Choose Your Own Capstone - Predicting Failure in Education

HarvardX PH125.9x Data Science: Capstone

Sara Darland

November 22, 2024

Contents

Introduction

Big data in education is a growing field, as seen by the "Big Data and Education" class offering on the EdX platform (Baker, 2023.) Teachers can use data to identify which students struggle and and create personalized education plans to enhance their learning. Administrators and legislators can use data to identify predictors of failure or success and create initiatives or policy to broadly improve student outcomes. Inspired by my personal experience in the classroom in the Tigard-Tualatin School District and my desire to improve educational outcomes, the goal of this project is to determine the machine learning technique that most accurately identifies students experiencing educational failure. When accurately identified, failing students can receive early intervention in an effort to improve their education outcome. This project explores a variety of regression and classification models and identifies the best model through evaluation of Root Mean Squared Error (RMSE) or Accuracy when the model is run on previously unseen data. This project is required coursework for the HarvardX PH125.9x Data Science: Capstone course.

Dataset

This project uses the Student Performance Factors dataset from the Kaggle website posted by user lainguyn123, who maintains the Kaggle user site "Practice Data Analysis With Me". The dataset is synthetic; it was generated for educational purposes and not sourced from real world institutions (lainguyn123). The dataset consists of student's exam scores and various factors that may affect student performance.

The raw dataset is imported from Github after previously being downloaded from Kaggle. The dataset is checked for blanks, which are replaced with NAs. There are 0 NAs present across 0 rows. The NAs occur in the following variables: NA, NA, NA. These variables are removed in order to keep as many observations in the dataset as possible. The initial view of the dataset shows there are many character variables. These variables are converted to factors in the data cleaning process. Additionally, the levels for each factor are reordered for consistency and to make intuitive sense (low = 1, medium = 2, high = 3). Finally, the variable "Pass" is added - this represents whether a student receives a passing or failing grade and is based on "Exam_Score".

```r
###########################
# Data cleaning
###########################
data <- replace(data, data == '', NA) # replace blanks with NA
any(is.na(data)) # check for NAs
sum(is.na(data)) # count NAs
sum(!complete.cases(data)) # count rows with NAs
colnames(data)[colSums(is.na(data))>0] #identify columns containing NAs

# preprocessing character data into factors, reordering levels (low = 1, high = 3)
data_clean <- data %>%
  mutate(Parental_Involvement = factor(Parental_Involvement, levels=c("Low", "Medium","High")),
         Access_to_Resources = factor(Access_to_Resources, levels=c("Low", "Medium","High")),
         Extracurricular_Activities = factor(Extracurricular_Activities),
         Motivation_Level = factor(Motivation_Level, levels=c("Low", "Medium","High")),
         Internet_Access = factor(Internet_Access),
         Family_Income = factor(Family_Income, levels=c("Low", "Medium","High")),
         School_Type = factor(School_Type),
         Peer_Influence = factor(Peer_Influence),
         Learning_Disabilities = factor(Learning_Disabilities),
         Gender = factor(Gender)) %>%
  select(-one_of(c("Teacher_Quality",
                   "Parental_Education_Level",
                    "Distance_from_Home")))%>% # exclude predictors with NAs
  mutate(Pass = case_when( # add "Pass" outcome
    Exam_Score >= 65 ~"1",
    Exam_Score < 65 ~ "0"))%>%
  mutate(Pass = factor(Pass))
```

The clean dataset consists of 6607 rows or observations, with the following 18 columns (lainguyn123):

- Hours_studied – number of hours student studied per week. It is an integer.
- Attendance – percentage of classes student attended. It is an integer.
- Parental_Involvement – level of parental involvement in student's education. It is a factor.
- Access_to_Resources – student's availability of educational resources. It is a factor.
- Extracurricular_Activities – student's participation in extracurriculars. It is a factor.
- Sleep_Hours – student's average number of hours of sleep per night. It is an integer.
- Previous_Scores – student's previous exam scores. It is an integer.
- Motivation_Level – student's level of motivation. It is a factor.
- Internet_Access – availability of internet to a student. It is a factor.
- Tutoring_Sessions – number of tutoring sessions a student attends per month. It is an integer.
- Family_Income – student's family income level. It is a factor.
- School_Type – type of school student's attends. It is a factor.
- Peer_Influence – influence of student's peers. It is a factor.
- Physical_Activity – student's average number of physically activity hours per week. It is an integer.
- Learning_Disabilities – presence of a learning disability. It is a factor.
- Gender – student's gender. It is a factor.
- Exam_Score – student's final exam score. It is an integer.
- Pass – pass/fail based on Exam_Score. It is a factor.

Data Exploration

The clean dataset is observed, it is in tidy format and each row corresponds to a unique student. For the sake of space only the first 5 rows and columns are shown below:

Table 1: First 5 rows and columns of clean dataset

Hours_Studied	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_Activities
23	84	Low	High	No
19	64	Low	Medium	No
24	98	Medium	Medium	Yes
29	89	Low	Medium	Yes
19	92	Medium	Medium	Yes

Outcomes

The variables that measure student performance are "Exam_Score" and "Pass". These variables are the outcomes that will be predicted; "Exam_Score" is a continuous integer and will be predicted using regression models and "Pass" is a categorical factor and will be predicted using classification models. Student's exam scores range from a minimum of 55 to a maximum of 101 with a mean of 67.236. The higher the score the better a student performed. The distribution of exam scores is not normal - it has a long tail on the right side with more students receiving very high scores than very low scores. Model development assumes a normal distribution which may result in poor model performance when predicting high scores as there is little data to use when training. However the project goal is to accurately predicting low scores so this is not a major concern.

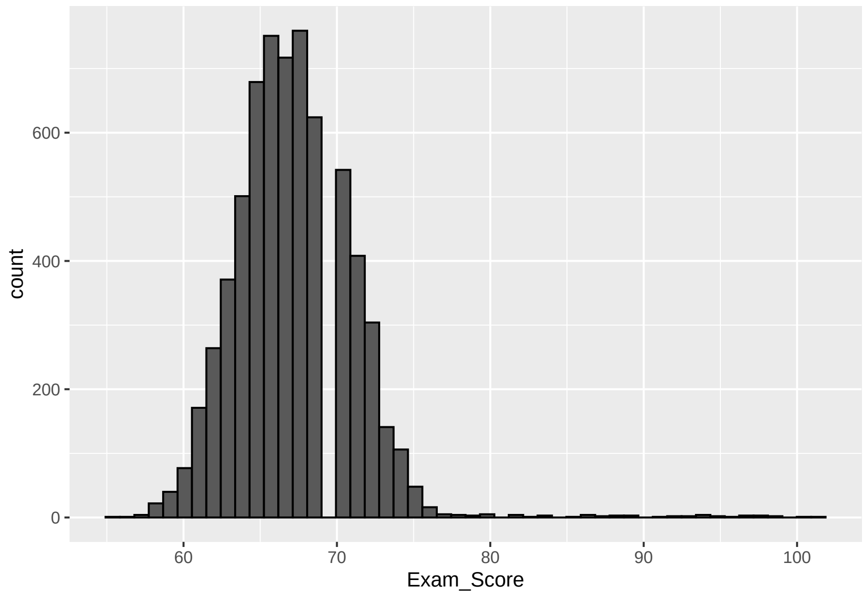

Figure 1: Exam Score distribution

The variable "Pass" represents whether a student passed or failed the exam; an exam score of 65 or greater indicates a student passed (Pass = 1) while an exam score of less than 65 indicates the student failed (Pass =0).

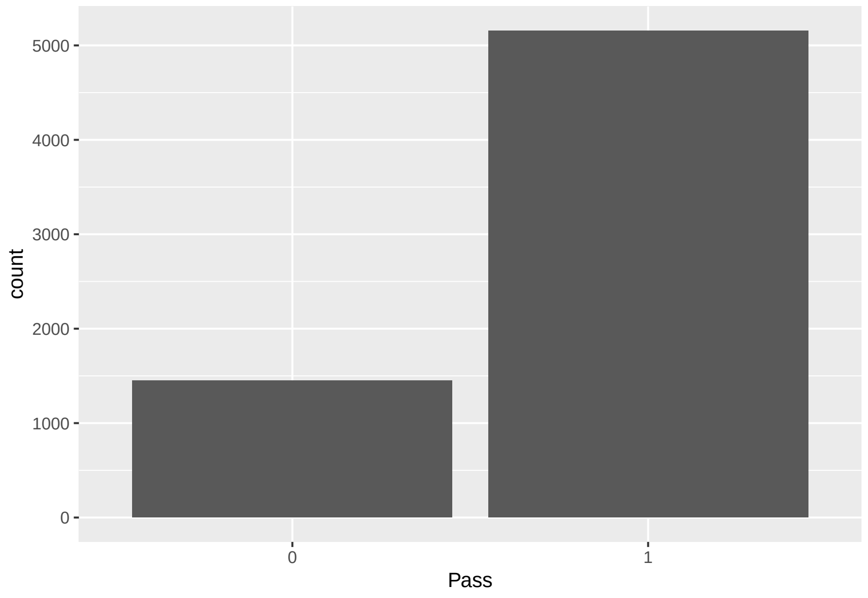

Figure 2: Pass distribution

The table below shows the proportion of students in the dataset that pass and fail.

Table 2: Proportion of Pass/Fail

Pass	proportion
0	0.22
1	0.78

Predictors

The 16 other variables are features used to predict the outcome measuring student success. These features can be split into three groups: student behavior and attitude, home environment, and school environment (lainguyn123).

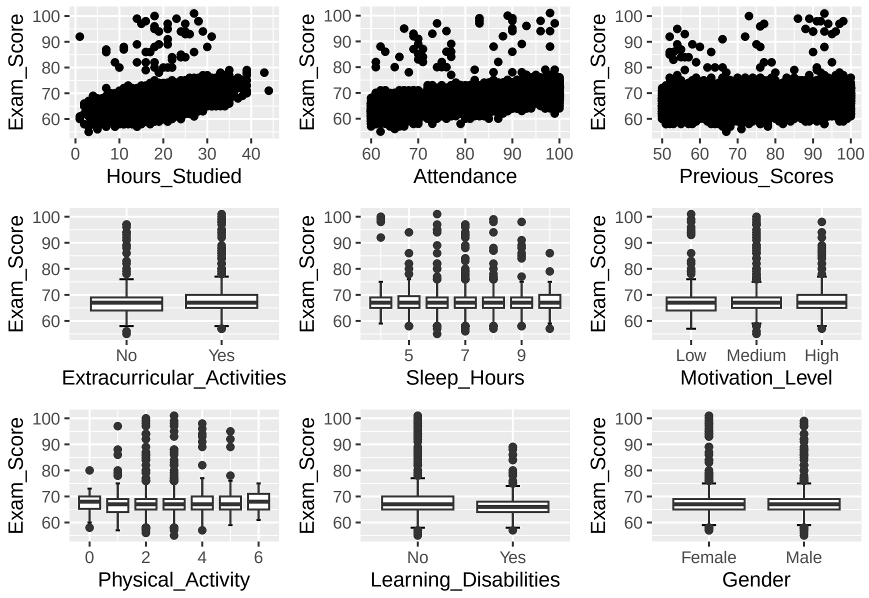

Figure 3: Student Behaviors and Attitudes

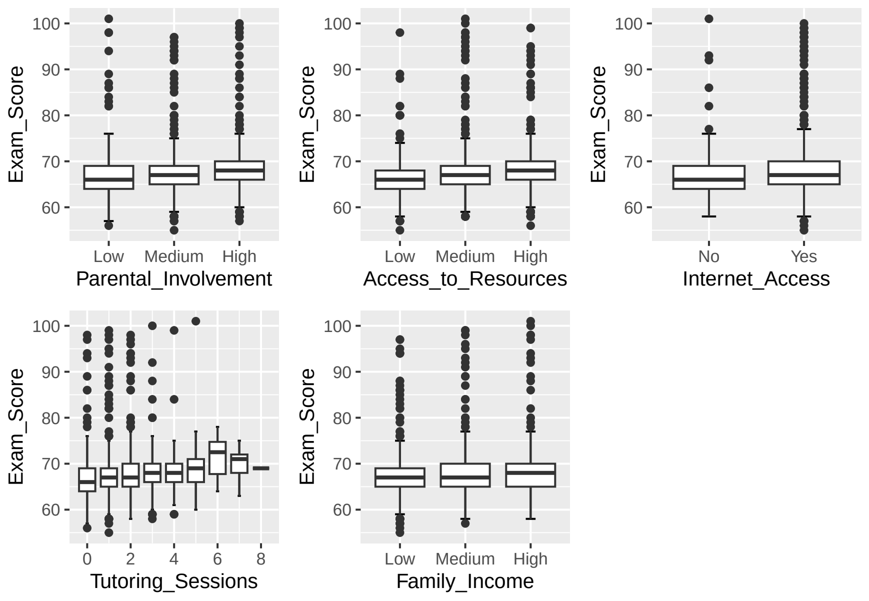

Figure 4: Student Home Environment

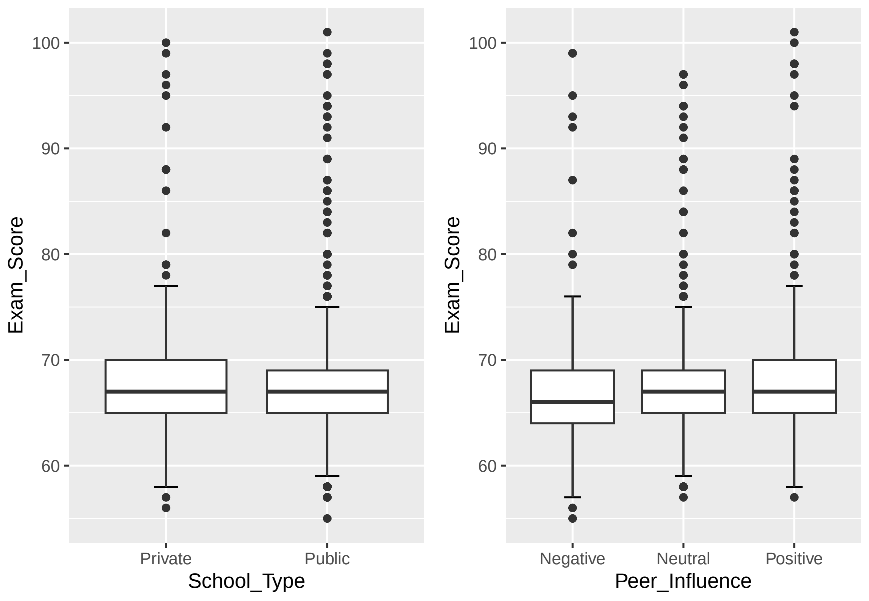

Figure 5: Student School Environment

Some features show obvious correlation such as Hours_Studied and Attendance; studying more hours and attending class a higher percentage of time positively impacts exam score. Previous score shows a very slight positive correlation as well. The other variables are not as clear as the error bars on the box plots overlap, indicating the differences may not be statistically significant and further analysis must be performed to understand the data. Viewing the proportion table for parent involvement shows the proportion of students that have low parental involvement and fail is 30.5%, which is approximately 8.5% higher than the overall failure rate. Students that have high parental involvement fail 14.8% of the time, which is approximately 7% lower than the overall rate. Failure rate for medium parental involvement is about the same as the overall rate. Through comparing proportions we see parental involvement is a significant feature and having low or high parental involvement will impact student performance.

Table 3: Parent Involvement proportion

Pass	Low	Medium	High
0	0.305	0.226	0.148
1	0.695	0.774	0.852

Some features appear to not have a statistically significant impact on student performance. For example, the number of hours per night a student sleeps all have pass/fail proportions that are similar to the overall rate.

Table 4: Sleep Hours proportion

Sleep_Hours	0	1
4	0.201	0.799
5	0.213	0.787
6	0.234	0.766
7	0.215	0.785
8	0.208	0.792
9	0.240	0.760
10	0.218	0.782

In addition to sleep hours, school type and gender appear to not be statistically relevant based on their graphs and proportion tables. All features were used in model development; graphs, and proportion tables for all 16 features are available in the code.

Methods

Exam score is predicted using regression models; these model's performance is evaluated using RMSE. RMSE is a measure of the difference between values predicted by a model and actual values observed in the dataset. The equation for RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum(\hat{y} - y)^2}$$

with y representing the actual score, $\hat{y}$ representing the prediction, and N representing the number of observations, with the sum occurring over all the observations. RMSE can be interpreted similarly to standard deviation; an RMSE of 1 indicates that the error in the model is equivalent to +/- 1 point in exam score. Pass is predicted using classification models; these model's performances are evaluated using accuracy. Accuracy is defined as the overall proportion that is predicted correctly (Irizzary, 2019). Accuracy is calculated using either the mean() function in base R or confusionMatrix() function from the caret package.

Train and Test sets

In order to avoid over training the model the cleaned dataset is split into a train set and a test set, with 20% of the cleaned data in the test set. The train set is used to develop the model, which is then tested on the test set to determine model performance. When predicting "Exam_Score" using regression models, the "Pass" variable is removed as it is based on "Exam_Score" and would result in grave overtraining. Similarly, the "Exam_Score" variable is removed when predicting "Pass" using classification models.

```r
#create test and train sets
set.seed(40, sample.kind = "Rounding") #if using R 3.6 or later
index <- createDataPartition(y = data_clean$Exam_Score, times = 1, p = 0.2, list = FALSE)
train <- data_clean[-index, ] %>% select(-Pass) # remove Pass for reg models
test <- data_clean[index,] %>% select(-Pass) # remove Pass for reg models
train_pass <- data_clean[-index, ] %>% select(-Exam_Score) # remove Exam_Score for class models
test_pass <- data_clean[index,] %>% select(-Exam_Score) # remove Exam_Score for class models
```

An even split can be confirmed using either the mean of "Exam_Score" or "Pass" = 1.

Table 5: Exam_Score mean

Train	Test
67.2	67.2

Models

The caret package (short for **C**lassification **A**nd **RE**gression **T**raining) (Kuhn 2017) is used extensively in model development. This package includes all functionality needed to create and develop a predictive model and streamlines syntax across a variety of models. The models **glm**, **knn**, and **rf** are used for this analysis because they can be used for continuous and categorical input and output in both both regression and classification models; to find the best model I compare the performances of regression and classification techniques. GLM (generalized linear model) is used because it finds a generalized linear fit and can accept variables and output features that have a variety of distributions. KNN (K nearest neighbor) is used to explore the impact of neighboring datapoints; it also allows for cross validation. The train control set-up of 5-fold cross validation with 2 repeats is used due to size of the dataset and processing limitations of my system. This is held constant for knn and rf models, regression and classification. RF (random forest) is used because it averages the results of many decision trees to make predictions. The tuning parameter mtry, which determines the number of randomly selected variables considered at each split when growing a decision tree, is tuned when running this model using a 3 step process. First the model is run using the default values of 2, 11, and 21 for mtry. The best default mtry value is identified, and the second step consists of running the model with a range of values for mtry that are centered around the value selected in step 1. Finally, the model is run one last time with the best value of mtry as selected in the tuning step.

```r
################
# Regression
################
#GLM
set.seed(41, sample.kind = "Rounding")#if using R 3.6 or later
train_glm <- train(Exam_Score~., data = train, method = "glm")
preds_glm <- predict(train_glm, test)
rmse_glm <- RMSE(preds_glm, test$Exam_Score)
results <- tibble(Model="GLM", RMSE = rmse_glm)

#KNN with cross validation
tc <- trainControl("cv", number=5, repeats = 2)
set.seed(42, sample.kind = "Rounding")#if using R 3.6 or later
train_knn <- train(Exam_Score~.,
                   data = train,
                   method = "knn",
                   trControl = tc,
                   preProcess = c("center", "scale"),
                   tuneLength = 15) # start with default k=5 and increment by 2 15 times
preds_knn <- predict(train_knn, test)
rmse_knn <- RMSE(preds_knn, test$Exam_Score)
results <- bind_rows(results,
                          data.frame(Model="KNN", RMSE = rmse_knn))
```

```r
#random forest with cross validation and hyperparameter tuning
#this code can take some time depending on processing power
#default
set.seed(43, sample.kind = "Rounding")#if using R 3.6 or later
rf_default <- train(Exam_Score~.,
                    data=train,
                    method = "rf",
                    trControl=tc) ###samples mtry = (2,11,12)
```

```r
#tune mtry
#this code can take some time depending on processing power
tg <- expand.grid(.mtry=c(6:13))
set.seed(44, sample.kind = "Rounding")#if using R 3.6 or later
rf_mtry <- train(Exam_Score~.,
                 data=train,
                 method = "rf",
                 trControl=tc,
                 tuneGrid=tg)
best_mtry <- rf_mtry$bestTune$mtry
```

```r
#final rf model using best_mtry from previous tuning
#this code can take some time depending on processing power
set.seed(45, sample.kind = "Rounding")#if using R 3.6 or later
train_rf <- train(Exam_Score~.,
                    data=train,
                    method = "rf",
                    trControl=tc,
                    tuneGrid= expand.grid(.mtry=best_mtry))
preds_rf <- predict(train_rf, test)
rmse_rf <- RMSE(preds_rf, test$Exam_Score)
results <- bind_rows(results,
                     data.frame(Model="Random Forest", RMSE = rmse_rf))
```

```r
######################
# Classification
######################
#Logistic
set.seed(46, sample.kind = "Rounding")
train_class_glm <- train(Pass ~ .,
                        data = train_pass,
                        method = "glm")
preds_class_glm <- predict(train_class_glm, test_pass)
acc_glm <- mean(preds_class_glm == test_pass$Pass)
results_acc <- tibble(Model="GLM", Accuracy = acc_glm)

#KNN with cross validation
```

```r
set.seed(47, sample.kind = "Rounding")
train_class_knn <- train(Pass~.,
                        data = train_pass,
                        method = "knn",
                        trControl = tc,
                        preProcess = c("center", "scale"),
                        tuneLength = 15)
preds_class_knn <- predict(train_class_knn, test_pass)
acc_knn <- confusionMatrix(preds_class_knn, test_pass$Pass)$overall[["Accuracy"]]
results_acc <- bind_rows(results_acc,
                          data.frame(Model="KNN", Accuracy = acc_knn))
```

```r
#random forest with cross validation and hyperparameter tuning
#default
set.seed(48, sample.kind = "Rounding")#if using R 3.6 or later
rf_class_default <- train(Pass~.,
                    data=train_pass,
                    method = "rf",
                    trControl=tc) ###samples mtry = (2,11,12)
```

```r
#tune mtry
# tg same as reg,  #+/- 5 from previous model best mtry 11
set.seed(49, sample.kind = "Rounding")#if using R 3.6 or later
rf_class_mtry <- train(Pass~.,
                 data=train_pass,
                 method = "rf",
                 trControl=tc,
                 tuneGrid=tg)
best_mtry <- rf_class_mtry$bestTune$mtry
```

```r
#final rf model using best_mtry from previous tuning
set.seed(50, sample.kind = "Rounding")#if using R 3.6 or later
train_class_rf <- train(Pass~.,
                  data=train_pass,
                  method = "rf",
                  trControl=tc,
                  tuneGrid= expand.grid(.mtry=best_mtry))
preds_class_rf <- predict(train_class_rf, test)
acc_rf <- confusionMatrix(preds_class_rf, test_pass$Pass)$overall[["Accuracy"]]
results_acc <- bind_rows(results_acc,
                         data.frame(Model="Random Forest", Accuracy = acc_rf))
```

Results

The results for the regression models predicting the outcome "Exam_Score" are below. The best regression model with the lowest RMSE is **glm**.

Table 6: Regression results

Model	RMSE
GLM	2.23
KNN	2.93
Random Forest	2.43

The results for the classification models predicting the outcome "Pass" are below. The best classification model with the highest accuracy is **glm**.

Table 7: Classification results

Model	Accuracy
GLM	0.958
KNN	0.844
Random Forest	0.919

This project seeks to accurately identify the students that will fail so they can receive intervention and support in hopes of improving their outcome. Therefore the overall best model maximizes the true positives, or the number of students that are predicted to receive a failing score and actually fail, and minimizes the false negatives, or the number of students that are predicted to receive a passing score but actually fail.

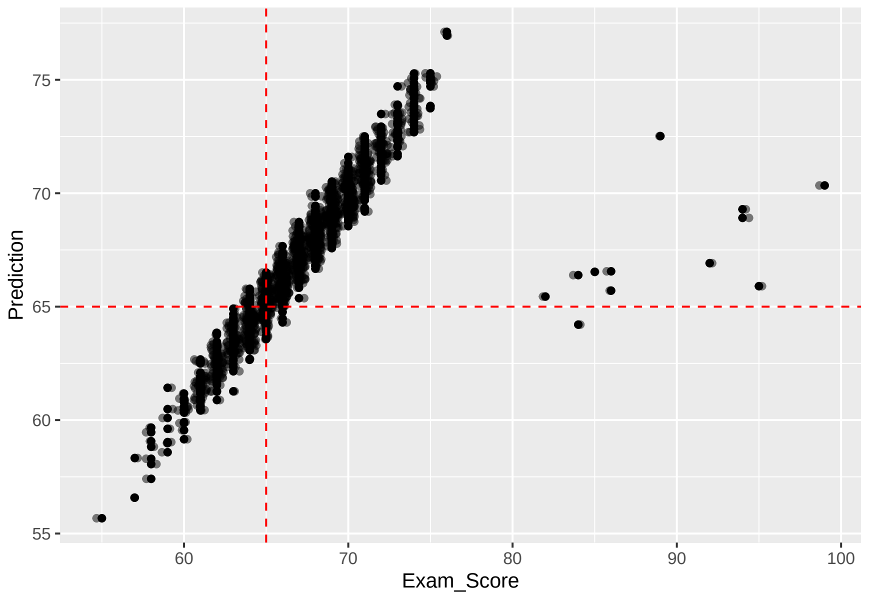

Figure 6: GLM Regression results plot

In the graph above, the datapoints in the upper left quadrant represent false negatives from the **glm** regression model. These datapoints represent students that need help (they will receive failing exam scores) but aren't identified as needing help (they are predicted to pass). There are 13 students at risk in this model. By viewing the confusion matrix of the glm classification model, we see the number of students predicted to pass but actually failed is 34.

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    0    1
##          0  267   21
##          1   34 1001
##
##                Accuracy : 0.958
##                  95% CI : (0.946, 0.969)
##     No Information Rate : 0.772
##     P-Value [Acc > NIR] : <0.0000000000000002
##
##                   Kappa : 0.88
##
##  Mcnemar's Test P-Value : 0.106
##
##             Sensitivity : 0.887
##             Specificity : 0.979
##          Pos Pred Value : 0.927
##          Neg Pred Value : 0.967
##              Prevalence : 0.228
##          Detection Rate : 0.202
##    Detection Prevalence : 0.218
##       Balanced Accuracy : 0.933
##
##        'Positive' Class : 0
##
```

The best model is therefore identified as the **glm** regression model as it minimizes the overall false negatives. This is confirmed by reviewing each model's sensitivity. GLM regression has the highest sensitivity.

```
glm_truepos <- compare_glm %>% filter(Preds < 65 & Exam_Score < 65) %>% nrow()
glm_sens <- glm_truepos/(glm_truepos+glm_falseneg)
glm_class_sens <- confusionMatrix(preds_class_glm, test_pass$Pass)$byClass[c("Sensitivity")]

tibble("GLM Regression" = glm_sens,
       "GLM Classification" = glm_class_sens) %>% kable(caption="Model Sensitivity")
```

Table 8: Model Sensitivity

GLM Regression	GLM Classification
0.957	0.887

Finally, we review the important variables used in the GLM regression model. Teachers, administrators, and legislators can use this information to create programs that address shortcomings in these areas. Attendance has the largest impact followed by hours studied; one idea to improve student success is to incentivize students to attend class and create dedicated time for studying during the school day.

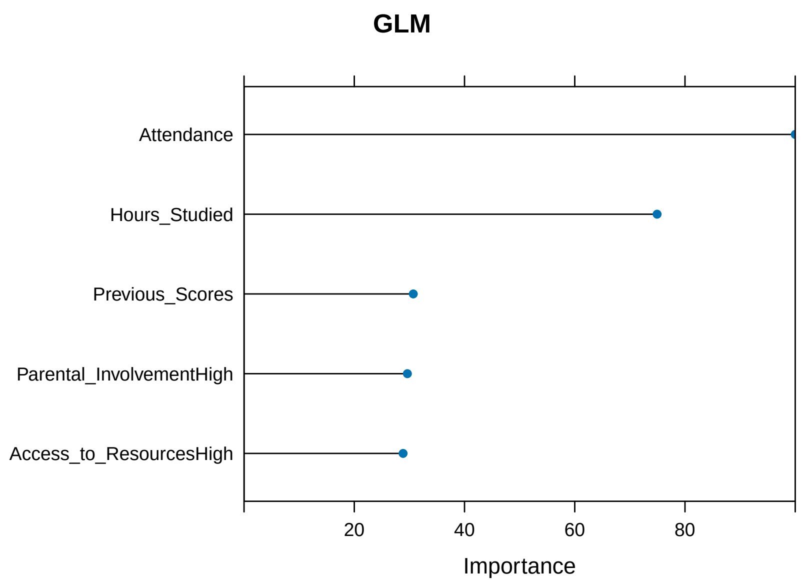

Figure 7: Important variables used in GLM regression

Conclusions

For this project I used various machine learning models and techniques to predict academic failure. The versatility of the caret package allowed exploration of continuous models to predict a student's exam score and categorical models to predict whether a student would pass or fail. The best model is **glm** regression because it has the best sensitivity and minimizes the number of students that are predicted to receive a passing score but actually fail. The model can be used to identify students that need support. It can also be used to understand the important variables that influence success, and programs can be created with the goal of improving student outcomes by removing personal, home, and school barriers.

A major limitation of this work is that it is completed on a synthetic dataset and cannot be applied to real world scenarios. In the future I intend to gather a dataset representing Oregon schools, run a similar analysis, and share the results with leaders of the Tigard-Tualatin School District and the Oregon Department of Education. For this future work I will use a more robust cross validation set-up and hypertune additional parameters for the models. Additionally I would like to better understand the interactions between predictors and plan to implement matrix factorization and principle component analysis in order to explore different machine learning techniques and to identify patterns in the data.

References

Baker, R.S. (2023). *Big Data and Education.* EdX. https://learning.edx.org/course/course-v1:PennX+BDE1x+1T2021/home.

Irizarry, R.A. (2019). *Introduction to Data Science Data Analysis and Prediction Algorithms with R.* https://rafalab.dfci.harvard.edu/dsbook/.

Kuhn, Max. (2019). *The caret Package.* https://topepo.github.io/caret/.

Lainguyn123. Student Performance Factors. Retrieved November 11, 2024 from https://www.kaggle.com/datasets/lainguyn123/student-performance-factors/data.