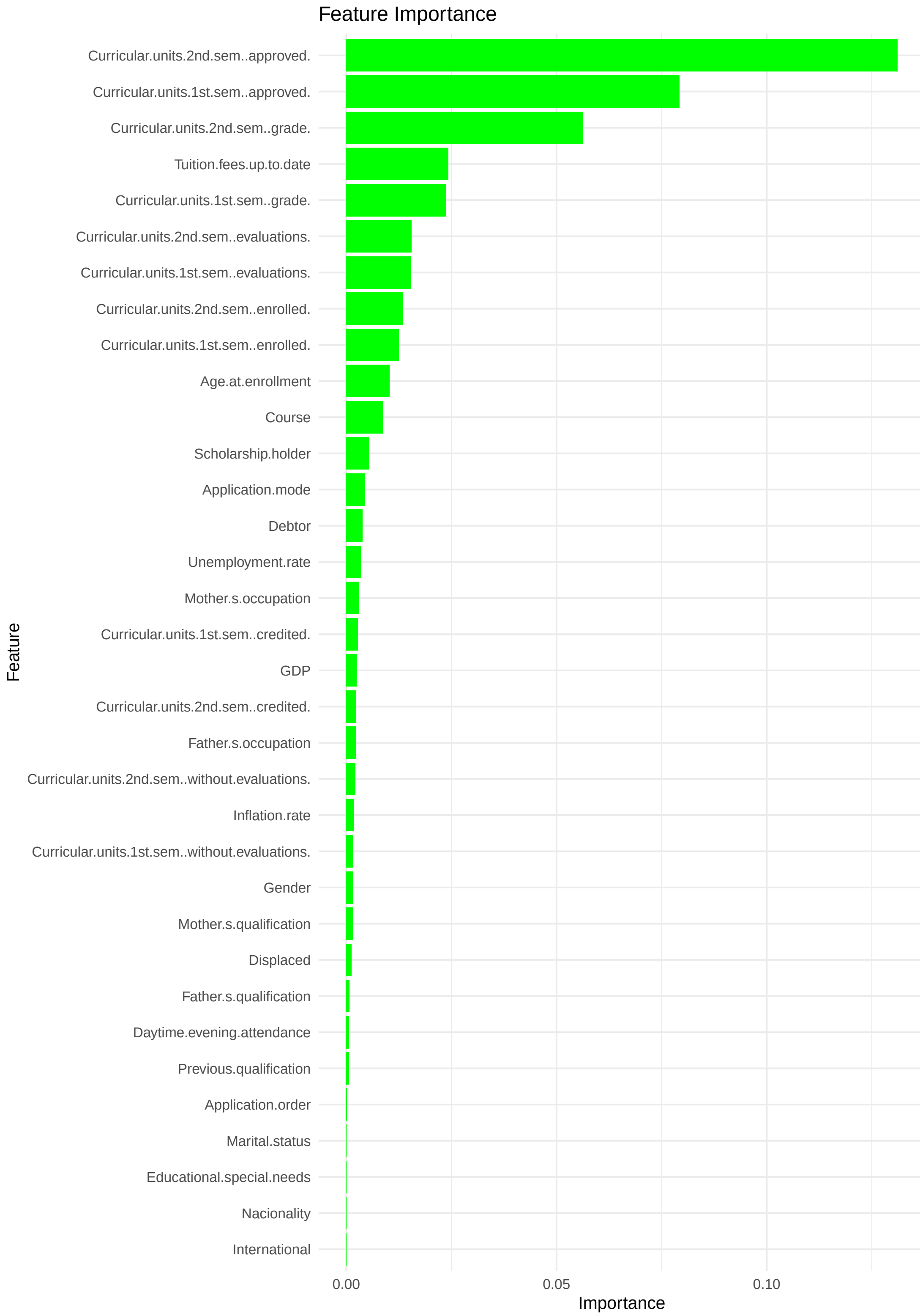
3.2.6 Split the dataset into training and testing sets

We split the data into ratio 70% for training and 30% for testing using random split.

3.2.7 Permutation Feature Importance for Random Forest

Feature importance plays an important role in understanding the data and also in the improvement and interpretation of the machine learning models. On the other hand,useless data results in bias that messes up the final results of a machine learning problem, so feature importance is frequently used to reduce de number of features used. The most important features differ depending on the technique used to calculate the importance of each feature and also the machine learning algorithm used (Saarela and Jauhiainen 2021). One of the simplest and most used techniques to measure feature importance is Permutation Feature Importance. In this technique, feature importance is calculated by noticing the increase or decrease in error when we permute the values of a feature. If permuting the values causes a huge change in the error, it means the feature is important for our model.

We performed a test to determine the most important features considering the Permutation Feature Importance, using F1 as the error metric, which is a metric more adequate for imbalanced data, taking into account the trade-off between precision and recall.



Feature Importance
Curricular.units.2nd.sem..approved.
Curricular.units.1st.sem..approved.
Curricular.units.2nd.sem..grade.
Tuition.fees.up.to.date
Curricular.units.1st.sem..grade.
Curricular.units.2nd.sem..evaluations.
Curricular.units.1st.sem..evaluations.
Curricular.units.2nd.sem..enrolled.
Curricular.units.1st.sem..enrolled.
Age.at.enrollment
Course
Scholarship.holder
Application.mode
Debtor
Unemployment.rate
Mother.s.occupation
Curricular.units.1st.sem..credited.
GDP
Curricular.units.2nd.sem..credited.
Father.s.occupation
Curricular.units.2nd.sem..without.evaluations.
Inflation.rate
Curricular.units.1st.sem..without.evaluations.
Gender
Mother.s.qualification
Displaced
Father.s.qualification
Daytime.evening.attendance
Previous.qualification
Application.order
Marital.status
Educational.special.needs
Nacionality
International
Feature
0.00
0.05
0.10
Importance

3.3 Data Analysis

3.3.1 Re-sample training and test data to balance the classes

```
 Dropout Enrolled Graduate
    1547     1547     1547
```

```
 Dropout Enrolled Graduate
     662      662      662
```

3.4 Define the model

We have implemented the following models" the Random Forest models (RF), the neural network model (NN), Support Vector Machine (SVM), Naive Bayes(NB), and decision tree (DT) algorithms.

3.4.1 Random Forest, Decision Tree, Naive Bayes and SVM models

3.4.2 Define the model training and testing functions

3.4.3 Train and test the models

```
           Dropout Enrolled Graduate
  Dropout      991      219       65
  Enrolled     433     1008      222
  Graduate     123      320     1260
```

```
svm_pred   Dropout Enrolled Graduate
  Dropout      428       80       27
  Enrolled     189      425       95
  Graduate      45      157      540
```

```
          Reference
Prediction Dropout Enrolled Graduate
  Dropout      464      108       30
  Enrolled     128      396      132
  Graduate      70      158      500
```

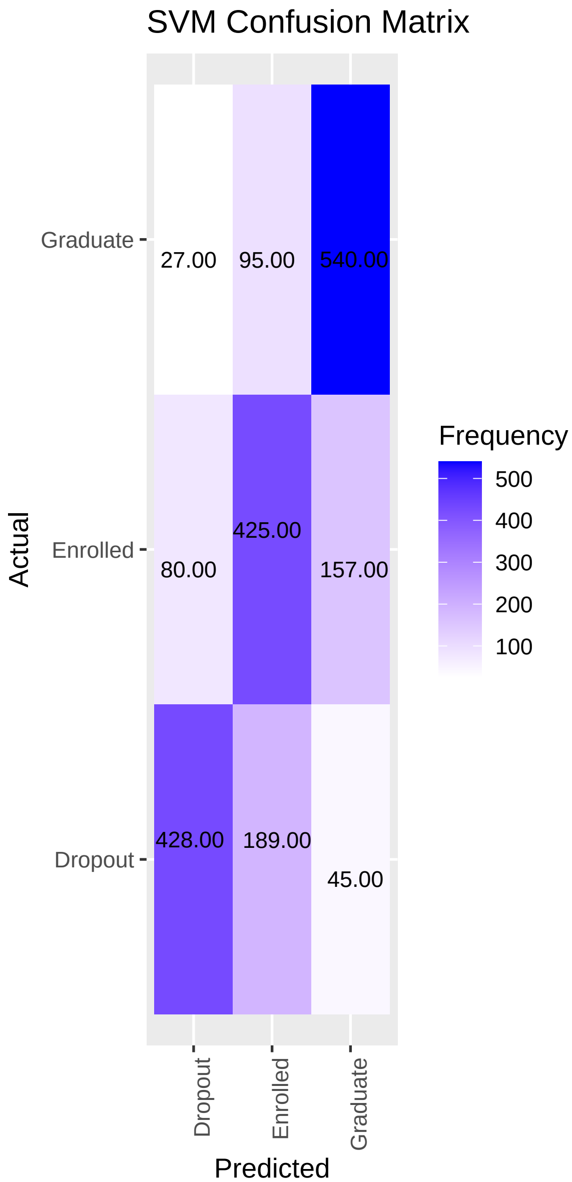
```
          Dropout Enrolled Graduate
  Dropout     1532       52       34
  Enrolled       1     1484        1
  Graduate      14       11     1512
```

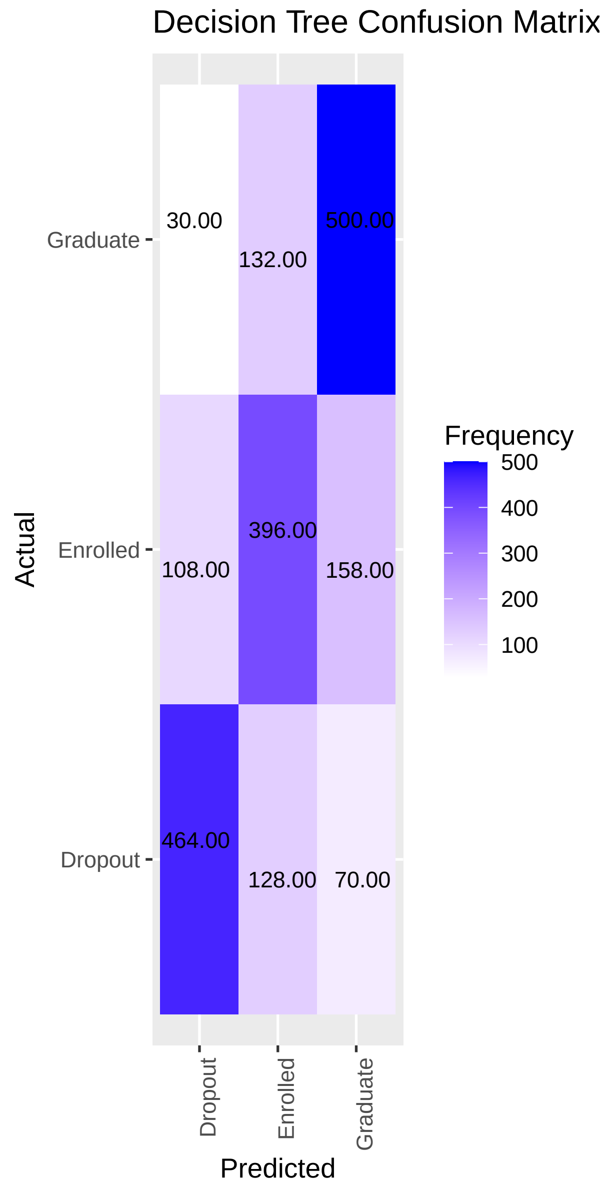
```
rf_pred    Dropout Enrolled Graduate
  Dropout      490      169       26
  Enrolled     121      305       78
  Graduate      51      188      558
```

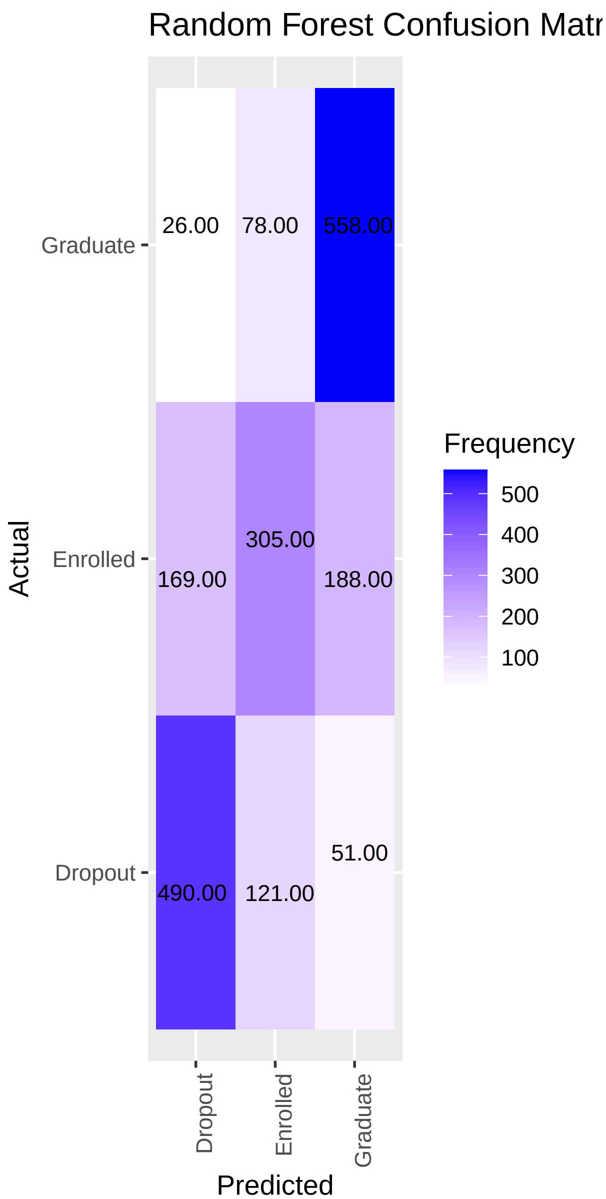
```
nb_train   Dropout Enrolled Graduate
  Dropout     1009      227       87
  Enrolled     321      731      134
  Graduate     217      589     1326
```

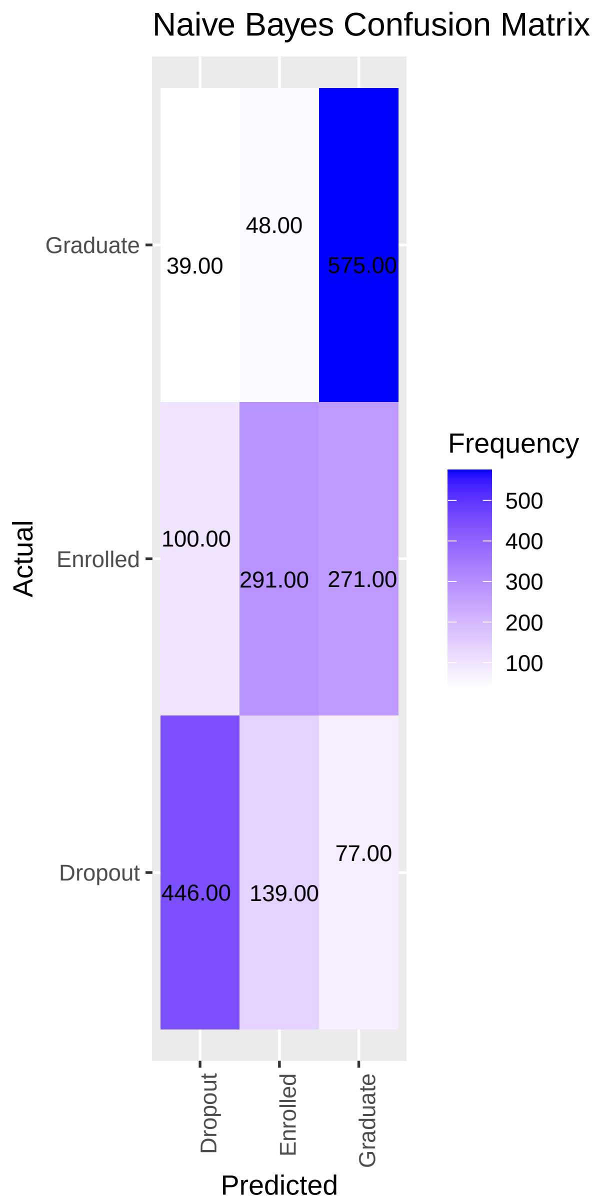
```
nb_pred    Dropout Enrolled Graduate
  Dropout      446      100       39
  Enrolled     139      291       48
  Graduate      77      271      575
```

3.4.4 Plot confusion matrix for each model



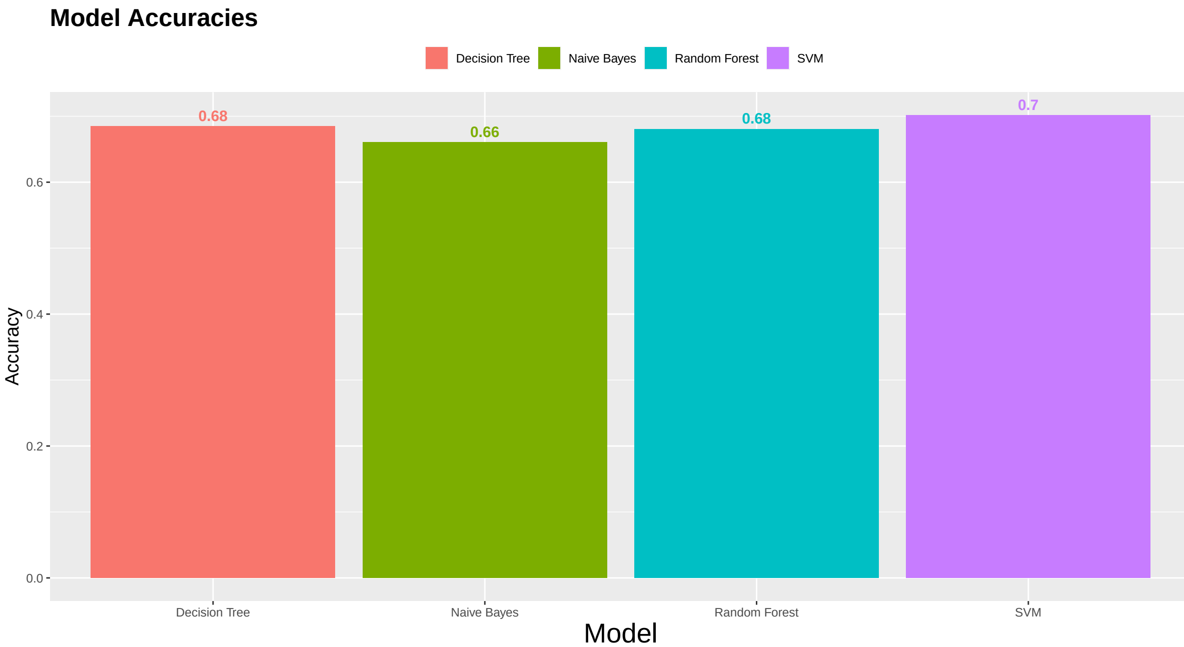






Create a data frame of the accuracies

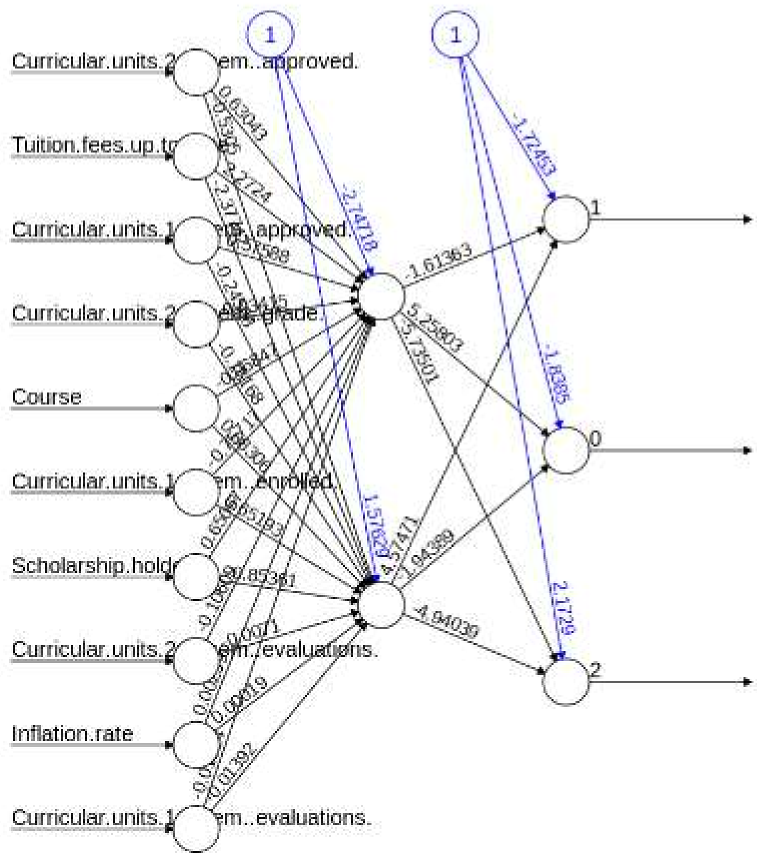
3.4.5 Plot the accuracies



3.4.6 neural net work

We created two hidden layers with two neurons and stepmax of 10^7 and the linear output argument is set to false to indicate that the output is binary. The results are shown below

```
nn = neuralnet(
     f,
data=train_data,
hidden=2,
linear.output = FALSE,
stepmax=1e7
)
plot(nn, rep = 1)
```



```
[ ] nn$result.matrix
```

A matrix: 34 × 1 of type dbl

error	5.039430e+02
reached.threshold	9.703382e-03
steps	2.228700e+04
Intercept.to.1layhid1	-2.747183e+00
Curricular.units.2nd.sem..approved..to.1layhid1	6.304325e-01
Tuition.fees.up.to.date.to.1layhid1	2.272404e+00
Curricular.units.1st.sem..approved..to.1layhid1	5.758844e-01
Curricular.units.2nd.sem..grade..to.1layhid1	3.415003e-02
Course.to.1layhid1	-6.847251e-02
Curricular.units.1st.sem..enrolled..to.1layhid1	-7.571709e-01
Scholarship.holder.to.1layhid1	6.508750e-01
Curricular.units.2nd.sem..evaluations..to.1layhid1	-1.060942e-01
Inflation.rate.to.1layhid1	2.388776e-03
Curricular.units.1st.sem..evaluations..to.1layhid1	-1.324310e-02
Intercept.to.1layhid2	1.576295e+00
Curricular.units.2nd.sem..approved..to.1layhid2	-5.305014e-01
Tuition.fees.up.to.date.to.1layhid2	-2.377416e+00
Curricular.units.1st.sem..approved..to.1layhid2	-2.459792e-01

```
# print the accuracy and confusion matrix
print('Accuracy : ' + str(accuracy_score(Y_test, y_pred)))
conf_m = confusion_matrix(Y_test, y_pred)
print(conf_m)
```

```
Accuracy : 0.6971751412429379
[[218   0  72]
 [ 65   0  95]
 [ 36   0 399]]
```

The accuracy of the neural network is 69.7%. We used

In this analysis, we compared the performance of four machine learning models for predicting a binary classification task. The models used are artificial neural networks (ANN), support vector machines (SVM), decision trees (DT), Random Forest (RF), and Naive Bayes networks (NB). We evaluate the accuracy of each model and plotted them respectively.

Conclusion

We fit each model to the training data and make predictions on the testing data. The ANN model has two hidden layers with 4 and 3 nodes, respectively. The SVM model uses a radial basis function kernel and has probability estimates enabled. The DT model uses the classification method and the default splitting criterion. The NB model is learned using the Hill-Climbing algorithm with the Akaike Information Criterion (AIC) as the scoring metric.

We calculated the accuracy for each model and plotted them respectively. The RF model has the highest accuracy, followed by the SVM and DT models. The Naive Bayes and NN models have the lowest accuracies. We did not do hyperparameter tunings on these models.

We examined the factors that could be used to identify a student at risk of dropout through Permutation Feature Importance using Random Forest.

A higher Accuracy score obtained from our model implies a better and acceptable classification performance because points representing model classification better than random guess are located above the diagonal line.

The analysis was made on the model performance and we considered the actual statistical composition of the dataset, which is highly unbalanced to the classes.

Random Forest (RF) (0.72) and SVM (0.71) had achieved a better accuracies compared to the other models such as DT (0.68), NB (0.65) and NN(0.70) respectively.

References

Adhatrao, Kalpesh, Aditya Gaykar, Amiraj Dhawan, Rohit Jha, and Vipul Honrao. 2013. "Predicting Students' Performance Using ID3 and C4.5 Classification Algorithms." *International Journal of Data Mining & Knowledge Management Process* 3 (5): 39–52. https://doi.org/10.5121/ijdkp.2013.3504.

Akçapınar, Gökhan, Arif Altun, and Petek Aşkar. 2019. "Using Learning Analytics to Develop Early-Warning System for at-Risk Students." *International Journal of Educational Technology in Higher Education* 16 (1). https://doi.org/10.1186/s41239-019-0172-z.

Ali, Haseeb, Mohd Najib Mohd Salleh, Rohmat Saedudin, Kashif Hussain, and Muhammad Faheem Mushtaq. 2019. "Imbalance Class Problems in Data Mining: A Review." *Indonesian Journal of Electrical Engineering and Computer Science* 14 (3): 1552. https://doi.org/10.11591/ijeecs.v14.i3.pp1552-1563.

Aman, Fazal, Azhar Rauf, Rahman Ali, Farkhund Iqbal, and Asad Masood Khattak. 2019. "A Predictive Model for Predicting Students Academic Performance." *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, July. https://doi.org/10.1109/iisa.2019.8900760.

Atchley, Thomas Wayne, Gary Wingenbach, and Cynthia Akers. 2013. "Comparison of Course Completion and Student Performance Through Online and Traditional Courses." *The International Review of Research in Open and Distributed Learning* 14 (4). https://doi.org/10.19173/irrodl.v14i4.1461.

Aulck, Lovenoor, Nishant Velagapudi, Joshua Blumenstock, and Jevin West. 2016. "Predicting Student Dropout in Higher Education." *arXiv Preprint arXiv:1606.06364*.

Bayer, Jaroslav, Hana Bydzovská, Jan Géryk, Tomás Obsivac, and Lubomir Popelinsky. 2012. "Predicting Drop-Out from Social Behaviour of Students." *International Educational Data Mining Society*.

Behr, Andreas, Marco Giese, Hervé D. Teguim K., and Katja Theune. 2020. "Dropping Out from Higher Education in Germany an Empirical Evaluation of Determinants for Bachelor Students." *Open Education Studies* 2 (1): 126–48. https://doi.org/10.1515/edu-2020-0104.

Bergin, Susan, Aidan Mooney, John Ghent, and Keith Quille. 2015. "Using Machine Learning Techniques to Predict Introductory Programming Performance." *International Journal of Computer Science and Software Engineering (IJCSSE)* 4 (12): 323–28.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. "SMOTE: Synthetic Minority over-Sampling Technique." *Journal of Artificial Intelligence Research* 16 (June): 321–57. https://doi.org/10.1613/jair.953.

Daud, Ali, Naif Radi Aljohani, Rabeeh Ayaz Abbasi, Miltiadis D. Lytras, Farhat Abbas, and Jalal S. Alowibdi. 2017. "Predicting Student Performance Using Advanced Learning Analytics." *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*. https://doi.org/10.1145/3041021.3054164.

Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. 2008. "ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning." *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, June. https:

//doi.org/10.1109/ijcnn.2008.4633969.

Hido, Shohei, Hisashi Kashima, and Yutaka Takahashi. 2009. "Roughly Balanced Bagging for Imbalanced Data." *Statistical Analysis and Data Mining* 2 (5â 6): 412–26. https://doi.org/10.1002/sam.10061.

Kehm, Barbara M., Malene Rode Larsen, and Hanna Bjørnøy Sommersel. 2019. "Student Dropout from Universities in Europe: A Review of Empirical Literature." *Hungarian Educational Research Journal* 9 (2): 147–64. https://doi.org/10.1556/063.9.2019.1.18.

Kumar, Mukesh, A. J. Singh, and Disha Handa. 2017. "Literature Survey on Educational Dropout Prediction." *International Journal of Education and Management Engineering* 7 (2): 8–19. https://doi.org/10.5815/ijeme.2017.02.02.

Kumar, S Anupama et al. 2011. "Efficiency of Decision Trees in Predicting Student's Academic Performance."

Lippmann, Richard. 1994. "Book Review: "Neural Networks, A Comprehensive Foundation", by Simon Haykin." *International Journal of Neural Systems* 05 (04): 363–64. https://doi.org/10.1142/s0129065794000372.

Martins, Mónica V., Daniel Tolledo, Jorge Machado, Luís M. T. Baptista, and Valentim Realinho. 2021. "Early Prediction of Student's Performance in Higher Education: A Case Study." In, 166–75. Springer International Publishing. https://doi.org/10.1007/978-3-030-72657-7_16.

Namoun, Abdallah, and Abdullah Alshanqiti. 2020. "Predicting Student Performance Using Data Mining and Learning Analytics Techniques: A Systematic Literature Review." *Applied Sciences* 11 (1): 237. https://doi.org/10.3390/app11010237.

Opitz, D. W., and R. F. Maclin. n.d. "An Empirical Evaluation of Bagging and Boosting for Artificial Neural Networks." *Proceedings of International Conference on Neural Networks (ICNN'97)*. https://doi.org/10.1109/icnn.1997.613999.

Plant, Nathaniel G., and K. Todd Holland. 2011. "Prediction and Assimilation of Surf-Zone Processes Using a Bayesian Network." *Coastal Engineering* 58 (1): 119–30. https://doi.org/10.1016/j.coastaleng.2010.09.003.

Quinlan, J. R. 1986. "Induction of Decision Trees." *Machine Learning* 1 (1): 81–106. https://doi.org/10.1007/bf00116251.

Realinho, Valentim, Jorge Machado, Luís Baptista, and Mónica V. Martins. 2022. "Predicting Student Dropout and Academic Success." *Data* 7 (11): 146. https://doi.org/10.3390/data7110146.

Saa, Amjad Abu, Mostafa Al-Emran, and Khaled Shaalan. 2019. "Mining Student Information System Records to Predict Students' Academic Performance." In, 229–39. Springer International Publishing. https://doi.org/10.1007/978-3-030-14118-9_23.

Saarela, Mirka, and Susanne Jauhiainen. 2021. "Comparison of Feature Importance Measures as Explanations for Classification Models." *SN Applied Sciences* 3 (2). https://doi.org/10.1007/s42452-021-04148-9.

Spelmen, Vimalraj S, and R Porkodi. 2018. "A Review on Handling Imbalanced Data." *2018 International Conference on Current Trends Towards Converging Technologies (ICCTCT)*, March. https://doi.org/10.1109/icctct.2018.8551020.

Wang, Shuo, and Xin Yao. 2009. "Diversity Analysis on Imbalanced Data Sets by Using

Ensemble Models." *2009 IEEE Symposium on Computational Intelligence and Data Mining*, March. https://doi.org/10.1109/cidm.2009.4938667.

Wang, Wan, Xinglu Liu, and Wai Kin Victor Chan. 2020. "Imbalanced Classification Problem Using Data-Driven and Random Forest Method." *Proceedings of the 3rd International Conference on Data Science and Information Technology*, July. https://doi.org/10.1145/3414274.3414278.

Wong, Bo K, Thomas A Bodnovich, and Yakup Selvi. 1997. "Neural Network Applications in Business: A Review and Analysis of the Literature (1988–1995)." *Decision Support Systems* 19 (4): 301–20. https://doi.org/10.1016/s0167-9236(96)00070-x.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2009. "Exploratory Undersampling for Class-Imbalance Learning." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39 (2): 539–50. https://doi.org/10.1109/tsmcb.2008.2007853.

Characteristic	**N = 4,424**
Marital.status	
1	3,919 (89%)
2	379 (8.6%)
3	4 (<0.1%)
4	91 (2.1%)
5	25 (0.6%)
6	6 (0.1%)
Application.mode	8 (1, 12)
Application.order	
0	1 (<0.1%)
1	3,026 (68%)
2	547 (12%)
3	309 (7.0%)
4	249 (5.6%)
5	154 (3.5%)
6	137 (3.1%)
9	1 (<0.1%)
Course	10 (6, 13)
Daytime.evening.attendance	3,941 (89%)
Previous.qualification	1 (1, 1)
Nacionality	1 (1, 1)
Mother.s.qualification	13 (2, 22)
Father.s.qualification	14 (3, 27)
Mother.s.occupation	6 (5, 10)
Father.s.occupation	8 (5, 10)
Displaced	2,426 (55%)
Educational.special.needs	51 (1.2%)
Debtor	503 (11%)
Tuition.fees.up.to.date	3,896 (88%)
Gender	1,556 (35%)
Scholarship.holder	1,099 (25%)
Age.at.enrollment	20 (19, 25)
International	110 (2.5%)
Curricular.units.1st.sem..credited.	0 (0, 0)
Curricular.units.1st.sem..enrolled.	6 (5, 7)
Curricular.units.1st.sem..evaluations.	8 (6, 10)
Curricular.units.1st.sem..approved.	5 (3, 6)
Curricular.units.1st.sem..grade.	12.3 (11.0, 13.4)
Curricular.units.1st.sem..without.evaluations.	0 (0, 0)
Curricular.units.2nd.sem..credited.	0 (0, 0)
Curricular.units.2nd.sem..enrolled.	6 (5, 7)
Curricular.units.2nd.sem..evaluations.	8 (6, 10)
Curricular.units.2nd.sem..approved.	5 (2, 6)
Curricular.units.2nd.sem..grade.	12.2 (10.8, 13.3)
Curricular.units.2nd.sem..without.evaluations.	0 (0, 0)
Unemployment.rate	11.10 (9.40, 13.90)
Inflation.rate	
-0.8	533 (12%)
-0.3	390 (8.8%)
0.3	362 (8.2%)
0.5	445 (10%)
0.6	414 (9.4%)
1.4	893 (20%)
2.6	571 (13%)
2.8	397 (9.0%)
3.7	419 (9.5%)
GDP	0.32 (-1.70, 1.79)

Appendix- Code

```
---
title: "Students Dropout Prediction Model in Higher Education Institutions Using Machine Learning
Algorithms"
author: "Luke Philip Ogweno, Hong Shi, and Divya Sharma"
format:
  pdf:
    include-in-header:
      - file: pdf-engine-opt=-shell-escape
      - file: packages.tex
      - file: usepackage[margin=0.5in]{geometry}
      - file: pdflatex()
      - macros.tex
editor: visual
bibliography: references.bib
---

## Towards a Students' Dropout Prediction Model in Higher Education Institutions Using Machine Learning
Algorithms

\newpage

# Abstract

\newpage

```{r, message = FALSE, warning = FALSE}
knitr::opts_chunk$set(echo = FALSE)

library(knitr)
library(tidyverse)
library(kableExtra)

```

Attributes used grouped by class of attribute is shown in Table 1 below

```{r, message = FALSE, warning = FALSE}
my_table <- head(cbind(
 "Demographic data" = c("Marital status",
                        "Nationality",
                        "Displaced",
                        "Gender",
                        "Age at enrollment",
                        "International"),
 "Socioeconomic data" = c("Mother's qualification",
                          "Father's qualification",
                          "Mother's occupation",
                          "Father's occupation",
                          "Educational special needs",
                          "Debtor",
                          "Tuition fees up to date",
                          "Scholarship holder"),
 "Macroeconomic data" = c("Unemployment rate",
                          "Inflation rate",
                          "GDP"),
 "Academic data at enrollment" = c("Application mode",
```
```

```
                                    "Application order",
                                    "Course",
                                    "Daytime/evening attendance",
                                    "Previous qualification"),
 "Academic data at the end of 1st semester" = c("Curricular units 1st sem (credited)",
                                                "Curricular units 1st sem (enrolled)",
                                                "Curricular units 1st sem (evaluations)",
                                                "Curricular units 1st sem (approved)",
                                                "Curricular units 1st sem (grade)",
                                                "Curricular units 1st sem (without evaluations)"),
 "Academic data at the end of 2nd semester" = c("Curricular units 2nd sem (credited)",
                                                "Curricular units 2nd sem (enrolled)",
                                                "Curricular units 2nd sem (approved)",
                                                "Curricular units 2nd sem (grade)",
                                                "Curricular units 2nd sem (without evaluations)"),
   "Target" = c("Target")
),
n = 30)

kable(my_table, "latex", booktabs = TRUE,
     longtable = TRUE,
     caption = "Attributes used grouped by class of attribute",
     row.names = FALSE, align = "c") %>%
 kable_styling(latex_options = "hold_position",
               full_width = FALSE,
               font_size = 6, position = "left") %>%
 column_spec(1, width = "2cm") %>%
 column_spec(2:8, width = "1.5cm") %>%
 column_spec(9:11, width = "2.5cm") %>%
 column_spec(12:17, width = "2cm") %>%
 column_spec(18, width = "1.5cm") %>%
 column_spec(19:23, width = "2cm") %>%
 column_spec(24, width = "1.5cm") %>%
 column_spec(25:29, width = "2cm")%>%
 knitr::knit_print(options = list(font.size = 6, rotate = -90))
```

\newpage

2 Literature review

\newpage

3 Methods

3.1 Libraries used

```{r, message = FALSE, warning = FALSE}

knitr::opts_chunk$set(echo=TRUE)
rm(list = ls())
# Import Required Libraries
library(readr)
library(dplyr)
library(tidyverse)
```

```
library(tidyr)
library(caret)
library(randomForest)
library(xgboost)
library(gbm)
library(neuralnet)
library(kernlab)
library(e1071)
#library(nnet)
library(naivebayes)
library(glmnet)
library(rpart)
library(rpart.plot)
library(class)
library(IPEDS)
library(ppsr)
library(ggplot2)
library(gtsummary)
library(gridExtra)
library(kableExtra)
library(gt)
library(pROC)
library(magrittr)
library(cowplot)
library(ranger)
library(lightgbm)
library(forcats)
library(parallel)
library(parallelML)
```

3.2 EDA and Feature Engineer

Overview of the dataset

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Load data
data <- read.csv("dataset.csv", header = TRUE, stringsAsFactors = TRUE)

# Set chunk options to fit the figure into an A4 page
knitr::opts_chunk$set(fig.width = 7, fig.height = 8.5)

data %>%
  tbl_summary() %>%
  as_kable_extra() %>%
  kable_styling(latex_options = c("hold_position"), font_size = 10)
```

3.2.1 Correlation matrix

```{r, message = FALSE, warning = FALSE, echo=FALSE, fig.width=8.27, fig.height=11.69}
ppsr::visualize_pps(df = data,
                    color_value_high = 'red',
                    color_value_low = 'yellow',
```

```
                   color_text = 'black',
                   verbose = FALSE) +
  theme_classic() +
  theme(plot.background = element_rect(fill = "lightgrey")) +
  theme(title = element_text(size = 15)) +
  labs(title = 'Checking for missing values',
       subtitle = 'Determining the correlation amongst the variables',
       caption = 'Correlation matrix',
       x = element_blank()) +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5, hjust=1, size = 8),
        axis.text.y = element_text(size = 8))


```

3.2.1 Data distribution and Basic statistics information

```{r, message = FALSE, warning = FALSE, fig.width=8.27, fig.height=11.69, echo=FALSE}

# create a new graphics device to display the plot

# Check class distribution
table(data$Target)

dev.new()
# Create bar plot
ggplot(data, aes(x = Target)) +
  geom_bar() +
  xlab("Target") +
  ylab("Count") +
  ggtitle("Basic statistics information about demographic data.") +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 1. Bar plot showing the distribution of target variable in the demographic
data.\nMean: 0.2185 | Median: 0.0 | Dispersion: 0.4132 | Min: 0 | Max: 1") +
  theme(plot.caption.position = "plot", plot.caption = element_text(size = 10)) +
  theme(plot.title = element_text(size = 15)) +
  theme(plot.background = element_rect(fill = "lightgrey")) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  theme(axis.line = element_line(colour = "black")) +
  theme(axis.text.x = element_text(size = 12), axis.text.y = element_text(size = 12)) +
  theme(axis.title.x = element_text(size = 12), axis.title.y = element_text(size = 12)) +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  theme(panel.border = element_blank()) +
  theme(panel.background = element_blank()) +
  theme(axis.ticks.length = unit(0.3, "cm")) +
  theme(axis.ticks = element_line(colour = "black")) +
  theme(panel.spacing = unit(1, "lines")) +
  theme(panel.grid.major.x = element_line(colour = "white"))
dev.off()

```

```{r, message = FALSE, warning = FALSE, echo=FALSE}
ggplot(data, aes(x = `Marital.status`, fill = Target)) +
  geom_bar() +
  labs(x = "Marital status", y = "Count", title = "Marital status by target")
```

\newpage

```{r, message = FALSE, warning = FALSE, fig.width=8.27, fig.height=11.69, echo=FALSE}
data <- subset(data, Target %in% c("Graduate", "Dropout"))
data$Target <- ifelse(data$Target == "Dropout", 0, 1)


ppsr::visualize_pps(df = data,
                    color_value_high = 'red',
                    color_value_low = 'yellow',
                    color_text = 'black',
                    verbose = FALSE) +
  theme_classic() +
  theme(plot.background = element_rect(fill = "lightgrey")) +
  theme(title = element_text(size = 15)) +
  labs(title = 'Correlation',
       subtitle = 'Determining the correlation amongst the variables',
       caption = 'Correlation matrix',
       x = element_blank()) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1, size = 8),
        axis.text.y = element_text(size = 8))
```

\newpage

3.2.2 Descriptive Analysis

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Convert variables to factors and remove rows with missing values
data <- data %>%
  mutate(Target = as.factor(Target)) %>%
  mutate(Marital.status = as.factor(Marital.status)) %>%
  mutate(Daytime.evening.attendance = as.factor(Daytime.evening.attendance)) %>%
  mutate(Displaced = as.factor(Displaced)) %>%
  mutate(Educational.special.needs = as.factor(Educational.special.needs)) %>%
  mutate(Debtor = as.factor(Debtor)) %>%
  mutate(Tuition.fees.up.to.date = as.factor(Tuition.fees.up.to.date)) %>%
  mutate(Gender = as.factor(Gender)) %>%
  mutate(Scholarship.holder = as.factor(Scholarship.holder)) %>%
  mutate(International = as.factor(International)) %>%
  na.omit()

# Plot distribution of each variable
ggplot(data, aes(x = Target)) +
  geom_bar() +
  ggtitle("Target Variable Distribution") +
  xlab("Target") +
  ylab("Count")

ggplot(data, aes(x = Marital.status)) +
  geom_bar() +
  ggtitle("Marital Status Distribution") +
  xlab("Marital Status") +
  ylab("Count")

ggplot(data, aes(x = Daytime.evening.attendance)) +
```

````
  geom_bar() +
  ggtitle("Daytime/Evening Attendance Distribution") +
  xlab("Daytime/Evening Attendance") +
  ylab("Count")

ggplot(data, aes(x = Displaced)) +
  geom_bar() +
  ggtitle("Displaced Distribution") +
  xlab("Displaced") +
  ylab("Count")

ggplot(data, aes(x = Educational.special.needs)) +
  geom_bar() +
  ggtitle("Educational Special Needs Distribution") +
  xlab("Educational Special Needs") +
  ylab("Count")

ggplot(data, aes(x = Debtor)) +
  geom_bar() +
  ggtitle("Debtor Distribution") +
  xlab("Debtor") +
  ylab("Count")

ggplot(data, aes(x = Tuition.fees.up.to.date)) +
  geom_bar() +
  ggtitle("Tuition Fees Up-to-date Distribution") +
  xlab("Tuition Fees Up-to-date") +
  ylab("Count")

ggplot(data, aes(x = Gender)) +
  geom_bar() +
  ggtitle("Gender Distribution") +
  xlab("Gender") +
  ylab("Count")

ggplot(data, aes(x = Scholarship.holder)) +
  geom_bar() +
  ggtitle("Scholarship Holder Distribution") +
  xlab("Scholarship Holder") +
  ylab("Count")

ggplot(data, aes(x = International)) +
  geom_bar() +
  ggtitle("International Distribution") +
  xlab("International") +
  ylab("Count")


```

\newpage

### Convert data into factors

```{r, message = FALSE, warning = FALSE, echo=FALSE}
data$Educational.special.needs <- as.factor(data$Educational.special.needs)
data$Debtor <- as.factor(data$Debtor)
````

```
data$Tuition.fees.up.to.date <- as.factor(data$Tuition.fees.up.to.date)
data$Gender <- as.factor(data$Gender)
data$Scholarship.holder <- as.factor(data$Scholarship.holder)
data$International <- as.factor(data$International)
```

3.2.3 Subset dataset for statistical summaries

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Subset the data
basic_demo_data <- data %>%
  select(Marital.status,
         Nacionality,
         Displaced,
         Gender,
         Age.at.enrollment,
         International)

basic_soc_data <- data %>%
  select(Mother.s.qualification,
         Father.s.qualification,
         Mother.s.occupation,
         Father.s.occupation,
         Educational.special.needs,
         Debtor, Tuition.fees.up.to.date,
         Scholarship.holder)

basic_macro_data <- data %>%
  select(Unemployment.rate, Inflation.rate, GDP)

basic_academic_enrollment_data <- data %>%
  select(Application.mode,
         Application.order,
         Course,
         Daytime.evening.attendance,
         Previous.qualification)

basic_academic_end_semester_data <- data %>%
  select(Curricular.units.1st.sem..credited.,
         Curricular.units.1st.sem..enrolled.,
         Curricular.units.1st.sem..evaluations.,
         Curricular.units.1st.sem..approved.,
         Curricular.units.1st.sem..grade.,
         Curricular.units.1st.sem..without.evaluations.)

basic_academic_end_second_semester_data <- data %>%
  select(Curricular.units.2nd.sem..credited.,
         Curricular.units.2nd.sem..enrolled.,
         Curricular.units.2nd.sem..evaluations.,
         Curricular.units.2nd.sem..approved.,
         Curricular.units.2nd.sem..grade.,
         Curricular.units.2nd.sem..without.evaluations.)

basic_target_data <- data %>%
  select(Target)
```

\newpage

````
```{r, message = FALSE, warning = FALSE, echo=FALSE}
print(ggplot(basic_demo_data, aes(x = Marital.status)) +
  geom_bar() +
  xlab("Marital status") +
  ylab("Count") +
  ggtitle("Table 2. Basic statistics information about demographic data") +
  theme(plot.caption = element_text(hjust = 0)) +
  labs(caption = "Figure 1. Distribution plot showing marital status in the demographic data.") +
  theme(plot.caption.position = "plot",
        plot.caption = element_text(size = 10)) +
  theme(plot.title = element_text(size = 15)) +
  theme(plot.background = element_rect(fill = "lightgrey")) +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank()) +
  theme(axis.line = element_line(colour = "black")) +
  theme(axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12)) +
  theme(axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12)) +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  theme(panel.border = element_blank()) +
  theme(panel.background = element_blank()) +
  theme(axis.ticks.length = unit(0.3, "cm")) +
  theme(axis.ticks = element_line(colour = "black")) +
  theme(panel.spacing = unit(1, "lines")) +
  theme(panel.grid.major.x = element_line(colour = "white")))

```
````

\newpage

Table 2. Basic statistics information about demographic data.

````
```{r setup, include=FALSE, echo=FALSE}
#knitr::opts_chunk$set(echo = FALSE)

ggplot(basic_demo_data, aes(x = Marital.status)) +
  geom_bar() +
  xlab("Marital status") +
  ylab("Count") +
  ggtitle("Table 2. Basic statistics information about demographic data") +
  labs(caption = "Figure 1. Distribution plot showing marital status in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
        plot.caption.position = "panel", # update the value of plot.caption.position
        plot.title = element_text(size = 15),
        plot.background = element_rect(fill = "lightgrey"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        plot.margin = unit(c(1,1,1,1), "cm"),
        panel.border = element_blank(),
````
````

```
      panel.background = element_blank(),
      axis.ticks.length = unit(0.3, "cm"),
      axis.ticks = element_line(colour = "black"),
      panel.spacing = unit(1, "lines"),
      panel.grid.major.x = element_line(colour = "white"))

# Select variables of interest
basic_demo_data <- data %>%
  select(Marital.status, Nacionality, Displaced, Gender, Age.at.enrollment, International)

# Identify non-factor variables
non_factor_vars <- names(basic_demo_data)[!sapply(basic_demo_data, is.factor)]

# Calculate summary statistics for each non-factor variable
summary_stats <- basic_demo_data %>%
  summarise(across(all_of(non_factor_vars), list(mean = mean, median = median, sd = sd, min = min, max
= max)))

# Add "N/A" for factor variables
factor_vars <- setdiff(names(basic_demo_data), non_factor_vars)
for(var in factor_vars){
  summary_stats[var, "mean"] <- "N/A"
  summary_stats[var, "median"] <- "N/A"
  summary_stats[var, "sd"] <- "N/A"
}

# Print table of summary statistics
kable(summary_stats, caption = "Table 1. Basic statistics information about demographic data") %>%
  kable_styling(full_width = FALSE)



ggplot(basic_demo_data, aes(x = Nacionality)) +
  geom_bar() +
  xlab("Nacionality") +
  ylab("Count") +
  ggtitle("Table 3. Basic statistics information about demographic data") +
  labs(caption = "Figure 2. Distribution plot showing Nationality status in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
      plot.caption.position = "panel", # update the value of plot.caption.position
      plot.title = element_text(size = 15),
      plot.background = element_rect(fill = "lightgrey"),
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      axis.line = element_line(colour = "black"),
      axis.text.x = element_text(size = 12),
      axis.text.y = element_text(size = 12),
      axis.title.x = element_text(size = 12),
      axis.title.y = element_text(size = 12),
      plot.margin = unit(c(1,1,1,1), "cm"),
      panel.border = element_blank(),
      panel.background = element_blank(),
      axis.ticks.length = unit(0.3, "cm"),
      axis.ticks = element_line(colour = "black"),
      panel.spacing = unit(1, "lines"),
      panel.grid.major.x = element_line(colour = "white"))
```

```
ggplot(basic_demo_data, aes(x = Displaced)) +
  geom_bar() +
  xlab("Displaced") +
  ylab("Count") +
  ggtitle("Table 2. Basic statistics information about demographic data") +
  labs(caption = "Figure 3. Distribution plot showing Displaced in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
        plot.caption.position = "panel", # update the value of plot.caption.position
        plot.title = element_text(size = 15),
        plot.background = element_rect(fill = "lightgrey"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        plot.margin = unit(c(1,1,1,1), "cm"),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks.length = unit(0.3, "cm"),
        axis.ticks = element_line(colour = "black"),
        panel.spacing = unit(1, "lines"),
        panel.grid.major.x = element_line(colour = "white"))

ggplot(basic_demo_data, aes(x = Gender)) +
  geom_bar() +
  xlab("Gender") +
  ylab("Count") +
  ggtitle("Table 2. Basic statistics information about demographic data") +
  labs(caption = "Figure 4. Distribution plot showing Gender in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
        plot.caption.position = "panel", # update the value of plot.caption.position
        plot.title = element_text(size = 15),
        plot.background = element_rect(fill = "lightgrey"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        plot.margin = unit(c(1,1,1,1), "cm"),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks.length = unit(0.3, "cm"),
        axis.ticks = element_line(colour = "black"),
        panel.spacing = unit(1, "lines"),
        panel.grid.major.x = element_line(colour = "white"))


ggplot(basic_demo_data, aes(x = Age.at.enrollment)) +
  geom_bar() +
  xlab("Age at enrollment") +
  ylab("Count") +
```

```
  ggtitle("Table 2. Basic statistics information about demographic data") +
  labs(caption = "Figure 5. Distribution plot showing Age at enrollment in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
        plot.caption.position = "panel", # update the value of plot.caption.position
        plot.title = element_text(size = 15),
        plot.background = element_rect(fill = "lightgrey"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        plot.margin = unit(c(1,1,1,1), "cm"),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks.length = unit(0.3, "cm"),
        axis.ticks = element_line(colour = "black"),
        panel.spacing = unit(1, "lines"),
        panel.grid.major.x = element_line(colour = "white"))


ggplot(basic_demo_data, aes(x = International)) +
  geom_bar() +
  xlab("International") +
  ylab("Count") +
  ggtitle("Table 2. Basic statistics information about demographic data") +
  labs(caption = "Figure 6. Distribution plot showing International in the demographic data.") +
  theme(plot.caption = element_text(hjust = 0),
        plot.caption.position = "panel", # update the value of plot.caption.position
        plot.title = element_text(size = 15),
        plot.background = element_rect(fill = "lightgrey"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"),
        axis.text.x = element_text(size = 12),
        axis.text.y = element_text(size = 12),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        plot.margin = unit(c(1,1,1,1), "cm"),
        panel.border = element_blank(),
        panel.background = element_blank(),
        axis.ticks.length = unit(0.3, "cm"),
        axis.ticks = element_line(colour = "black"),
        panel.spacing = unit(1, "lines"),
        panel.grid.major.x = element_line(colour = "white"))



` ` `



` ` `{r, message = FALSE, warning = FALSE, fig.width=8.27, fig.height=11.69, echo=FALSE}
# Load the data
my_data <- read.csv("dataset.csv")

ggplot(my_data, aes(x = Target)) +
```

~~~
  geom_bar(fill = c("blue", "red", "green")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "",
       x = "Target",
       y = "Count",
       caption = "Figure 2. Distribution of student records among the three categories considered for academic success")


```
~~~

Figure 3 shows the same imbalanced nature of data when grouping the student outcomes by course and Gender.

~~~
```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Define the course key as a named vector
course_key <- c("1" = "Biofuel Production Technologies",
                "2" = "Animation and Multimedia Design",
                "3" = "Social Service (evening attendance)",
                "4" = "Agronomy",
                "5" = "Communication Design",
                "6" = "Veterinary Nursing",
                "7" = "Informatics Engineering",
                "8" = "Equiniculture",
                "9" = "Management",
                "10" = "Social Service",
                "11" = "Tourism",
                "12" = "Nursing",
                "13" = "Oral Hygiene",
                "14" = "Advertising and Marketing Management",
                "15" = "Journalism and Communication",
                "16" = "Basic Education",
                "17" = "Management (evening attendance)")
Gender_key <- c("1" = "Male", "2" = "Female")
# Add a new column with the course character values
my_data$Course_character <- course_key[as.character(my_data$Course)]
my_data$Gender_character <- Gender_key[as.character(my_data$Gender)]
```

```{r, message = FALSE, warning = FALSE, fig.width=8.27, fig.height=11.69, echo=FALSE}

my_data_counts <- my_data %>%
  group_by(Course_character, Target) %>%
  summarise(Count = n(), .groups = "drop")

my_data_counts_gender <- my_data %>%
  group_by(Gender_character, Target) %>%
  summarise(Count = n(), .groups = "drop")

# Create plot for course count in ascending order
plot_course <- ggplot(my_data_counts,
                      aes(x = reorder(Course_character, Count),
                          y = Count, fill = Target)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("blue", "red", "green")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "",
~~~

```
      x = "Course",
      y = "Count",
      caption = "Figure 3. Student outcomes grouped by: (a) Course") +
  coord_flip()

# Create plot for gender count
plot_gender <- ggplot(my_data_counts_gender,
                      aes(
                        x = Gender_character,
                        y = Count,
                        fill = Target)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("blue", "red", "green")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(
    title = "",
      x = "Gender",
      y = "Count",
      caption = "Figure 3. Student outcomes grouped by: (b) Gender")

# Combine plots with a common caption
plot_grid(plot_course,
          plot_gender,
          ncol = 1,
          align = "v",
          axis = "tb",
          rel_heights = c(1, 1),
          labels = c("(a) Course", "(b) Gender")) +
  theme(plot.caption = element_text(hjust = 0.5))


```

3.2.6 Split the dataset into training and testing sets

We split the data into ratio 70% for training and 30% for testing using random split.

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Split data into training and test sets
set.seed(123)
# Load data
dat <- read.csv("dataset.csv", header = TRUE, stringsAsFactors = TRUE)
train_index <- createDataPartition(dat$Target, p = 0.7, list = FALSE, times = 1)
train_data <- dat[train_index, ]
test_data <- dat[-train_index, ]

```

```{r, message = FALSE, warning = FALSE, fig.width=8.27, fig.height=11.69, echo=FALSE}
# Train RF model
set.seed(123)
# Train RF model
rf_model <- ranger(Target ~ ., data = train_data, importance = 'permutation')
```

````r
# Get feature importance
importance <- importance(rf_model)

# Get feature importance
importance <- importance(rf_model)

# Create data frame for feature importance
importance_df <- data.frame(
  feature = names(importance),
  importance = importance
)

# Order the data frame by importance in descending order
importance_df <- importance_df[order(importance_df$importance, decreasing = TRUE),]

# Plot feature importance with a cutoff of top 10 features
ggplot(data = importance_df, aes(x = importance, y = reorder(factor(feature), importance))) +
  geom_col(fill = "green") +
  labs(title = "Feature Importance", x = "Importance", y = "Feature") +
  theme_minimal()


```
````

3.3 Data Analysis

3.3.1 Re-sample training and test data to balance the classes

````r
```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Re-sample training data to balance the classes
train_data_balanced <- train_data %>%
  group_by(Target) %>%
  sample_n(size = max(table(train_data$Target)), replace = TRUE) %>%
  ungroup()

# Check class distribution in balanced training data
table(train_data_balanced$Target)


# Re-sample training data to balance the classes
test_data_balanced <- test_data %>%
  group_by(Target) %>%
  sample_n(size = max(table(test_data$Target)), replace = TRUE) %>%
  ungroup()

# Check class distribution in balanced training data
table(test_data_balanced$Target)


# Use the balanced training data for model training
# and the test data for model evaluation
```
````

3.4 Define the model

We have implemented the following models" the Random Forest models (RF),
the neural network model (NN), Support Vector Machine (SVM), Naive
````

Bayes(NB), and decision tree (DT) algorithms.

3.4.1 Random Forest, Decision Tree, Naive Bayes and SVM models

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Define the formula for the model
formula <- Target ~ Curricular.units.2nd.sem..approved.  + Tuition.fees.up.to.date + 
Curricular.units.1st.sem..approved. + Curricular.units.2nd.sem..grade.+ Course + 
Curricular.units.1st.sem..enrolled. + Scholarship.holder + Curricular.units.2nd.sem..evaluations. + 
Inflation.rate + Curricular.units.1st.sem..evaluations.

```

3.4.2 Define the model training and testing functions

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Define the model training and testing functions
train_model <- function(model_name, formula, train_data_balanced) {
  if (model_name == "SVM") {
    model <- svm(formula, train_data_balanced, kernel = "linear", cost = 10, gamma = 0.1)
  } else if (model_name == "DT") {
    model <- rpart(formula, train_data_balanced, method = "class")
  } else if (model_name == "ANN") {
    model <- nnet(formula, train_data_balanced, size = 5, decay = 1e-5, maxit = 1000)
  } else if (model_name == "LR") {
    model <- glm(formula, train_data_balanced, family = "binomial")
  } else if (model_name == "NB") {
    model <- naiveBayes(formula,train_data_balanced)
  } else if (model_name == "RF") {
    model <- randomForest(formula, train_data_balanced, ntree = 500, importance = TRUE)
  } else if (model_name == "BAG") {
    model <- bagging(formula, train_data_balanced, nbagg = 25)
  } else if (model_name == "BOOST") {
    model <- boosting(formula, train_data_balanced, mfinal = 500)
  }
  return(model)
}

test_model <- function(model, test_data_balanced) {
  predictions <- predict(model, newdata = test_data_balanced, type = "class")
  accuracy <- confusionMatrix(predictions, test_data_balanced$Target)$overall["Accuracy"]
  return(accuracy)
}

```

3.4.3 Train and test the models

```{r, message = FALSE, warning = FALSE, echo=FALSE}
# Train and test the models
svm_model <- train_model("SVM", formula, train_data_balanced)

if (is.null(test_data_balanced) || !all(colnames(test_data_balanced) %in% colnames(train_data_balanced))) 
{
  stop("Test data is not compatible with model")
}
print(table(svm_model$fitted, train_data_balanced$Target))
```

```
#print(table(svm_model, train_data_balanced$Target))

svm_pred <- predict(svm_model, newdata = test_data_balanced, type = "class")
print(table(svm_pred, test_data_balanced$Target))


svm_acc <- test_model(svm_model, test_data_balanced)

tree_model <- train_model("DT", formula, train_data_balanced)
tree_pred <- predict(tree_model, newdata = test_data_balanced, type = "class")
tree_acc <- test_model(tree_model, test_data_balanced)
# Print confusion matrix
conf_mat <- confusionMatrix(tree_pred, test_data_balanced$Target)
print(conf_mat$table)

rf_model <- train_model("RF", formula, train_data_balanced)
print(table(predict(rf_model, train_data_balanced, type = "class"), train_data_balanced$Target))

rf_pred <- predict(rf_model, newdata = test_data_balanced, type = "class")
print(table(rf_pred, test_data_balanced$Target))

rf_acc <- test_model(rf_model, test_data_balanced)

#ada_model <- gbm(Target ~ ., data = train_data_balanced, distribution = "multinomial", n.trees = 500,
interaction.depth = 3)
#ada_pred <- predict(ada_model, newdata = test_data_balanced, type = "class")
#ada_acc <- test_model(ada_pred, test_data_balanced$Target)

nb_model <- train_model("NB", formula, train_data_balanced)
nb_train <- predict(nb_model, newdata = train_data_balanced, type = "class")
print(table(nb_train, train_data_balanced$Target))

nb_pred <- predict(nb_model, newdata = test_data_balanced, type = "class")
print(table(nb_pred, test_data_balanced$Target))

nb_acc <- test_model(nb_model, test_data_balanced)

test_model <- function(model, test_data_balanced) {
  predictions <- predict(model, test_data_balanced, type = "class")
  accuracy <- confusionMatrix(predictions, test_data_balanced$Target,
mode="everything")$overall["Accuracy"]
  return(accuracy)
}


```

3.4.4 Plot confusion matrix for each model

```{r, message = FALSE, warning = FALSE, fig.width=6.69, fig.height=6.69, echo=FALSE}
#library(ggplot2)
library(ggpubr)
library(ggrepel)

# Get the confusion matrices for each model
svm_cm <- confusionMatrix(svm_pred, test_data_balanced$Target)
tree_cm <- confusionMatrix(tree_pred, test_data_balanced$Target)
```

```
rf_cm <- confusionMatrix(rf_pred, test_data_balanced$Target)
nb_cm <- confusionMatrix(nb_pred, test_data_balanced$Target)

# Create data frames for plotting
svm_df <- data.frame(
  Actual = factor(rep(colnames(svm_cm$table), each = ncol(svm_cm$table))),
  Predicted = factor(colnames(svm_cm$table), levels = colnames(svm_cm$table)),
  Frequency = as.vector(svm_cm$table)
)

tree_df <- data.frame(
  Actual = factor(rep(colnames(tree_cm$table), each = ncol(tree_cm$table))),
  Predicted = factor(colnames(tree_cm$table), levels = colnames(tree_cm$table)),
  Frequency = as.vector(tree_cm$table)
)

rf_df <- data.frame(
  Actual = factor(rep(colnames(rf_cm$table), each = ncol(rf_cm$table))),
  Predicted = factor(colnames(rf_cm$table), levels = colnames(rf_cm$table)),
  Frequency = as.vector(rf_cm$table)
)

nb_df <- data.frame(
  Actual = factor(rep(colnames(nb_cm$table), each = ncol(nb_cm$table))),
  Predicted = factor(colnames(nb_cm$table), levels = colnames(nb_cm$table)),
  Frequency = as.vector(nb_cm$table)
)

# Plot the confusion matrices
svm_plot <- ggplot(svm_df, aes(x = Predicted, y = Actual, fill = Frequency)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "SVM Confusion Matrix", x = "Predicted", y = "Actual") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  geom_text_repel(aes(label = sprintf("%.2f", Frequency)), size = 3)

tree_plot <- ggplot(tree_df, aes(x = Predicted, y = Actual, fill = Frequency)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Decision Tree Confusion Matrix", x = "Predicted", y = "Actual") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  geom_text_repel(aes(label = sprintf("%.2f", Frequency)), size = 3)

rf_plot <- ggplot(rf_df, aes(x = Predicted, y = Actual, fill = Frequency)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Random Forest Confusion Matrix", x = "Predicted", y = "Actual") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  geom_text_repel(aes(label = sprintf("%.2f", Frequency)), size = 3)

nb_plot <- ggplot(nb_df, aes(x = Predicted, y = Actual, fill = Frequency)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "blue") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Naive Bayes Confusion Matrix", x = "Predicted", y = "Actual") +
 geom_text_repel(aes(label = sprintf("%.2f", Frequency)), size = 3)
```

````
# Arrange the plots in a grid
plot_grid(svm_plot, tree_plot, ncol = 2)
plot_grid(rf_plot, nb_plot, ncol = 2)

```
````

Create a data frame of the accuracies

````
```{r, message = FALSE, warning = FALSE, echo=FALSE}
accuracies <- data.frame(Model = c("SVM",
                                   "Decision Tree",
                                   "Random Forest",
                                   "Naive Bayes"),
Accuracy = c(svm_acc, tree_acc, rf_acc, nb_acc))
```
````

3.4.5 Plot the accuracies

````
```{r, message = FALSE, warning = FALSE, fig.width=12.27, fig.height=6.69, echo=FALSE}
# Create ggplot object
accuracy_plot <- ggplot(accuracies,
                        aes(x = Model,
                            y = Accuracy,
                            fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "Model Accuracies",
       x = "Model",
       y = "Accuracy") +
  theme(plot.title = element_text(size = 18, face = "bold"),
        axis.title.x = element_text(size = 20),
        axis.title.y = element_text(size = 14),
        legend.title = element_blank(),
        legend.position = "top") +
  geom_text(aes(label = paste0( round(Accuracy, 2)),
                color = Model),
            vjust = -0.5,
            size = 4, fontface = "bold")

# Print the plot
accuracy_plot
```
````
````