PSTAT 131: Final Project

Sharon Nguyen, Matt Lee, Paul Song

3/8/2022

Contents

Alcohol Consumption in Secondary Students

By Sharon Nguyen, Matt Lee, Paul Song

Abstract

For this project our group aimed to predict the grades of students from a Portuguese secondary school from a dataset containing a number of variables that were obtained from a survey. We first analyzed each variable and considered them by objective importance. After looking through each predictor we found it best to order the data rather than keep it in its original form. The way the dataset accounted for grades was redundant so our group decided to render the variable to binary values. We thought it best to use histograms in visualizing the relationship of the predictors. The classification methods that we saw best fit were our Decision Tree, Random Forest, and K-Nearest-Neighbors in predicting final grade.

Data layout and reformatting was crucial to our project. Without it, a lot of the classification methods would not run correctly or at all. We found that the organizational method of the data is what made it difficult to process for our algorithms. If the data were not so ordinal and grouped together in categorical ways, we would have a better understanding and have more information at our disposal.

Introduction

By finding key variables that predict final grades of students within a Portuguese secondary school, we can learn the best ways to provide support for students academically.. We want to know where students are most affected by external factors so that we can further adjust our resources to the quickly changing societal needs. The variables we considered were based on our own opinions about possible influences to final grades.

For example, Drinking alcohol on weekends is not an uncommon practice amongst European high school students, and the assumption is this has a negative correlation to students' grades. However, we are here to find out whether this is indeed the case.

In our project, we decided to take a dataset containing numerous characteristics of students in a secondary (high) school in Porto, Portugal in an attempt to predict the final grade of students based on these attributes. We mixed and matched different characteristics to find significant or intriguing correlation between such characteristics.

As a group, we've decided to go with the classification route; hence, using a decision tree, random forest, and K-Nearest-Neighbors. We converted the type of our original dataset containing numeric variables to binary (Pass/No Pass) and ordered data.

Why we chose this data

We chose this particular dataset because as college students, final grades matter the most. What we do in our freetime greatly influences our performance in school. Drinking alcohol on a weekend is not uncommon for many college students. This project is relatable in terms of what factors affect us as students. We wanted to take it upon ourselves to investigate a real world problem that we struggle with and can seek knowledge from.

Grades are heavily influenced by a lot of factors. When taking into account final grades, many factors come into play. Did the student have access to internet? How long does it take for the student to get to school? Does the student have a lot of free time on their hands? Or even, did the student spend too much time consuming alcohol? Utilizing the dataset from UCI Machine Learning, our group is attempting to see the significance that certain variables have in predicting a student's final grade.

We chose to look at the data concerning students in a Portuguese class as it has a total of 649 observations.

Loading Data and Packages

```r
dat <- read.csv("student-por.csv")

# Select wanted variables for analysis
data <- dat %>% select(traveltime, studytime, failures, higher, internet, famrel, freetime, goout, Dalc

str(data)
```

```
## 'data.frame':    649 obs. of  14 variables:
##  $ traveltime: int  2 1 1 1 1 1 1 2 1 1 ...
##  $ studytime : int  2 2 2 3 2 2 2 2 2 2 ...
##  $ failures  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ higher    : chr  "yes" "yes" "yes" "yes" ...
##  $ internet  : chr  "no" "yes" "yes" "yes" ...
##  $ famrel    : int  4 5 4 3 4 5 4 4 4 5 ...
##  $ freetime  : int  3 3 3 2 3 4 4 1 2 5 ...
##  $ goout     : int  4 3 2 2 2 2 4 4 2 1 ...
##  $ Dalc      : int  1 1 2 1 1 1 1 1 1 1 ...
##  $ Walc      : int  1 1 3 1 2 2 1 1 1 1 ...
##  $ health    : int  3 3 3 5 5 5 3 1 1 5 ...
##  $ absences  : int  4 2 6 0 0 6 0 2 0 0 ...
##  $ G2        : int  11 11 13 14 13 12 12 13 16 12 ...
##  $ G3        : int  11 11 12 14 13 13 13 13 17 13 ...
```

Variable Analysis

These are the key variables that our final project will utilize within our modeling. A brief explanation of each is provided down below.

`traveltime`: home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)

`studytime`: weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)

`failures`: number of past class failures (numeric: n if 1<=n<3, else 4)

`higher`: wants to take higher education (binary: yes or no)

`internet`: Internet access at home (binary: yes or no)

`famrel`: quality of family relationships (numeric: from 1 - very bad to 5 - excellent)

`freetime`: free time after school (numeric: from 1 - very low to 5 - very high)

`goout`: going out with friends (numeric: from 1 - very low to 5 - very high)

`Dalc`: workday alcohol consumption (numeric: from 1 - very low to 5 - very high)

`Walc`: weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)

`health`: current health status (numeric: from 1 - very bad to 5 - very good)

`absences`: number of school absences (numeric: from 0 to 93)

`G2`: second period grade (numeric: from 0 to 20)

`G3`: final grade (numeric: from 0 to 20, output target)

Data Cleaning

```
# Make variables ordered
data$traveltime <- factor(data$traveltime, ordered=TRUE, labels = c('<15 min', '15 to 30 min.', '30 min
  # ordered(data$traveltime, levels = c(1:4), labels = c('<15 min', '15 to 30 min.', '30 min. to 1 hour
data$studytime <- factor(data$traveltime, ordered=TRUE, labels = c('<2 hours', '2 to 5 hours', '5 to 10
  # ordered(data$studytime, levels = c(1:4), labels = c('<2 hours', '2 to 5 hours', '5 to 10 hours', '>
data$failures <- ordered(data$failures, levels = c(0:3), labels = c('0', '1', '2', '3'))
data$higher <- factor(data$higher, labels = c('no', 'yes'))
data$internet <- factor(data$internet, labels = c('no', 'yes'))
data$famrel <- factor(data$famrel, ordered=TRUE, labels = c("very bad", "bad", "fair", "good", "exceller

# freetime to Walc
for(i in 7:10){
  data[,i] <- factor(data[,i], ordered=TRUE, labels = c("very low", "low", "medium", "high", "very high
}

data$health <- factor(data$health, ordered=TRUE, labels = c("very bad", "bad", "fair", "good", "very goo
# data$G2 <- ordered(data$G2, levels = c(0:20))
# data$G3 <- ordered(data$G3, levels = c(0:20))

# 2 Binary Pass/No Pass
#data <- data %>% mutate(grade=ifelse(G3/20 >= 0.7, 1, 0))

data <- data %>% mutate(grade=factor(ifelse(G3/20 >= 0.7, 'Pass', 'No Pass'),
                                      levels = c('No Pass', 'Pass')))




# view strucutre
str(data)
```

```
## 'data.frame':    649 obs. of  15 variables:
##  $ traveltime: Ord.factor w/ 4 levels "<15 min"<"15 to 30 min."<..: 2 1 1 1 1 1 1 2 1 1 ...
##  $ studytime : Ord.factor w/ 4 levels "<2 hours"<"2 to 5 hours"<..: 2 1 1 1 1 1 1 2 1 1 ...
##  $ failures  : Ord.factor w/ 4 levels "0"<"1"<"2"<"3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ higher    : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ internet  : Factor w/ 2 levels "no","yes": 1 2 2 2 1 2 2 1 2 2 ...
##  $ famrel    : Ord.factor w/ 5 levels "very bad"<"bad"<..: 4 5 4 3 4 5 4 4 4 5 ...
##  $ freetime  : Ord.factor w/ 5 levels "very low"<"low"<..: 3 3 3 2 3 4 4 1 2 5 ...
##  $ goout     : Ord.factor w/ 5 levels "very low"<"low"<..: 4 3 2 2 2 2 4 4 2 1 ...
##  $ Dalc      : Ord.factor w/ 5 levels "very low"<"low"<..: 1 1 2 1 1 1 1 1 1 1 ...
##  $ Walc      : Ord.factor w/ 5 levels "very low"<"low"<..: 1 1 3 1 2 2 1 1 1 1 ...
##  $ health    : Ord.factor w/ 5 levels "very bad"<"bad"<..: 3 3 3 5 5 5 3 1 1 5 ...
##  $ absences  : int  4 2 6 0 0 6 0 2 0 0 ...
##  $ G2        : int  11 11 13 14 13 12 12 13 16 12 ...
##  $ G3        : int  11 11 12 14 13 13 13 13 17 13 ...
##  $ grade     : Factor w/ 2 levels "No Pass","Pass": 1 1 1 2 1 1 1 1 2 1 ...
```

$$\text{grade} = \begin{cases} \text{No Pass}, & \text{if } [\frac{G3}{20} \times 100] < 70\% \\ \text{Pass}, & \text{if } [\frac{G3}{20} \times 100] \geq 70\% \end{cases}$$

Here we wanna order the data because a lot of the variables are ordinal data. If we do not do this than the models will not interpret out data correctly.

Data Split

We split the data by 70% training and 30% test.

```
set.seed(123)
num_samp <- 0.7 * nrow(data)

# t <- model.matrix(grade ~ .-G3, data)
#
# #
# train = sample(nrow(t), num_samp)
# x.train = t[train, ]
# y.train = data[train, ]$grade
# #
# # # The rest as test data
# x.test = t[-train, ]
# y.test = data[-train, ]$grade

student = data %>%
   select(absences, G2, grade)

# Sample 70% observations as training data
train = sample(nrow(data), num_samp)
data.train = student[train,]
# The rest 30% as test data
data.test = student[-train,]

# YTrain is the true labels for grade on the training set
# XTrain is the standardized design matrix
y.train = data.train$grade
x.train = data.train %>% select(-grade)

# YTest is the true labels for grade on the test set, Xtest is the design matrix
y.test = data.test$grade
x.test = data.test %>% select(-grade)


train.indices <- sample(nrow(data), num_samp)
train <- data[train.indices,]
test <- data[-train.indices,]
```

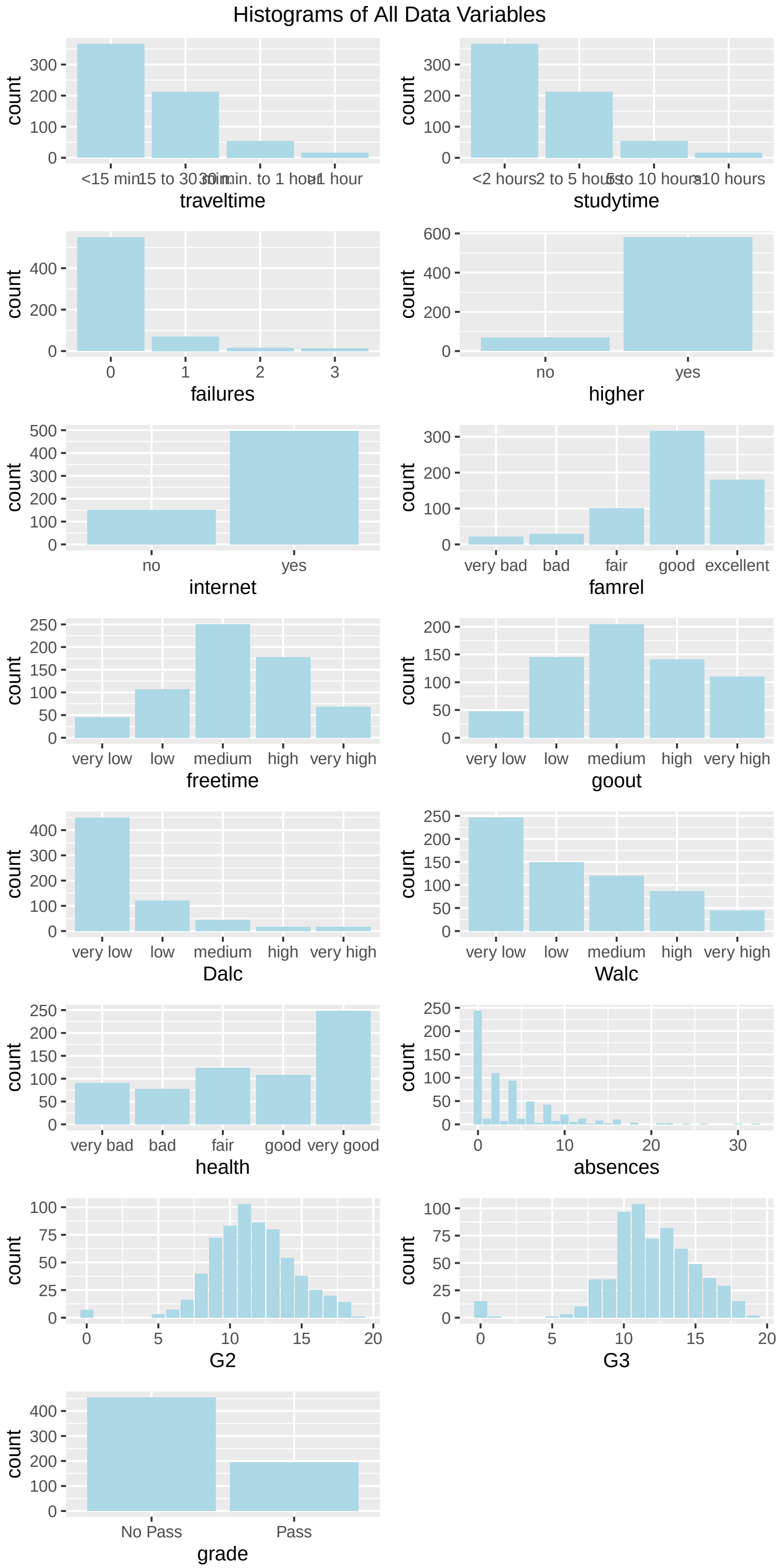
Exploratory Data Analysis

```
for(i in 1:14){
 print(table(data[,i]))
}
```

```
##
##           <15 min     15 to 30 min. 30 min. to 1 hour           >1 hour
##               366                213                54                16
##
##      <2 hours  2 to 5 hours 5 to 10 hours     >10 hours
##           366           213            54            16
##
##   0   1   2   3
## 549  70  16  14
##
##  no yes
##  69 580
##
##  no yes
## 151 498
##
##  very bad       bad      fair      good excellent
##        22        29       101       317       180
##
##  very low       low    medium      high very high
##        45       107       251       178        68
##
##  very low       low    medium      high very high
##        48       145       205       141       110
##
##  very low       low    medium      high very high
##       451       121        43        17        17
##
##  very low       low    medium      high very high
##       247       150       120        87        45
##
##  very bad       bad      fair      good very good
##        90        78       124       108       249
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  18  21  22
## 244  12 110   7  93  12  49   3  42   7  21   5  12   1   8   2  10   3   2   2
##  24  26  30  32
##   1   1   1   1
##
##   0   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##   7   3   7  16  40  72  83 103  86  80  54  38  25  20  14   1
##
##   0   1   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##  15   1   1   3  10  35  35  97 104  72  82  63  49  36  29  15   2
```

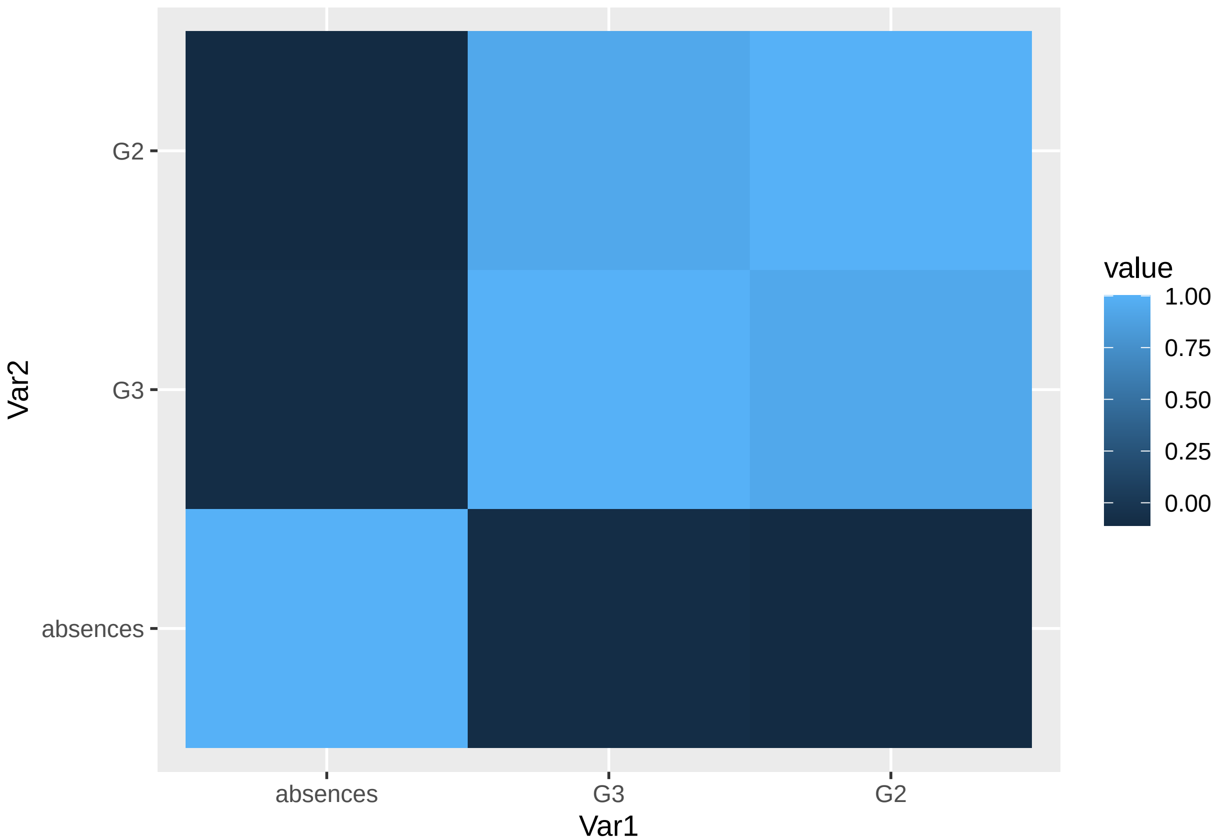
We have a lot of categorical data so making tables was the best way to show the frequency of data among the variables since there is also ordinal data. Afterwards to show the distribution we used histograms.

```
plotlist <- list()
for(i in 1:15){
 p <- ggplot(data, aes_string(x=data[,i])) + geom_histogram(fill='lightblue', stat="count") + xlab(colna
 plotlist[[i]] <- p
}
grid.arrange(grobs=plotlist, ncol=2, top = "Histograms of All Data Variables")
```


Histograms of All Data Variables
count
traveltime
studytime
<15 min
<2 hours
failures
higher
no
yes
internet
famrel
very bad
bad
fair
good
excellent
freetime
goout
very low
low
medium
high
very high
Dalc
Walc
health
very good
absences
G2
G3
grade
No Pass
Pass

```r
# correlation matrix
cor_train <- train

cor_train <- cor_train %>% select(absences, G3, G2)
cormat <- round(cor(cor_train), 2)
melted_cormat <- melt(cormat)
ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_raster()
```




```r
# Get lower triangle of the correlation matrix
get_lower_tri<-function(cormat){
  cormat[upper.tri(cormat)] <- NA
  return(cormat)
}
# Get upper triangle of the correlation matrix
get_upper_tri <- function(cormat){
  cormat[lower.tri(cormat)]<- NA
  return(cormat)
}

upper_tri <- get_upper_tri(cormat)
upper_tri
```

```
##          absences    G3    G2
## absences        1 -0.09 -0.11
## G3             NA  1.00  0.93
```

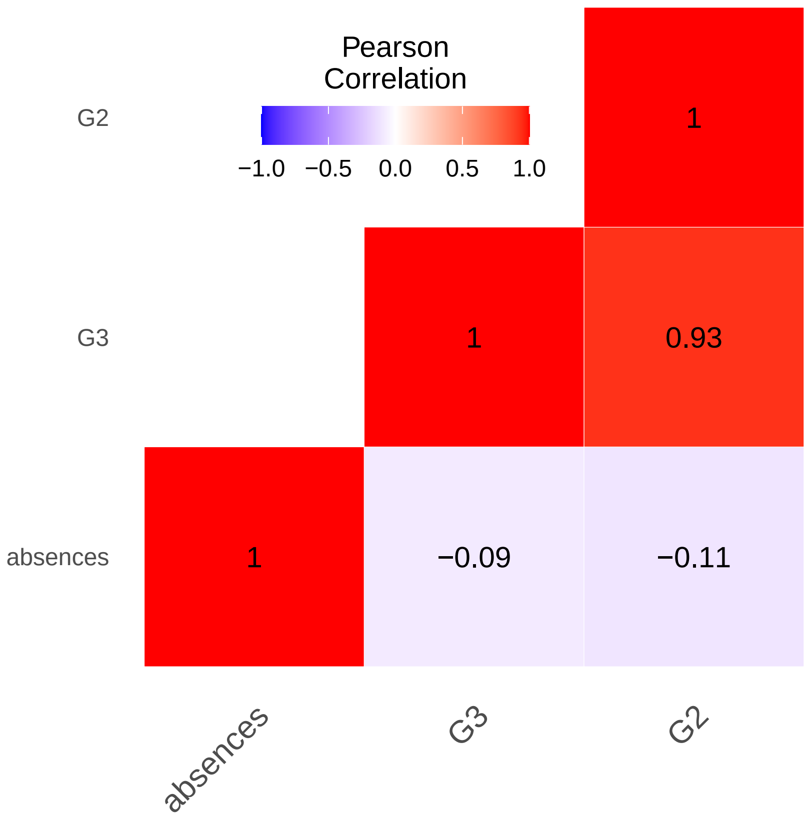
```
## G2             NA    NA  1.00
```

```r
# Melt the correlation matrix
melted_cormat <- melt(upper_tri, na.rm = TRUE)

reorder_cormat <- function(cormat){
# Use correlation between variables as distance
dd <- as.dist((1-cormat)/2)
hc <- hclust(dd)
cormat <-cormat[hc$order, hc$order]
}

# Reorder the correlation matrix
cormat <- reorder_cormat(cormat)
upper_tri <- get_upper_tri(cormat)
# Melt the correlation matrix
melted_cormat <- melt(upper_tri, na.rm = TRUE)
# Create a ggheatmap
ggheatmap <- ggplot(melted_cormat, aes(Var2, Var1, fill = value))+
 geom_tile(color = "white")+
 scale_fill_gradient2(low = "blue", high = "red", mid = "white",
   midpoint = 0, limit = c(-1,1), space = "Lab",
    name="Pearson\nCorrelation") +
  theme_minimal()+ # minimal theme
 theme(axis.text.x = element_text(angle = 45, vjust = 1,
    size = 12, hjust = 1))+
 coord_fixed()
# # Print the heatmap
# print(ggheatmap)


ggheatmap +
geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  panel.grid.major = element_blank(),
  panel.border = element_blank(),
  panel.background = element_blank(),
  axis.ticks = element_blank(),
  legend.justification = c(1, 0),
  legend.position = c(0.6, 0.7),
  legend.direction = "horizontal")+
  guides(fill = guide_colorbar(barwidth = 7, barheight = 1,
                title.position = "top", title.hjust = 0.5))
```



From the heat map, we can see absences and final grade (G3) have low correlation. However, we can also see that second period grade (G2) and final grades (G3) have very positive correlation.

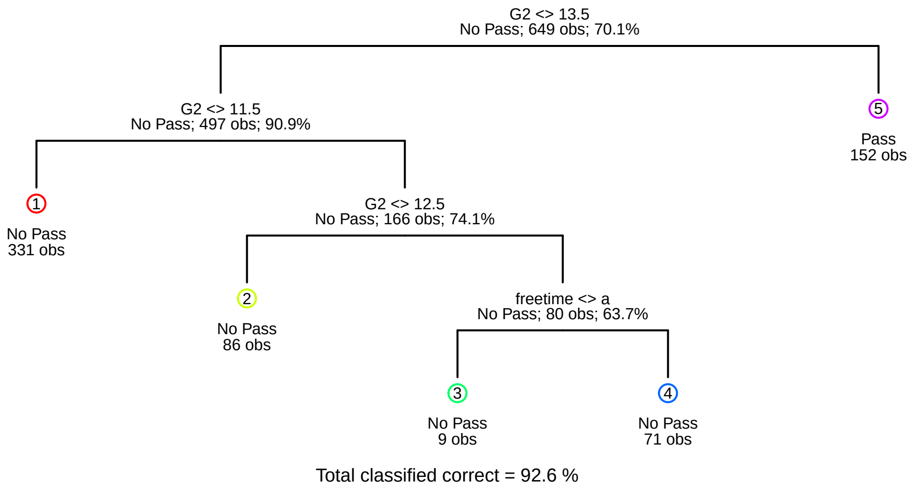
Decision Tree

```
# Entire Data Tree with chosen variables
dat.tree <- tree(grade~ .-G3, data = data)
summary(dat.tree)
```

```
##
## Classification tree:
## tree(formula = grade ~ . - G3, data = data)
## Variables actually used in tree construction:
## [1] "G2"       "freetime"
## Number of terminal nodes:  5
## Residual mean deviance:  0.3515 = 226.4 / 644
## Misclassification error rate: 0.07396 = 48 / 649
```

```
draw.tree(dat.tree, nodeinfo=TRUE, cex = 0.5)
title("Classification Tree Built on All Data Set")
```

Classification Tree Built on All Data Set

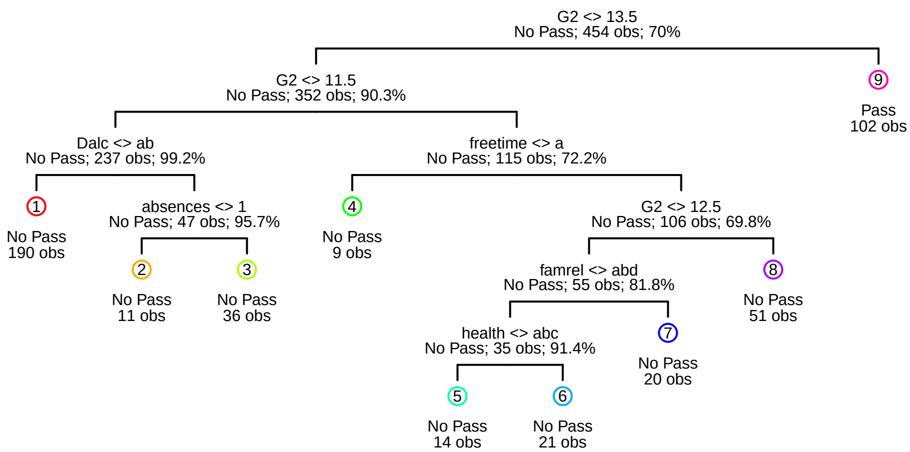



```
data.tree <- tree(grade~ .-G3, data = data, subset = train.indices)
summary(data.tree)
```

```
##
## Classification tree:
## tree(formula = grade ~ . - G3, data = data, subset = train.indices)
## Variables actually used in tree construction:
## [1] "G2"       "Dalc"     "absences" "freetime" "famrel"   "health"
## Number of terminal nodes:  9
## Residual mean deviance:  0.271 = 120.6 / 445
## Misclassification error rate: 0.07489 = 34 / 454
```

```
# length(which(train$grade == 0))
# plot(data.tree)
# text(data.tree, pretty = 0, cex = 0.8)
draw.tree(data.tree, nodeinfo=TRUE, cex = 0.5)
title("Classification Tree Built on Training Set")
```

Classification Tree Built on Training Set



Total classified correct = 92.5 %

```r
# Predict on test set
tree.pred <- predict(data.tree, test, type="class")

# Obtain confusion matrix
error = table(tree.pred, test$grade)
error
```

```
##
## tree.pred No Pass Pass
##   No Pass     134   11
##   Pass          3   47
```

```r
# Test accuracy rate
sum(diag(error))/sum(error)
```

```
## [1] 0.9282051
```

```r
# Test error rate (Classification Error)
1-sum(diag(error))/sum(error)
```

```
## [1] 0.07179487
```

This approach leads to correct predictions for 93% of the locations in the test set. In other words, the test error rate is 7%. This is really equivalent to:

```r
mean(tree.pred != test$grade)
```

```
## [1] 0.07179487
```

Pruning

k-fold Cross-validation

```
# set random see
set.seed(123)

# K-fold cross validation
cv <- cv.tree(data.tree, FUN = prune.misclass, K=10)
cv$size
```

```
## [1] 9 2 1
```

```
# Cross-validation error
cv$dev
```

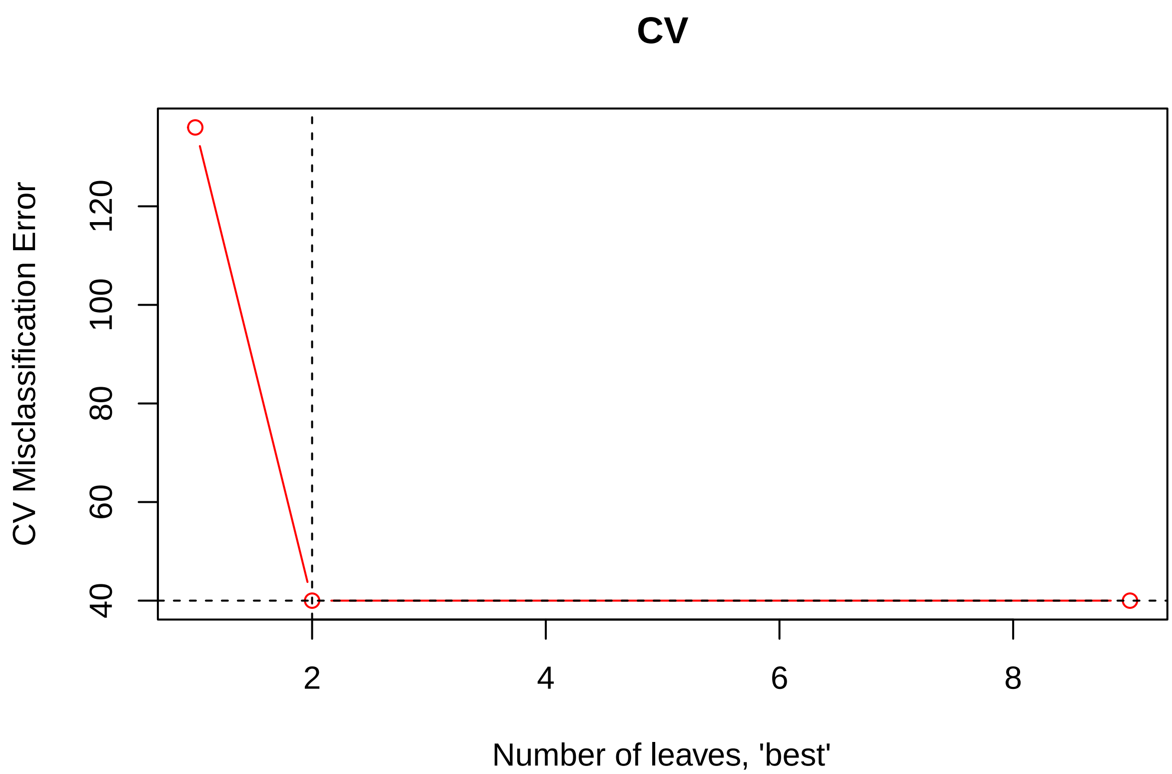
```
## [1]  40  40 136
```

```
# Best size
# tree with 2 nodes is the lowest error
best.cv = min(cv$size[cv$dev == min(cv$dev)])
best.cv
```

```
## [1] 2
```

Error vs. Best Size plot

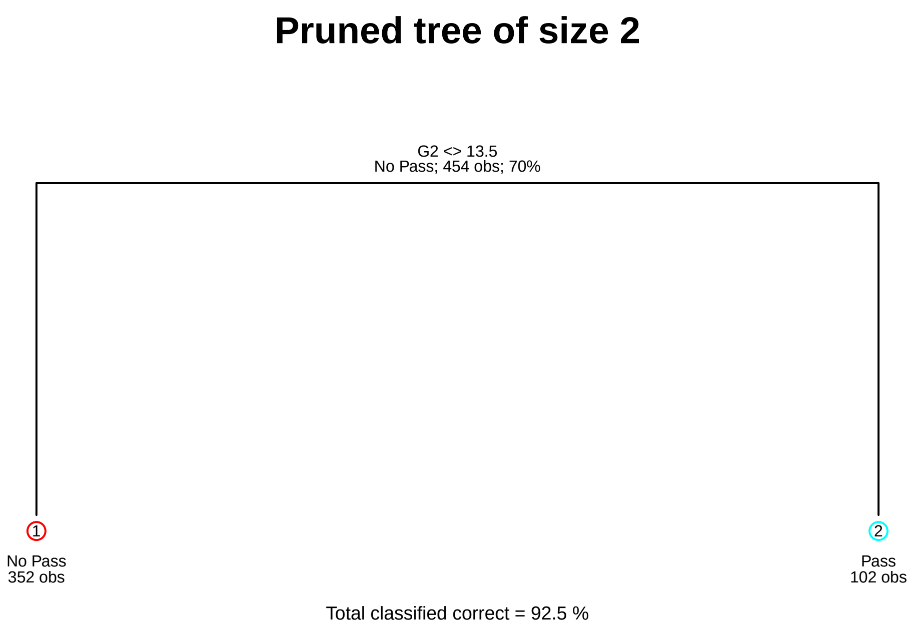
```
# Plot size vs. cross-validation error rate
plot(cv$size , cv$dev, type="b",
     xlab = "Number of leaves, \'best\'", ylab = "CV Misclassification Error",
     col = "red", main="CV")
abline(v=best.cv, lty=2)
# Add lines to identify complexity parameter
min.error = which.min(cv$dev) # Get minimum error index
abline(h = cv$dev[min.error],lty = 2)
```



Prune tree

```
# Prune data.tree
pt.cv = prune.misclass (data.tree, best=best.cv)

# Plot pruned tree
draw.tree(pt.cv, nodeinfo=TRUE, cex = 0.5)
title("Pruned tree of size 2")
```



Respective Test Error Rate for model pt.cv

```
# Predict on test set
pred.pt.cv = predict(pt.cv, test, type="class") # Obtain confusion matrix
err.pt.cv = table(pred.pt.cv, test$grade)
err.pt.cv
```

```
##
## pred.pt.cv No Pass Pass
##    No Pass     134   11
##    Pass          3   47
```

```
 # Test accuracy rate
sum(diag(err.pt.cv))/sum(err.pt.cv)
```

```
## [1] 0.9282051
```

```
# Test error rate (Classification Error)
1-sum(diag(err.pt.cv))/sum(err.pt.cv)
```

```
## [1] 0.07179487
```

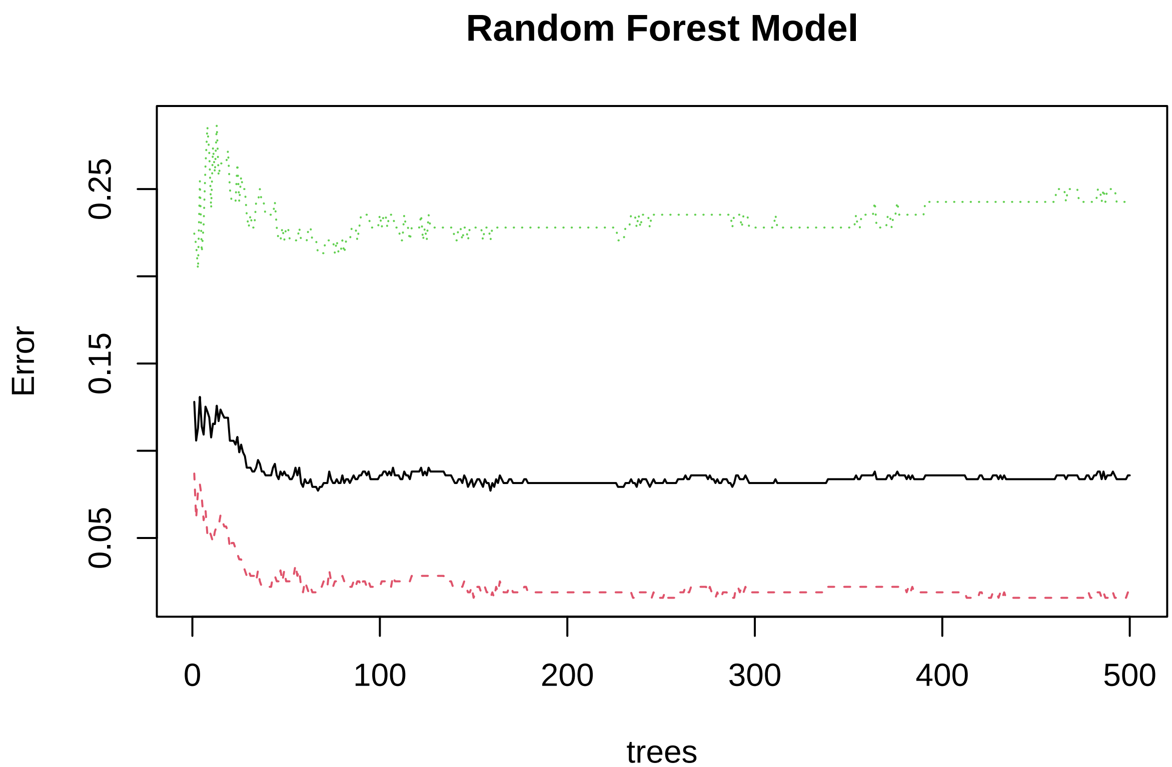
The test error rate for pt.cv is 7%, which is identical to the test error rate for when we used function `tree()`. This means we get a simpler tree for free (without any cost in prediction error rate) by pruning. Thus, we prefer the pruned tree.

Random Forest

```r
# Random Forest
data.rf <- randomForest(grade ~ .-G3, data = data, subset = train.indices, norm.votes = FALSE)
print(data.rf)
```

```
##
## Call:
##  randomForest(formula = grade ~ . - G3, data = data, norm.votes = FALSE,      subset = train.indices)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 8.59%
## Confusion matrix:
##         No Pass Pass class.error
## No Pass     312    6  0.01886792
## Pass         33  103  0.24264706
```

```r
plot(data.rf, main='Random Forest Model')
```




```r
# test error rate calculations
yhat.rf <- predict(data.rf, newdata=test)

# confusion matrix
rf.err <- table(pred = yhat.rf, truth=test$grade)
test.rf.err <- 1 - sum(diag(rf.err))/sum(rf.err)
test.rf.err
```

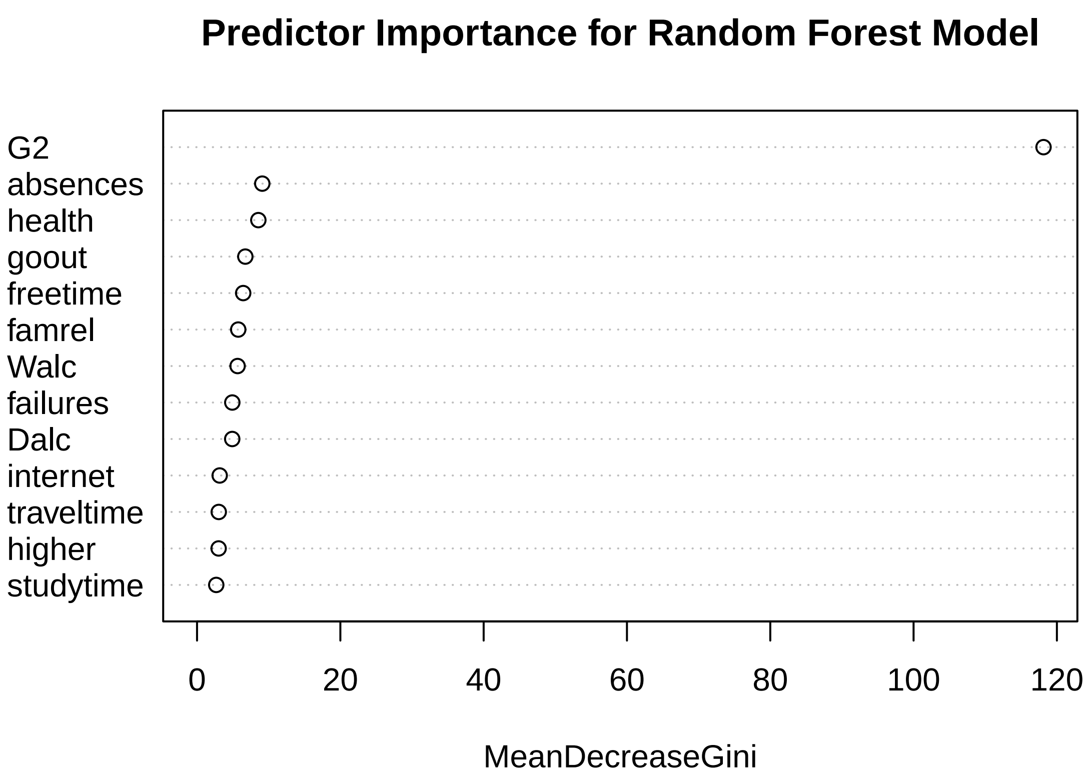
```
## [1] 0.07692308
```

Our test error rate is 7.18%

```r
# list of important variables based on Mean Decrease Gini
importance(data.rf)
```

```
##            MeanDecreaseGini
## traveltime         3.030843
## studytime          2.680907
## failures           4.920657
## higher             3.010082
## internet           3.158961
## famrel             5.747480
## freetime           6.441471
## goout              6.738672
## Dalc               4.911359
## Walc               5.658708
## health             8.555206
## absences           9.104251
## G2               118.134810
```

```r
# plot of important variables
varImpPlot(data.rf, sort=T, main='Predictor Importance for Random Forest Model')
```



We see within our plot that G2 has the highest importance in predicting for final grades.

K Nearest Neighbors (KNN)

```r
# KNN
set.seed(123)

pred.ytrain <- knn(train=x.train, test=x.train, cl=y.train, k=5)

conf.train <- table(predicted=pred.ytrain, true=y.train)
conf.train
```

```
##          true
## predicted No Pass Pass
##   No Pass     310   31
##   Pass          3  110
```

```r
# training error rate
1 - sum(diag(conf.train)/sum(conf.train))
```

```
## [1] 0.07488987
```

```r
# training classifier, making prediction on test set - KNN
pred.ytest <- knn(train=x.train, test=x.test, cl=y.train, k=5)

conf.test <- table(predicted=pred.ytest, true=y.test)
conf.test
```

```
##          true
## predicted No Pass Pass
##   No Pass     141   11
##   Pass          1   42
```

```r
# testing error rate
1 - sum(diag(conf.test)/sum(conf.test))
```

```
## [1] 0.06153846
```

```r
# data.plot.k5 <- data.frame(grade = data.test$grade,
#                            G2 = data.test$G2,
#                            Pred = pred.ytest)
#
# # black dots represent the actual data points in test set
# # red line represents the 2-nn prediction
# data.plot.k5 %>% ggplot(aes(x=G2, y=grade)) + geom_point() +
#     geom_line(aes(x = G2, y=Pred, color="red")) + theme(legend.position = "none") +
#   ggtitle("k = 5")
```

After training the classifier our predictions on the training set led to a training error rate of 7.49%. Afterwards, we trained the classifier and made predictions on the test set, resulting in a testing error rate equal to 6.15%. We then decided to increase our k and see if it would alter the training error rate and test error rate.

```r
set.seed(123)

pred.ytrain <- knn(train=x.train, test=x.train, cl=y.train, k=10)

conf.train <- table(predicted=pred.ytrain, true=y.train)
conf.train
```

```
##          true
## predicted No Pass Pass
##   No Pass     309   36
##   Pass          4  105
```

```r
# training error rate
1 - sum(diag(conf.train)/sum(conf.train))
```

```
## [1] 0.08810573
```

```r
# training classifier, making prediction on test set - KNN
pred.ytest <- knn(train=x.train, test=x.test, cl=y.train, k=10)

conf.test <- table(predicted=pred.ytest, true=y.test)
conf.test
```

```
##          true
## predicted No Pass Pass
##   No Pass     142   14
##   Pass          0   39
```

```r
# testing error rate
1 - sum(diag(conf.test)/sum(conf.test))
```

```
## [1] 0.07179487
```

Similar to the initial K-Nearest-Neighbors approach, after training the classifier, our predictions on the training set led to a 8.81%% training error rate and a testing error rate of 7.71%. Our training error rate increased by 1.32% and our testing error rate remained the same.

Conclusion

In conclusion, after running different classification algorithms to predict `grade`, we were only able to accurately use G2 as a predictor for final grades. By cleaning out unecessary variables within the dataset, we were able to make substantial statistical models. Utilizing a heatmap was a good indicator to combine our G2 and G3 as they were shown to have correlation. Altering our grade value to becoming binary helped with the classification methods in keeping our training error rate and test error rate to stay around 7 - 8%. The UCI Machine Learning database deleted the dataset due to the lack of information and difficulty in statistical modeling.

Due to the dataset containing a lot of ordinal values within each predictor, it was difficult calculating accurate classification models.

References

<https://www.studyineurope.eu/study-in-portugal/grades>

<https://www.kaggle.com/uciml/student-alcohol-consumption?select=student-mat.csv>.