# Student Grade Predictions

## 2023-05-06

```
#install.packages("tidyverse")
#install.packages("readr")
#install.packages("rpart")
#install.packages("rpart.plot")
#install.packages("caret")
#install.packages("e1071")
#install.packages("randomForest")
#install.packages("arules")
#install.packages("arulesViz")
```

```
set.seed(4321)
options(warn=-1)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.2.1     v dplyr   1.1.2
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readr)
library(rpart)
library(rpart.plot)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(e1071)
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(arules)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
##
## Attaching package: 'arules'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
library(arulesViz)
```

```
#Insert the path to the data on your device here: (use / not \)
path <- "C:/Users/Owner/Documents/SU_Q3/IST 707/Project/student-mat.csv"
```

```
student_df <- read.csv(path)
head(student_df)
```

```
##   school sex age address famsize Pstatus Medu Fedu     Mjob     Fjob     reason
## 1     GP   F  18       U     GT3       A    4    4  at_home  teacher     course
## 2     GP   F  17       U     GT3       T    1    1  at_home    other     course
## 3     GP   F  15       U     LE3       T    1    1  at_home    other      other
## 4     GP   F  15       U     GT3       T    4    2   health services       home
## 5     GP   F  16       U     GT3       T    3    3    other    other       home
## 6     GP   M  16       U     LE3       T    4    3 services    other reputation
##   guardian traveltime studytime failures schoolsup famsup paid activities
## 1   mother          2         2        0       yes     no   no         no
## 2   father          1         2        0        no    yes   no         no
## 3   mother          1         2        3       yes     no  yes         no
## 4   mother          1         3        0        no    yes  yes        yes
## 5   father          1         2        0        no    yes  yes         no
## 6   mother          1         2        0        no    yes  yes        yes
##   nursery higher internet romantic famrel freetime goout Dalc Walc health
## 1     yes    yes       no       no      4        3     4    1    1      3
## 2      no    yes      yes       no      5        3     3    1    1      3
## 3     yes    yes      yes       no      4        3     2    2    3      3
## 4     yes    yes      yes      yes      3        2     2    1    1      5
## 5     yes    yes       no       no      4        3     2    1    2      5
## 6     yes    yes      yes       no      5        4     2    1    2      5
##   absences G1 G2 G3
## 1        6  5  6  6
## 2        4  5  5  6
## 3       10  7  8 10
## 4        2 15 14 15
## 5        4  6 10 10
## 6       10 15 15 15
```

### Data Prep

```
#Our Target Audience are those who are really struggling, achieving a 55% in the class or less
11/20
```

```
## [1] 0.55
```

```
#creating the target variable
student_df$Target <- ifelse(student_df$G3 <= 11, 1, 0)
student_df$Target <- as.factor(student_df$Target)
```

Now, the variable types require updating to reflect their true nature

```
str(student_df)
```

```
## 'data.frame':    395 obs. of  34 variables:
##  $ school    : chr  "GP" "GP" "GP" "GP" ...
##  $ sex       : chr  "F" "F" "F" "F" ...
##  $ age       : int  18 17 15 15 16 16 16 17 15 15 ...
##  $ address   : chr  "U" "U" "U" "U" ...
##  $ famsize   : chr  "GT3" "GT3" "LE3" "GT3" ...
##  $ Pstatus   : chr  "A" "T" "T" "T" ...
##  $ Medu      : int  4 1 1 4 3 4 2 4 3 3 ...
##  $ Fedu      : int  4 1 1 2 3 3 2 4 2 4 ...
##  $ Mjob      : chr  "at_home" "at_home" "at_home" "health" ...
##  $ Fjob      : chr  "teacher" "other" "other" "services" ...
##  $ reason    : chr  "course" "course" "other" "home" ...
##  $ guardian  : chr  "mother" "father" "mother" "mother" ...
##  $ 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 3 0 0 0 0 0 0 0 ...
##  $ schoolsup : chr  "yes" "no" "yes" "no" ...
##  $ famsup    : chr  "no" "yes" "no" "yes" ...
##  $ paid      : chr  "no" "no" "yes" "yes" ...
##  $ activities: chr  "no" "no" "no" "yes" ...
##  $ nursery   : chr  "yes" "no" "yes" "yes" ...
##  $ higher    : chr  "yes" "yes" "yes" "yes" ...
##  $ internet  : chr  "no" "yes" "yes" "yes" ...
##  $ romantic  : chr  "no" "no" "no" "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  6 4 10 2 4 10 0 6 0 0 ...
##  $ G1        : int  5 5 7 15 6 15 12 6 16 14 ...
##  $ G2        : int  6 5 8 14 10 15 12 5 18 15 ...
##  $ G3        : int  6 6 10 15 10 15 11 6 19 15 ...
##  $ Target    : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 1 1 ...
```

The values will be updated before the models are created

Check for Null values

```
#The number of rows matches the df even after na.omit() which tells us there is no Null values i
n the data frame
student_df %>%
  na.omit() %>%
  nrow()
```

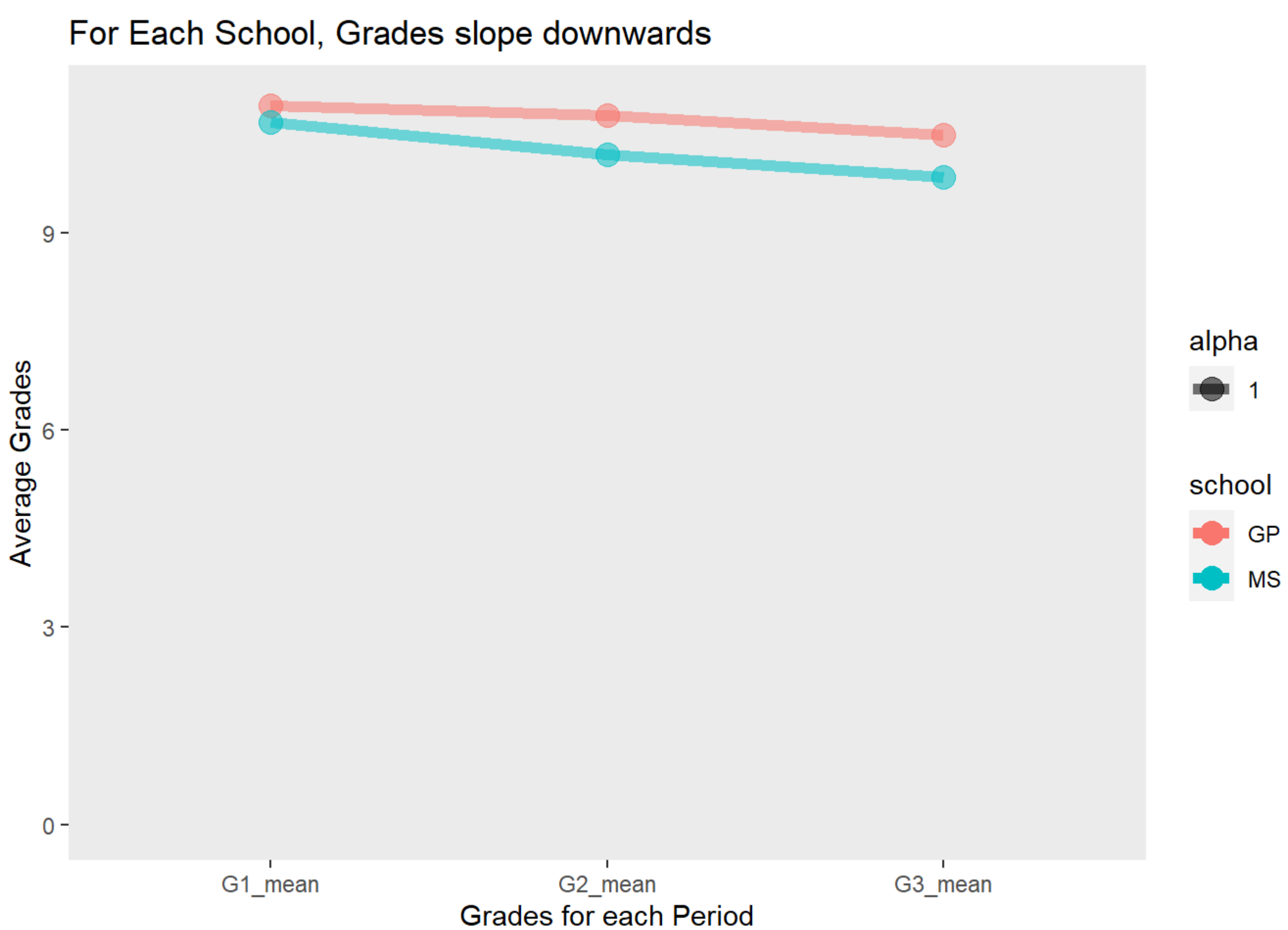
```
## [1] 395
```

Some outliers but want to not remove them as we are focusing on those who fail

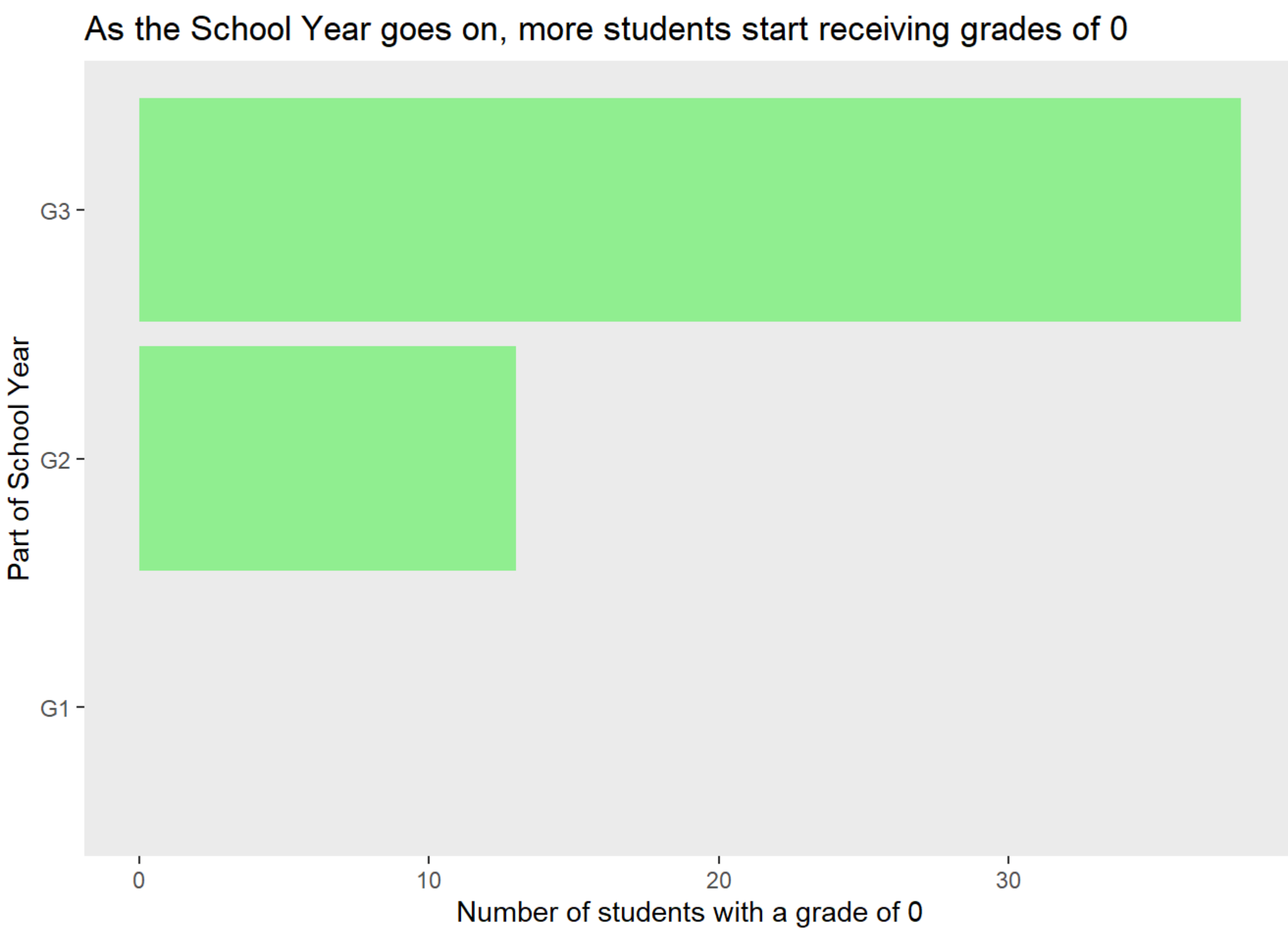
```
student_df %>%
  ggplot() +
  aes(x=sex, y=G3) +
  geom_boxplot(fill = "lightgreen") +
  ggtitle("There are two outliers, both students had 0's") +
  coord_flip() +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```


### Exploratory Data Analysis

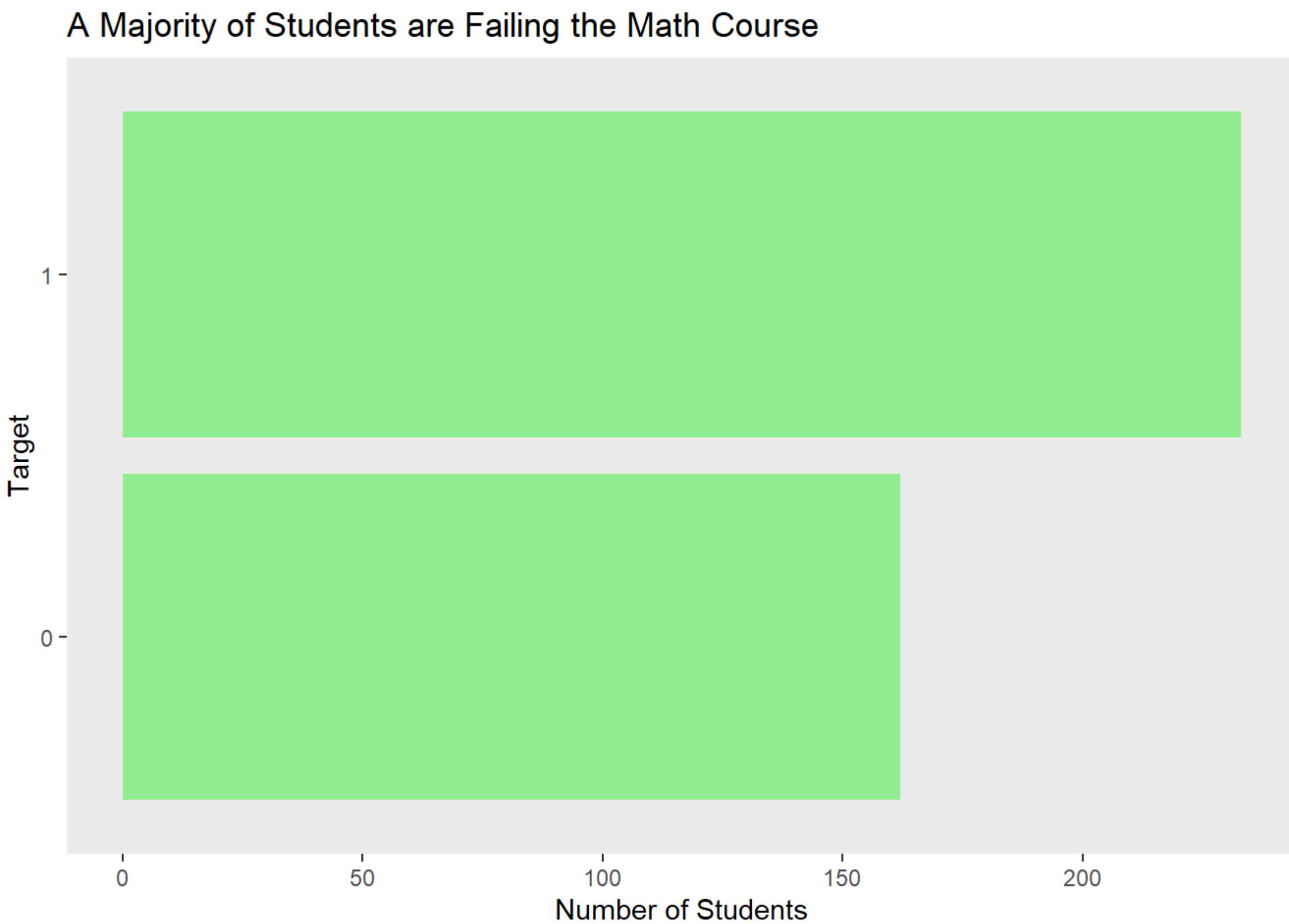
```
student_df %>%
  select(c(school, G1, G2, G3)) %>%
  group_by(school) %>%
  summarise_at(.vars = vars(G1, G2, G3),
               .funs = c(mean = "mean")) %>%
  pivot_longer(cols = c('G1_mean', 'G2_mean', 'G3_mean'),
               names_to = 'Grades',
               values_to = 'Average_Grade') %>%
  ggplot() +
  aes(x = Grades, y = Average_Grade, group = school, color = school, alpha = 1) +
  geom_line(linewidth = 2) +
  geom_point(size = 4) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ylab("Average Grades") +
  xlab("Grades for each Period") +
  ggtitle("For Each School, Grades slope downwards") +
  scale_y_continuous(limits = c(0,11))
```



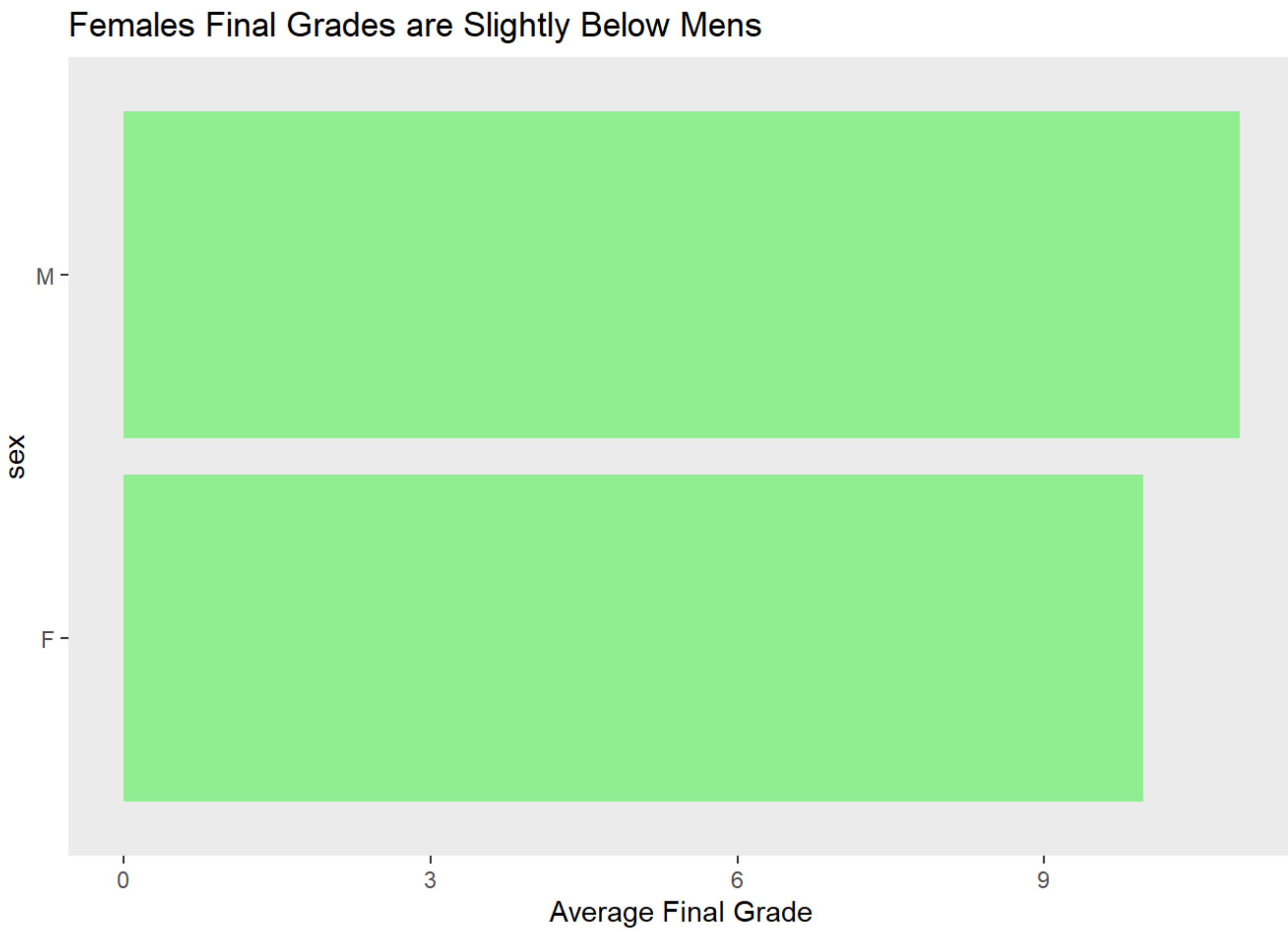
```
## students who get zeros increase a significant amount by period
G1 <- sum(student_df$G1 == 0, na.rm=TRUE)
G2 <- sum(student_df$G2 == 0, na.rm=TRUE)
G3 <- sum(student_df$G3 == 0, na.rm=TRUE)
zero_grades <- data.frame(G3, G2, G1)
zero_grades %>%
  pivot_longer(cols = c(G3, G2,G1),names_to = "Time_of_Year", values_to = "number_of_students")
%>%
  ggplot() +
  aes(x = Time_of_Year, y = number_of_students) +
  geom_bar(stat = 'identity', fill = "lightgreen")  +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  coord_flip() +
  xlab("Part of School Year") +
  ylab("Number of students with a grade of 0") +
  ggtitle("As the School Year goes on, more students start receiving grades of 0")
```



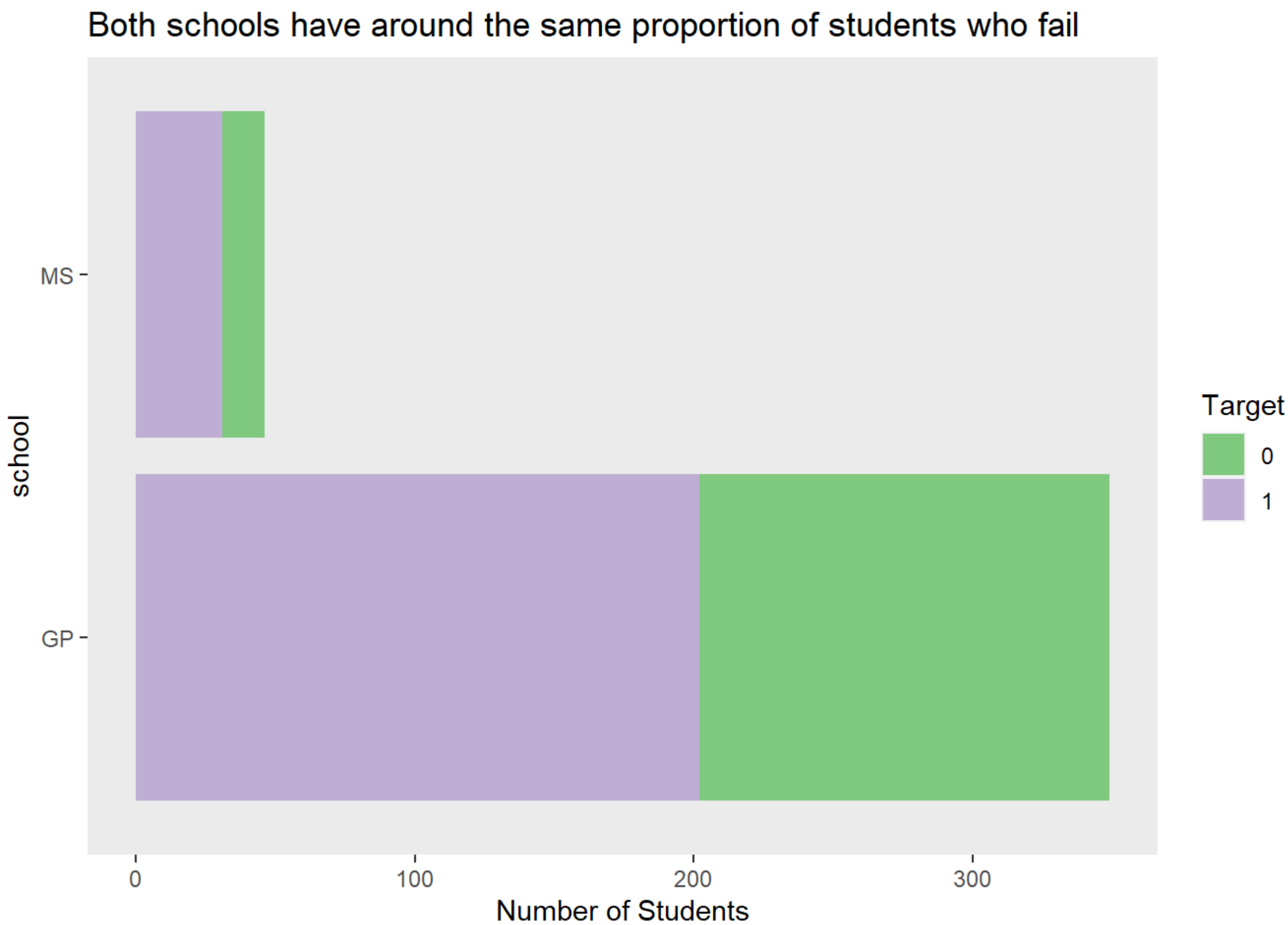
```
student_df %>%
  ggplot() +
  aes(x=Target) +
  geom_histogram(stat = "count", fill = "lightgreen") +
  ggtitle("A Majority of Students are Failing the Math Course") +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ylab("Number of Students") +
  coord_flip()
```



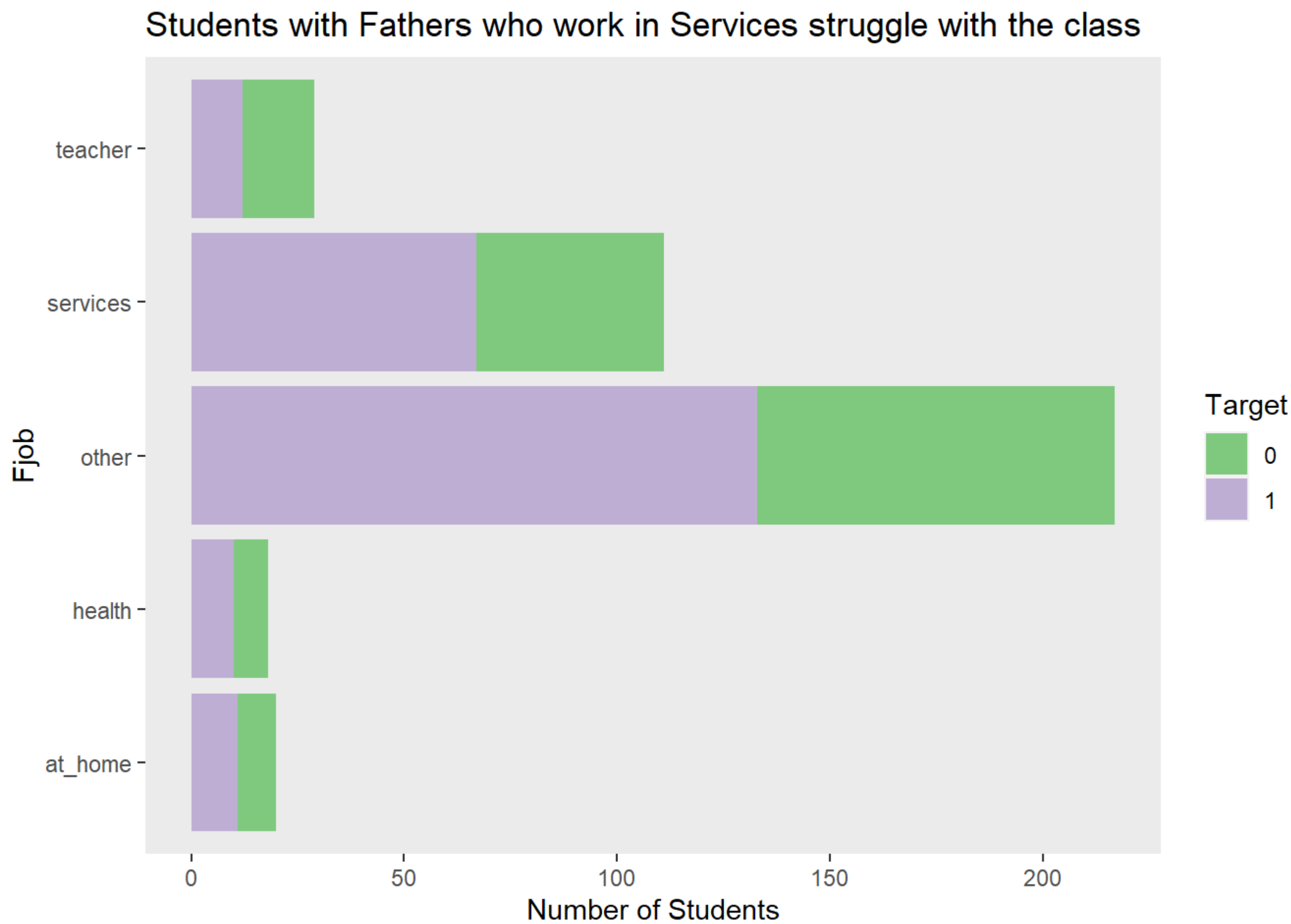

```
student_df %>%
  group_by(sex) %>%
  summarize(average_final_grade = mean(G3)) %>%
  ggplot() +
  aes(x = sex, y=average_final_grade) +
  geom_bar(stat = "identity", fill = "lightgreen") +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ggtitle("Females Final Grades are Slightly Below Mens") +
  ylab("Average Final Grade") +
  coord_flip()
```



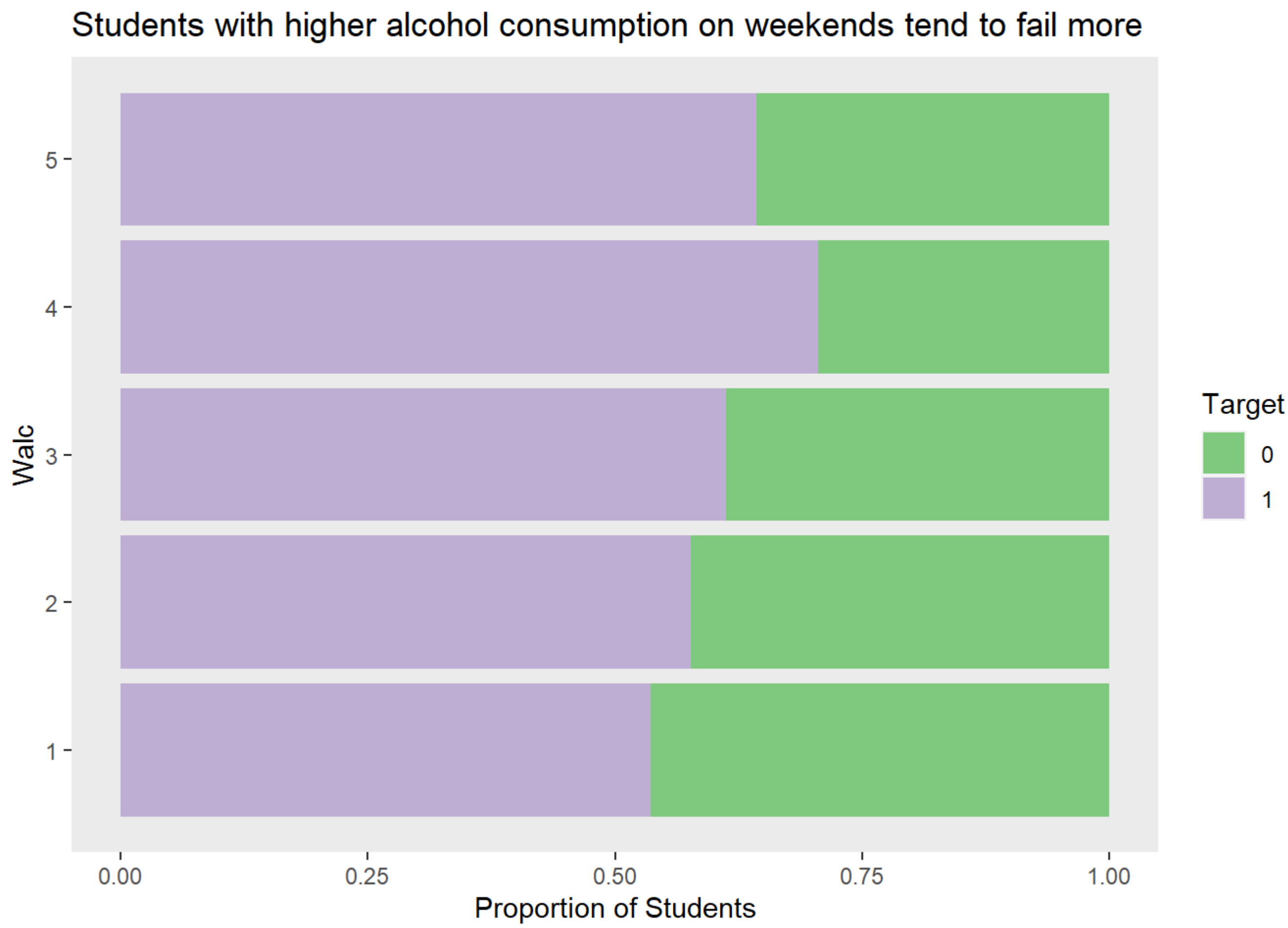

```
student_df %>%
  group_by(school, Target) %>%
  count() %>%
  ggplot() +
  aes(x=school, y=n, fill = Target) +
  geom_bar(stat = "identity") +
  scale_fill_brewer(palette = "Accent") +
  coord_flip() +
   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ylab("Number of Students") +
  ggtitle("Both schools have around the same proportion of students who fail")
```



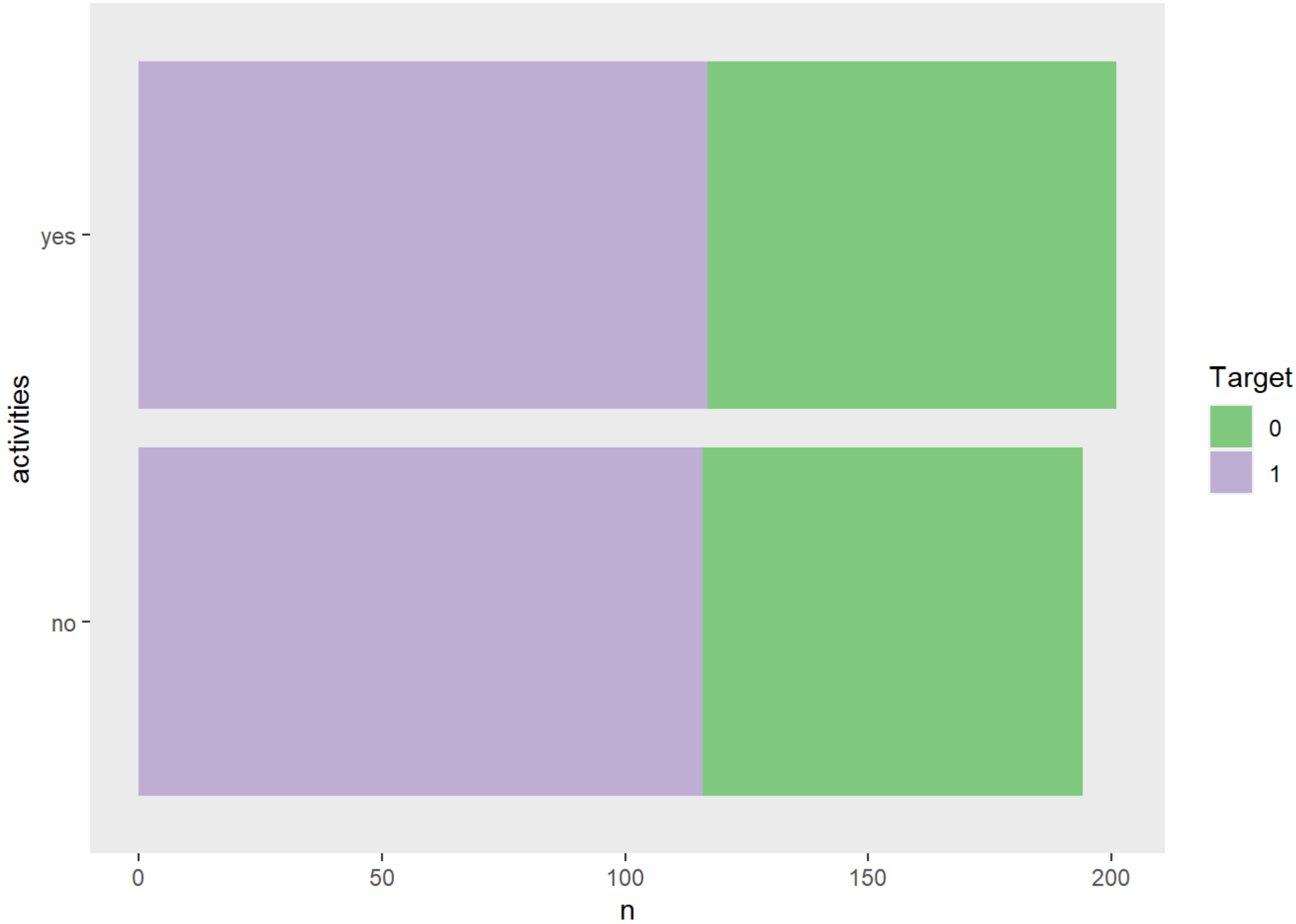

```
student_df %>%
  group_by(Fjob, Target) %>%
  count() %>%
  ggplot() +
  aes(x=Fjob, y=n, fill = Target) +
  geom_bar(stat = "identity") +
  scale_fill_brewer(palette = "Accent") +
  coord_flip() +
   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ylab("Number of Students") +
  ggtitle("Students with Fathers who work in Services struggle with the class")
```



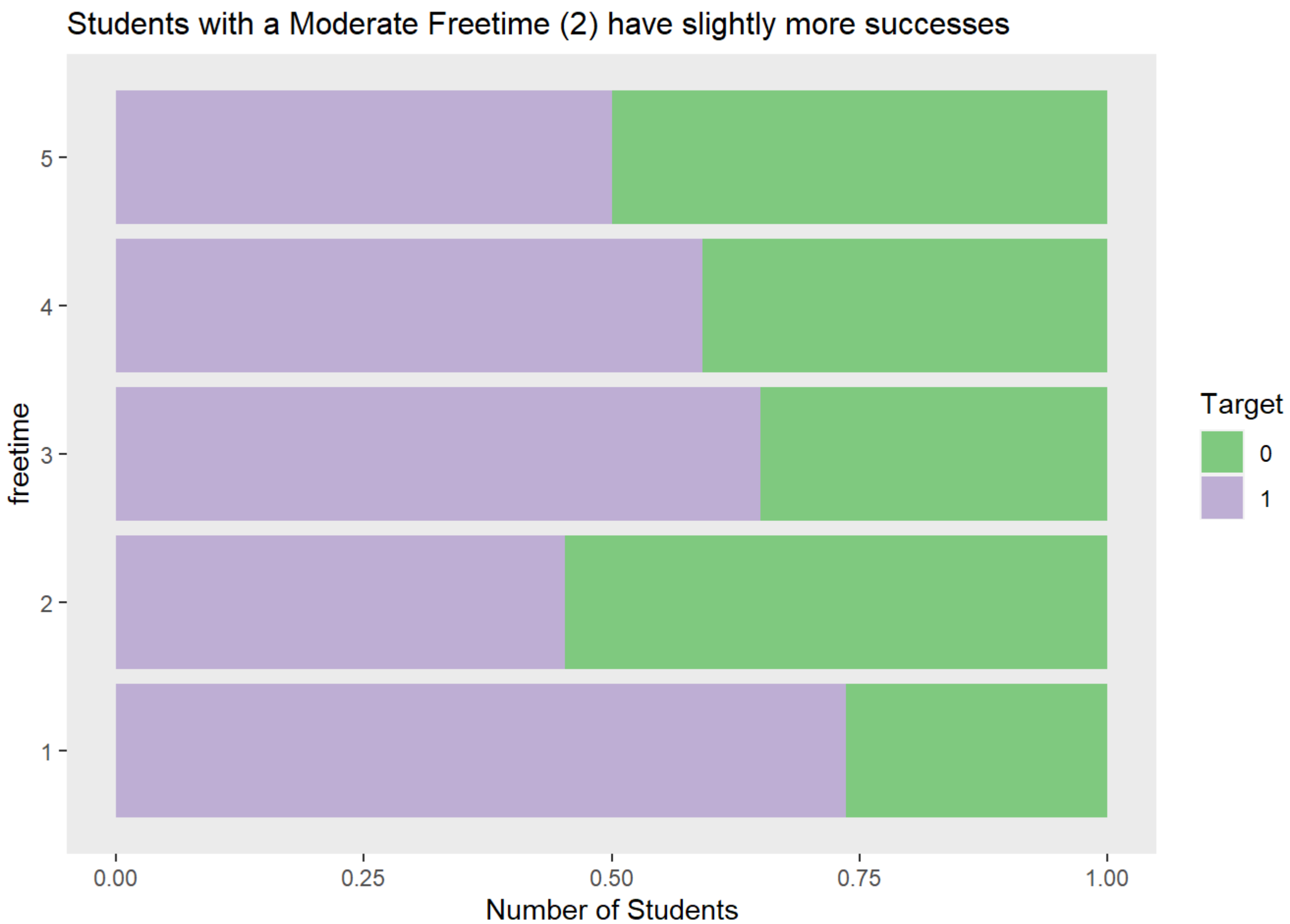

```
student_df %>%
  group_by(Walc, Target) %>%
  count() %>%
  ggplot() +
  aes(x=Walc, y=n, fill = Target) +
  geom_bar(position = "fill",stat = "identity") +
  scale_fill_brewer(palette = "Accent") +
  coord_flip() +
   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +
  ylab("Proportion of Students") +
  ggtitle("Students with higher alcohol consumption on weekends tend to fail more")
```



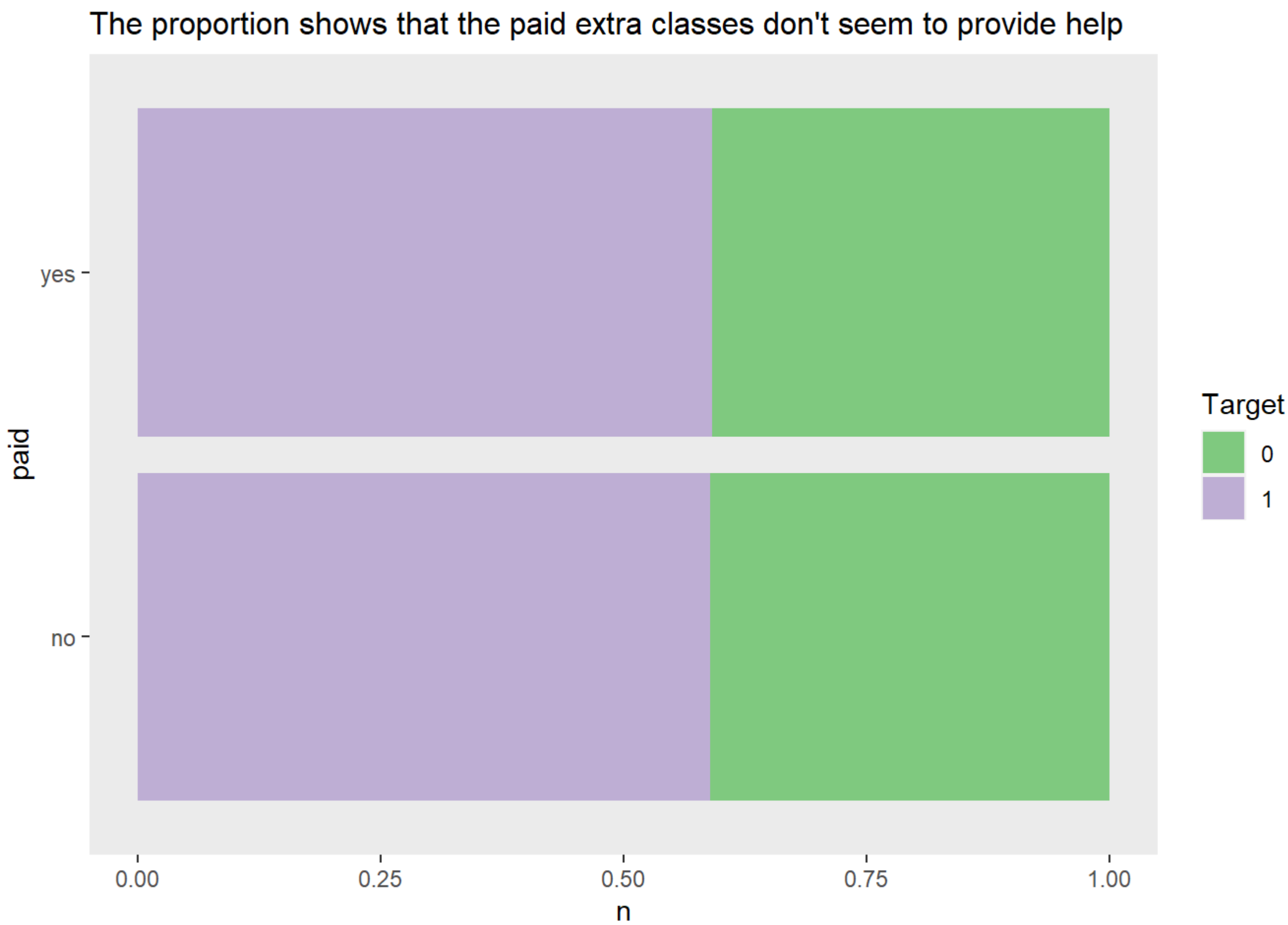

```
student_df %>%
  group_by(activities, Target) %>%
  count() %>%
  ggplot() +
  aes(x=activities, y=n, fill = Target) +
  geom_bar(stat = "identity") +
  scale_fill_brewer(palette = "Accent") +
 theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())  +
  coord_flip()
```




```
student_df %>%
  group_by( freetime, Target) %>%
  count() %>%
  ggplot() +
  aes(fill = Target, y = n, x=freetime) +
  geom_bar(position = "fill", stat="identity")  +
 ggtitle("Students with a Moderate Freetime (2) have slightly more successes") +
  coord_flip() +
   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), plot.title = el
ement_text(size = 12)) +
  scale_fill_brewer(palette = "Accent") +
  ylab("Number of Students")
```




```
student_df %>%
  group_by(paid, Target) %>%
  count() %>%
  ggplot() +
  aes(fill = Target, y = n, x=paid) +
  geom_bar(position = "fill", stat="identity")  +
  ggtitle("The proportion shows that the paid extra classes don't seem to provide help") +
  coord_flip() +
   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), plot.title = el
ement_text(size = 12)) +
  scale_fill_brewer(palette = "Accent")
```




```
student_df <- student_df %>%
  select(-c(G3, G2, G1, paid))
```

**Modeling**

Updating variable types for model performance

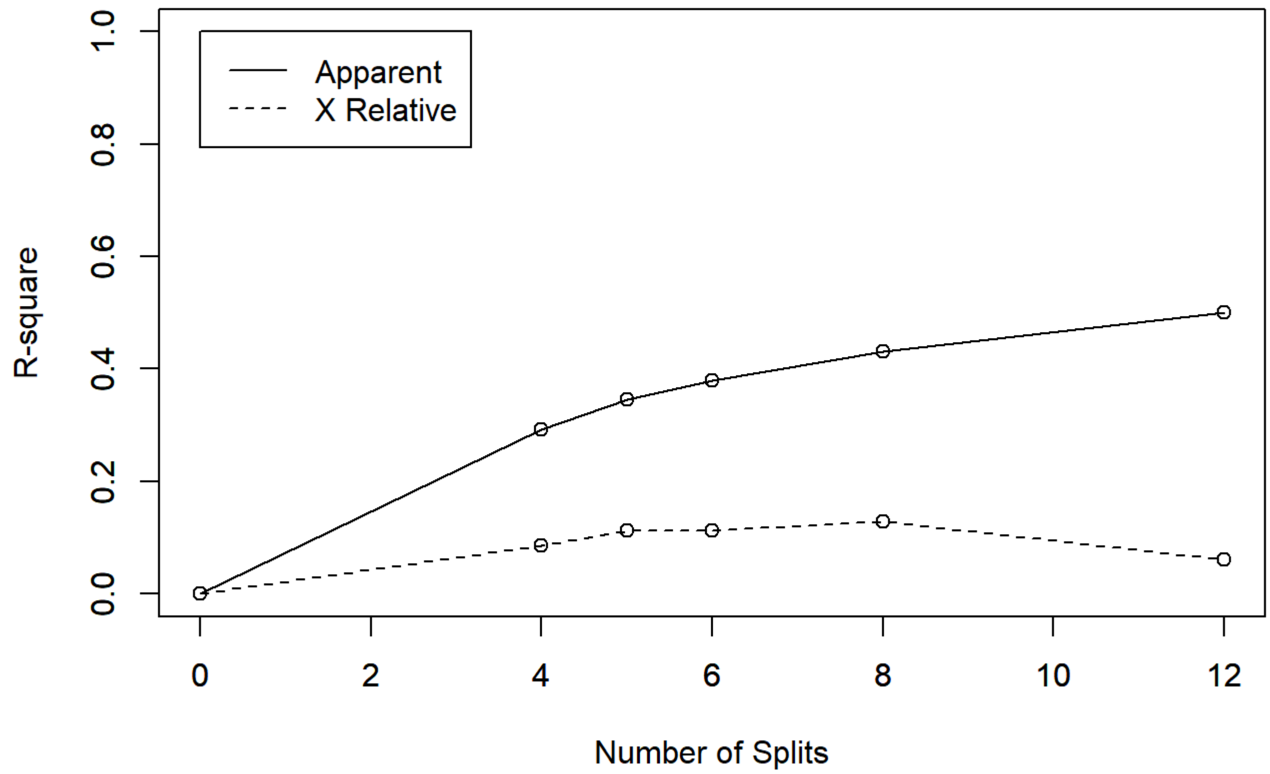
```
student_df <- student_df %>%
  mutate(school = as.factor(school)) %>%
  mutate(sex = as.factor(sex)) %>%
  mutate(address = as.factor(address)) %>%
  mutate(famsize = as.factor(famsize)) %>%
  mutate(Pstatus = as.factor(Pstatus)) %>%
  mutate(Medu = factor(Medu, ordered = TRUE, levels = c(0, 1, 2, 3, 4))) %>%
  mutate(Fedu = factor(Fedu, ordered = TRUE, levels = c(0, 1, 2, 3, 4))) %>%
  mutate(schoolsup = as.factor(schoolsup)) %>%
  mutate(famsup = as.factor(famsup)) %>%
  mutate(activities = as.factor(activities)) %>%
  mutate(nursery = as.factor(nursery)) %>%
  mutate(higher = as.factor(higher)) %>%
  mutate(internet = as.factor(internet))  %>%
  mutate(romantic = as.factor(romantic)) %>%
  mutate(famrel = factor(famrel, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5))) %>%
  mutate(freetime = factor(freetime, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5))) %>%
  mutate(goout = factor(goout, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5))) %>%
  mutate(Dalc = factor(Dalc, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5))) %>%
  mutate(Walc = factor(Walc, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5))) %>%
  mutate(health = factor(health, ordered = TRUE, levels = c(0, 1, 2, 3, 4, 5)))
```

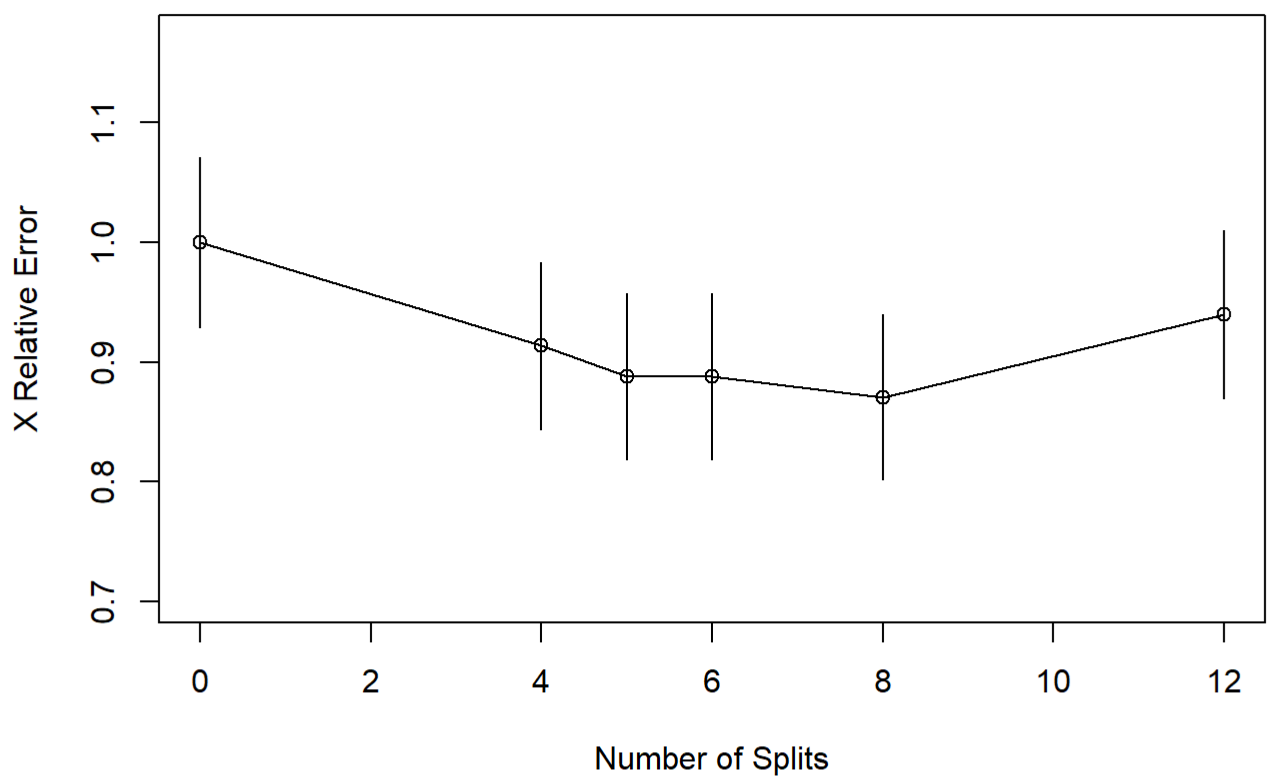
```
sample <- sample(c(TRUE, FALSE), nrow(student_df), replace=TRUE, prob=c(0.7,0.3))
train_student  <- student_df[sample, ]
test_student   <- student_df[!sample, ]
```

*Decision Tree Unpruned*

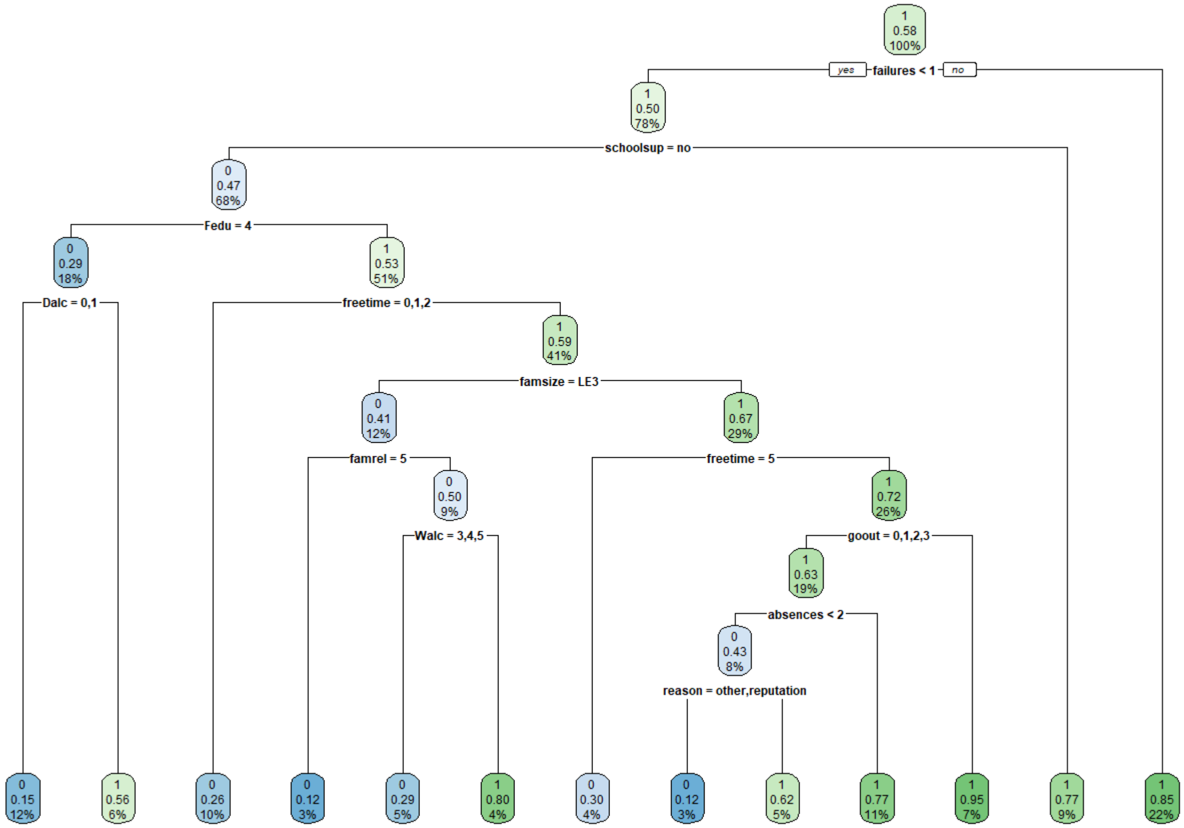
```
tree_model_unprune <- rpart(Target ~., data = train_student
, method = 'class'
, model = T
)
rsq.rpart(tree_model_unprune)
```

```
##
## Classification tree:
## rpart(formula = Target ~ ., data = train_student, method = "class",
##     model = T)
##
## Variables actually used in tree construction:
##  [1] absences  Dalc      failures  famrel    famsize   Fedu      freetime
##  [8] goout     reason    schoolsup Walc
##
## Root node error: 116/276 = 0.42029
##
## n= 276
##
##         CP nsplit rel error  xerror     xstd
## 1 0.056034      0   1.00000 1.00000 0.070693
## 2 0.051724      4   0.70690 0.91379 0.069657
## 3 0.034483      5   0.65517 0.88793 0.069267
## 4 0.025862      6   0.62069 0.88793 0.069267
## 5 0.017241      8   0.56897 0.87069 0.068987
## 6 0.010000     12   0.50000 0.93966 0.070010
```


Apparent
X Relative
R-square
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
12
Number of Splits


X Relative Error
0.7
0.8
0.9
1.0
1.1
0
2
4
6
8
10
12
Number of Splits

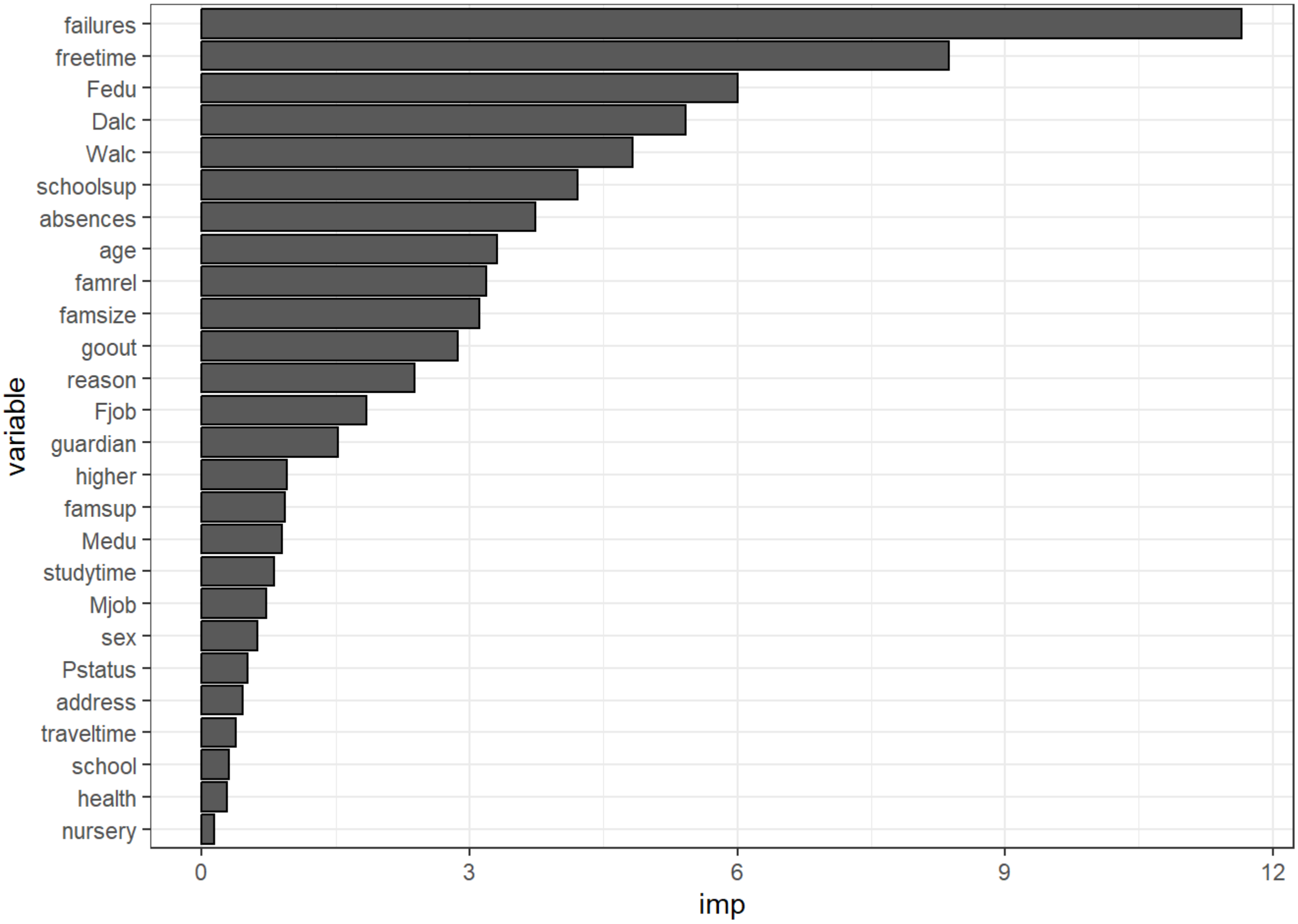
```
rpart.plot(tree_model_unprune)
```




```
preds_unprune <- predict(tree_model_unprune, test_student, type="class")
confusionMatrix(data = preds_unprune, reference = test_student$Target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 22 21
##          1 24 52
##
##                Accuracy : 0.6218
##                  95% CI : (0.5284, 0.7091)
##     No Information Rate : 0.6134
##     P-Value [Acc > NIR] : 0.4653
##
##                   Kappa : 0.1929
##
##  Mcnemar's Test P-Value : 0.7656
##
##             Sensitivity : 0.4783
##             Specificity : 0.7123
##          Pos Pred Value : 0.5116
##          Neg Pred Value : 0.6842
##              Prevalence : 0.3866
##          Detection Rate : 0.1849
##    Detection Prevalence : 0.3613
##       Balanced Accuracy : 0.5953
##
##        'Positive' Class : 0
##
```

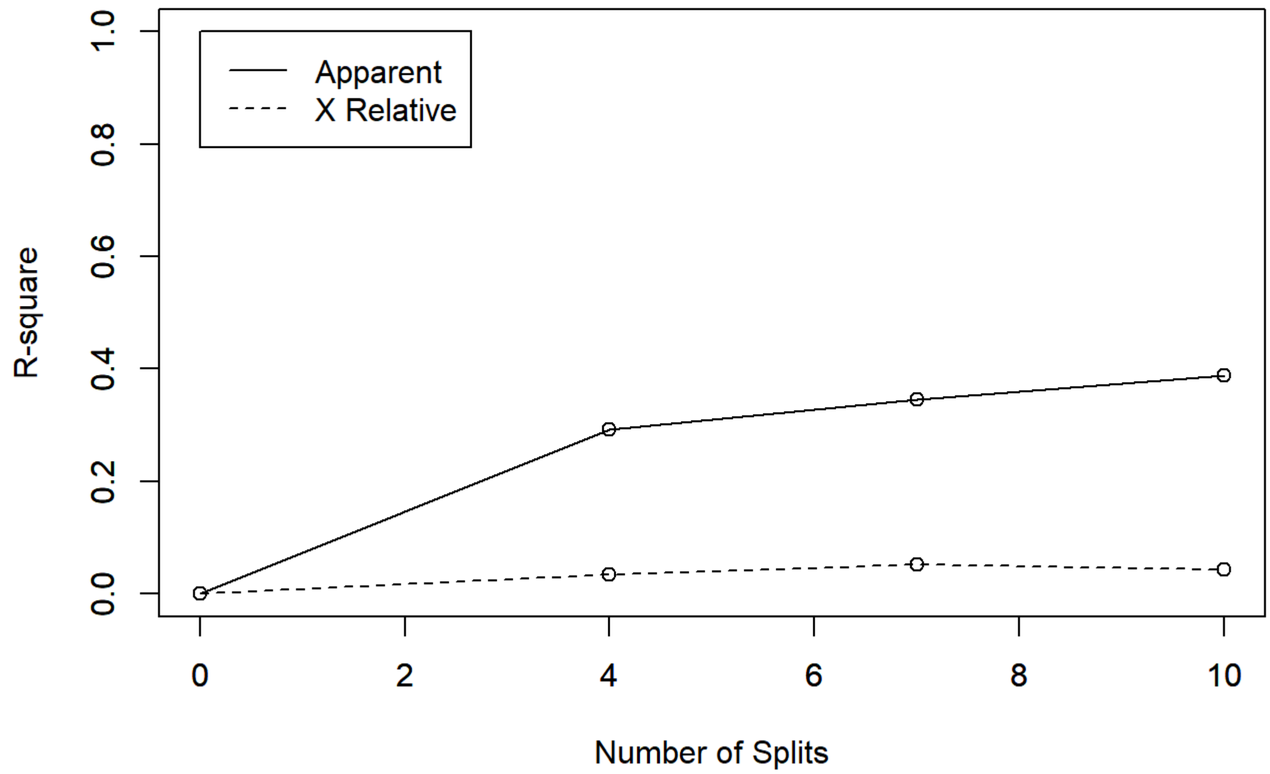
```
df <- data.frame(imp = tree_model_unprune$variable.importance)

 df %>%
  tibble::rownames_to_column() %>%
  dplyr::rename("variable" = rowname) %>%
  dplyr::arrange(imp) %>%
  dplyr::mutate(variable = forcats::fct_inorder(variable)) %>%
  ggplot() +
  geom_col(aes(x=variable, y = imp),
           col = "black", show.legend = F) +
  coord_flip() +
  scale_fill_grey() +
  theme_bw()
```

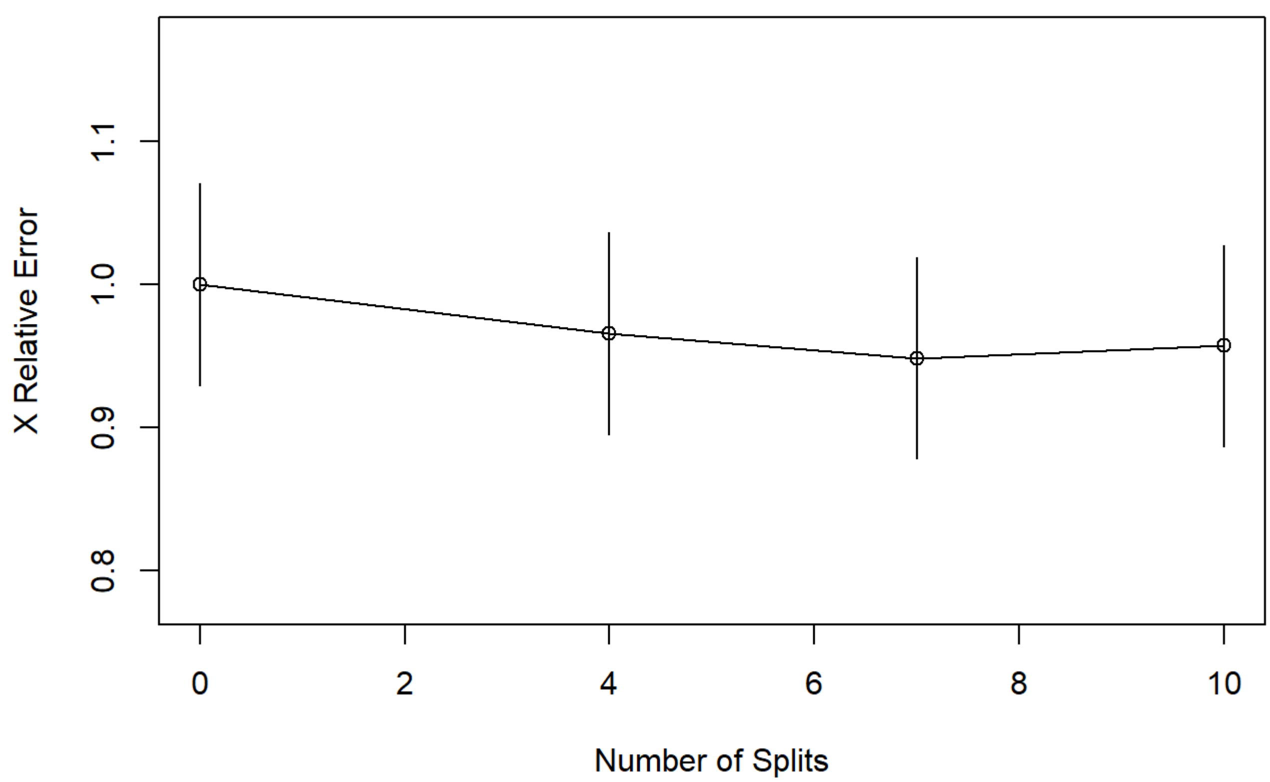



*Pruned-1*

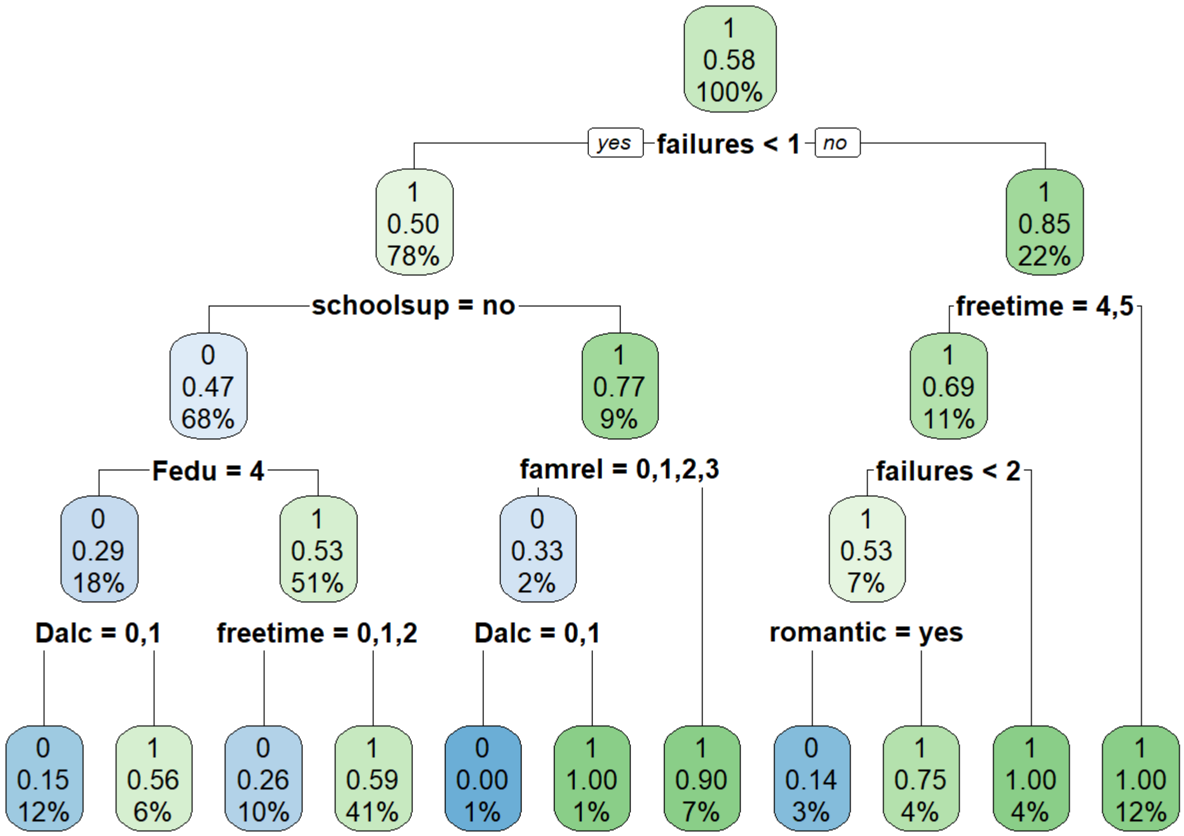
```
#last model to remove to the prune to 0 to remove some potential stems if above 0 and adding som
e cross validation to the model and focusing on percentage rather than missing values
tree_model_prune_1 <- rpart(Target ~., data = train_student
, method = 'class'
,control = rpart.control(minsplit = 6,  maxdepth = 4)
, model = T
)
rsq.rpart(tree_model_prune_1)
```

```
##
## Classification tree:
## rpart(formula = Target ~ ., data = train_student, method = "class",
##     model = T, control = rpart.control(minsplit = 6, maxdepth = 4))
##
## Variables actually used in tree construction:
## [1] Dalc      failures  famrel    Fedu      freetime  romantic  schoolsup
##
## Root node error: 116/276 = 0.42029
##
## n= 276
##
##         CP nsplit rel error  xerror     xstd
## 1 0.056034      0   1.00000 1.00000 0.070693
## 2 0.017241      4   0.70690 0.96552 0.070326
## 3 0.014368      7   0.65517 0.94828 0.070119
## 4 0.010000     10   0.61207 0.95690 0.070225
```


Apparent
X Relative
R-square
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
Number of Splits


X Relative Error
0.8
0.9
1.0
1.1
0
2
4
6
8
10
Number of Splits

```
rpart.plot(tree_model_prune_1)
```




```
preds_prune_1 <- predict(tree_model_prune_1, test_student, type="class")
confusionMatrix(data = preds_prune_1, reference = test_student$Target)
```

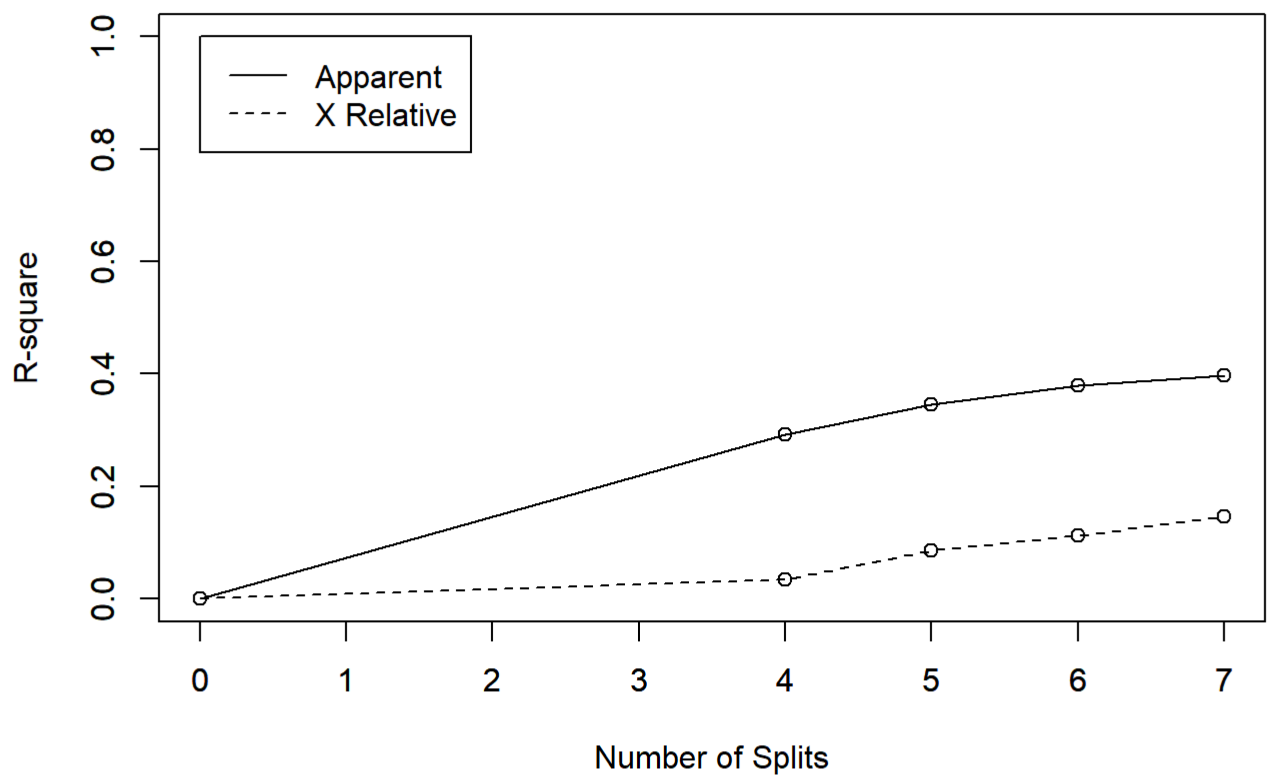
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 15 18
##          1 31 55
##
##                Accuracy : 0.5882
##                  95% CI : (0.4943, 0.6776)
##     No Information Rate : 0.6134
##     P-Value [Acc > NIR] : 0.74623
##
##                   Kappa : 0.0839
##
##  Mcnemar's Test P-Value : 0.08648
##
##             Sensitivity : 0.3261
##             Specificity : 0.7534
##          Pos Pred Value : 0.4545
##          Neg Pred Value : 0.6395
##              Prevalence : 0.3866
##          Detection Rate : 0.1261
##    Detection Prevalence : 0.2773
##       Balanced Accuracy : 0.5398
##
##        'Positive' Class : 0
##
```

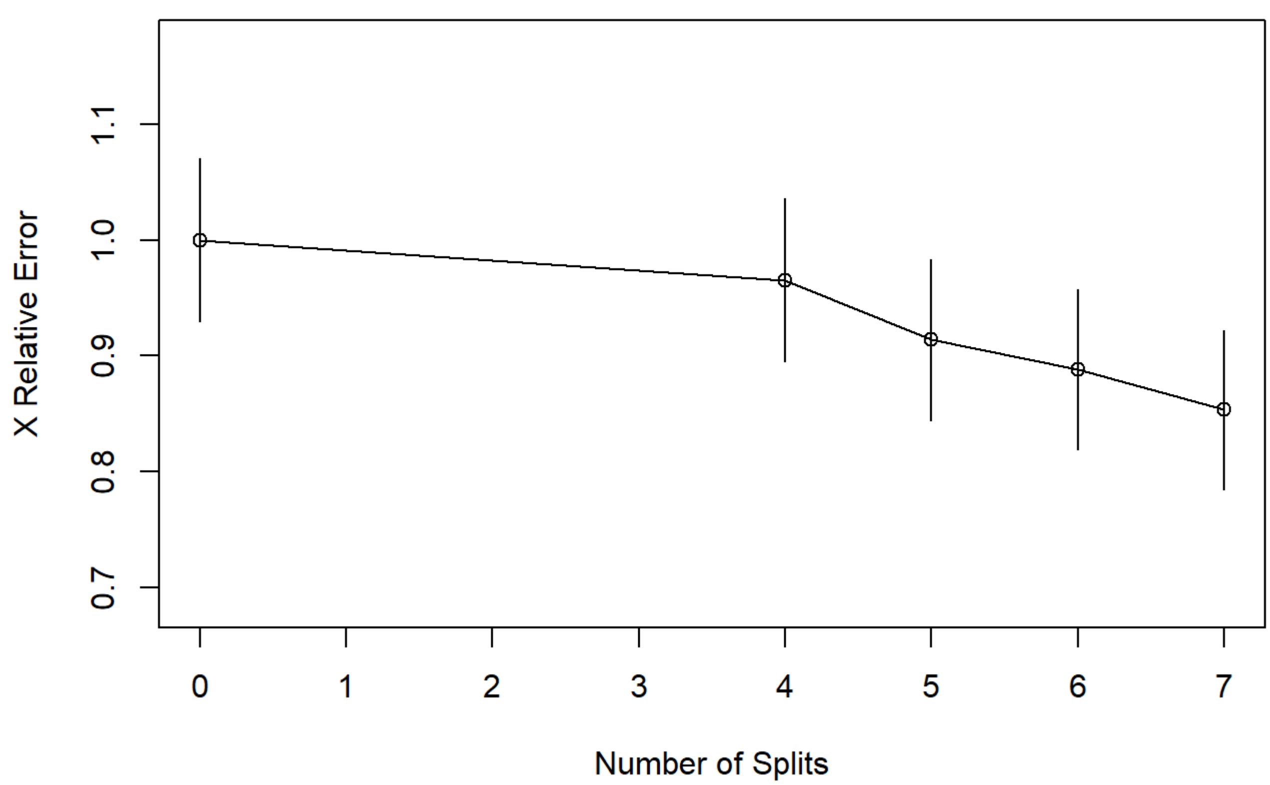
*Pruned-2*

```
train_student_1 <- train_student %>% select(c("failures", "Target", "freetime", "Fedu", "Dalc",
"Walc",  "schoolsup", "absences", "age", "famrel", "famsize"))

test_student_1 <- train_student %>% select(c("failures", "Target", "freetime", "Fedu", "Dalc",
"Walc",  "schoolsup", "absences", "age", "famrel", "famsize"))
```

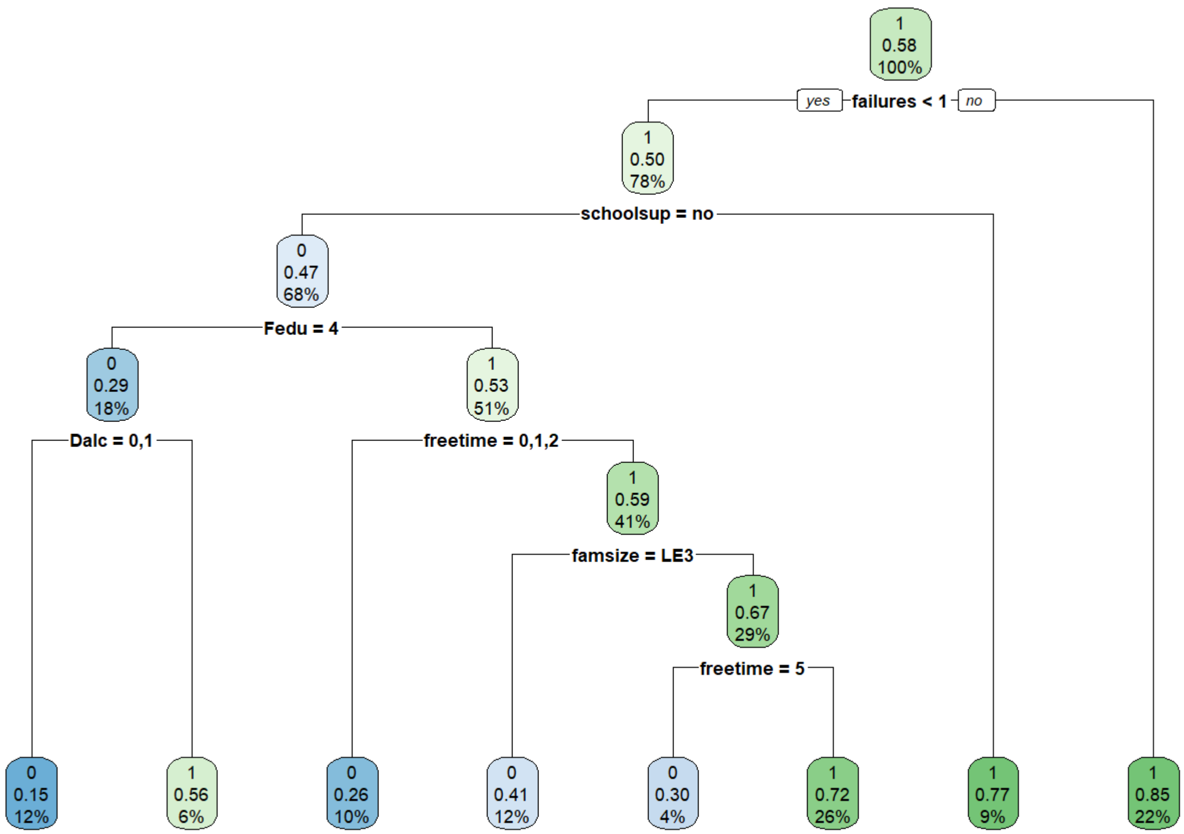
```
#last model to remove to the prune to 0 to remove some potential stems if above 0 and adding som
e cross validation to the model and focusing on percentage rather than missing values
tree_model_prune_2 <- rpart(Target ~., data = train_student_1
, method = 'class'
, control = rpart.control(minbucket = 10,  maxcompete = 2, xval = 10)
, model = T
)
rsq.rpart(tree_model_prune_2)
```

```
##
## Classification tree:
## rpart(formula = Target ~ ., data = train_student_1, method = "class",
##     model = T, control = rpart.control(minbucket = 10, maxcompete = 2,
##         xval = 10))
##
## Variables actually used in tree construction:
## [1] Dalc      failures  famsize   Fedu      freetime  schoolsup
##
## Root node error: 116/276 = 0.42029
##
## n= 276
##
##         CP nsplit rel error  xerror     xstd
## 1 0.056034      0   1.00000 1.00000 0.070693
## 2 0.051724      4   0.70690 0.96552 0.070326
## 3 0.034483      5   0.65517 0.91379 0.069657
## 4 0.017241      6   0.62069 0.88793 0.069267
## 5 0.010000      7   0.60345 0.85345 0.068690
```


Apparent
X Relative
R-square
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
7
Number of Splits


X Relative Error
0.7
0.8
0.9
1.0
1.1
0
1
2
3
4
5
6
7
Number of Splits

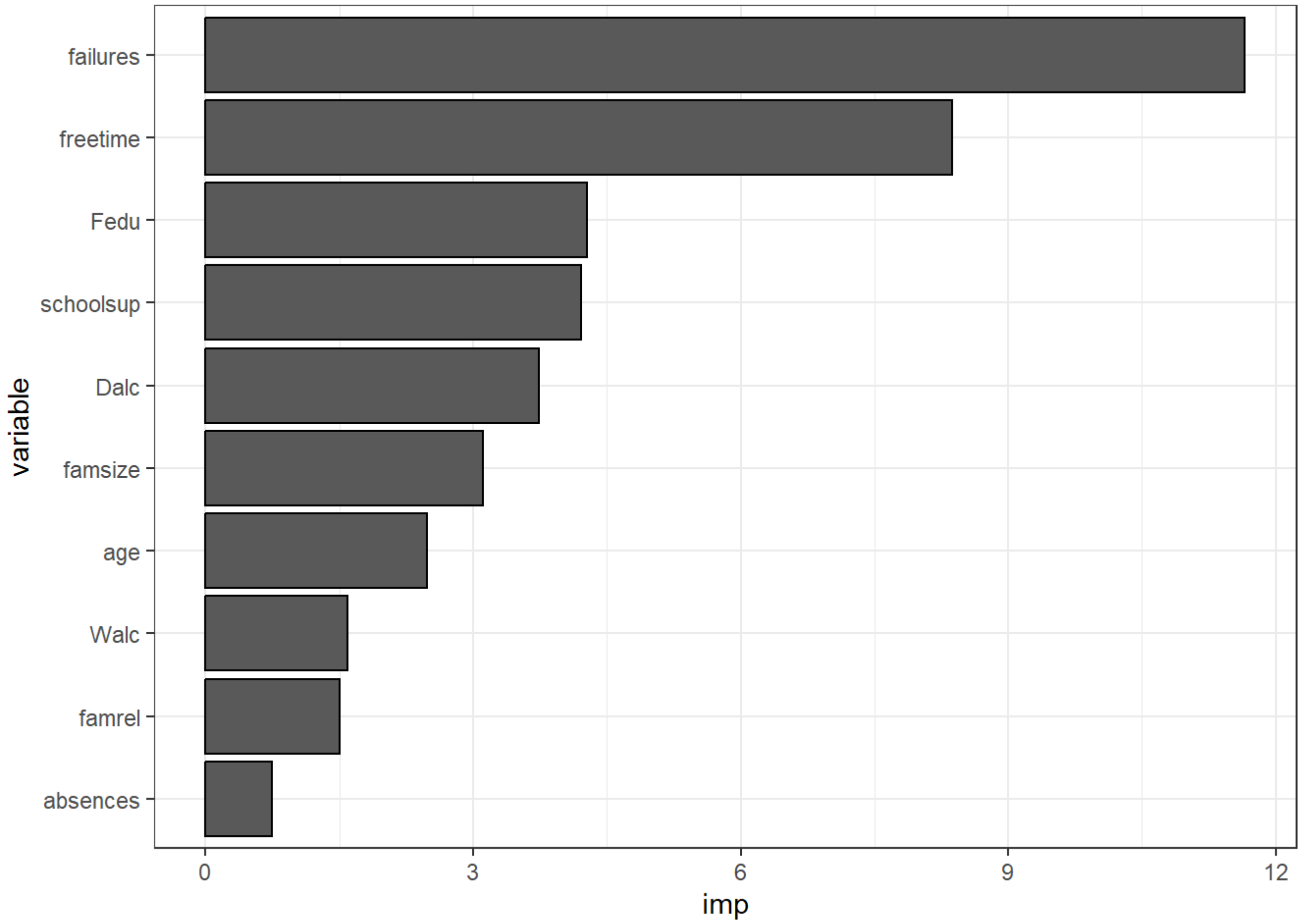
```
rpart.plot(tree_model_prune_2)
```




```
preds_prune_2 <- predict(tree_model_prune_2, test_student_1, type="class")
confusionMatrix(data = preds_prune_2, reference = test_student_1$Target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  74  28
##          1  42 132
##
##                Accuracy : 0.7464
##                  95% CI : (0.6908, 0.7966)
##     No Information Rate : 0.5797
##     P-Value [Acc > NIR] : 5.695e-09
##
##                   Kappa : 0.4707
##
##  Mcnemar's Test P-Value : 0.1202
##
##             Sensitivity : 0.6379
##             Specificity : 0.8250
##          Pos Pred Value : 0.7255
##          Neg Pred Value : 0.7586
##              Prevalence : 0.4203
##          Detection Rate : 0.2681
##    Detection Prevalence : 0.3696
##       Balanced Accuracy : 0.7315
##
##        'Positive' Class : 0
##
```

```
df <- data.frame(imp = tree_model_prune_2$variable.importance)

 df %>%
  tibble::rownames_to_column() %>%
  dplyr::rename("variable" = rowname) %>%
  dplyr::arrange(imp) %>%
  dplyr::mutate(variable = forcats::fct_inorder(variable)) %>%
  ggplot() +
  geom_col(aes(x=variable, y = imp),
           col = "black", show.legend = F) +
  coord_flip() +
  scale_fill_grey() +
  theme_bw()
```




*SVM*

```
# SVM model 1 - Tuning with a polynomial kernel

polynomialModel1 <- svm(Target ~ ., data = train_student_1, kernel = "polynomial", cost = 0.15,
scale = FALSE)

# Prediction and accuracy


polymodel1Pred <- predict(polynomialModel1, newdata = test_student_1)

polymodel1CM <- confusionMatrix(table(polymodel1Pred, test_student_1$Target))
polymodel1CM
```

```
## Confusion Matrix and Statistics
##
##
## polymodel1Pred   0   1
##              0  89  23
##              1  27 137
##
##                Accuracy : 0.8188
##                  95% CI : (0.7682, 0.8624)
##     No Information Rate : 0.5797
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6265
##
##  Mcnemar's Test P-Value : 0.6714
##
##             Sensitivity : 0.7672
##             Specificity : 0.8562
##          Pos Pred Value : 0.7946
##          Neg Pred Value : 0.8354
##              Prevalence : 0.4203
##          Detection Rate : 0.3225
##    Detection Prevalence : 0.4058
##       Balanced Accuracy : 0.8117
##
##        'Positive' Class : 0
##
```

```
# The accuracy of the model is 60.5% and the misclassification rate is 39.5%
```

```
# SVM model 2 - Tuning with a radial kernel

radialModel2 <- svm(Target ~ ., data = train_student_1, kernel = "radial", cost = 0.95, scale =
FALSE)

# Prediction and accuracy

radmodel2Pred <- predict(radialModel2, newdata = test_student_1)

radmodel2CM <- confusionMatrix(table(radmodel2Pred, test_student_1$Target))
radmodel2CM
```

```
## Confusion Matrix and Statistics
## 
##             
## radmodel2Pred   0   1
##             0  64  22
##             1  52 138
##                                           
##                Accuracy : 0.7319          
##                  95% CI : (0.6755, 0.7832)
##     No Information Rate : 0.5797          
##     P-Value [Acc > NIR] : 1.065e-07       
##                                           
##                   Kappa : 0.4295          
##                                           
##  Mcnemar's Test P-Value : 0.0007485       
##                                           
##             Sensitivity : 0.5517          
##             Specificity : 0.8625          
##          Pos Pred Value : 0.7442          
##          Neg Pred Value : 0.7263          
##              Prevalence : 0.4203          
##          Detection Rate : 0.2319          
##    Detection Prevalence : 0.3116          
##       Balanced Accuracy : 0.7071          
##                                           
##        'Positive' Class : 0               
## 
```

```
# The accuracy of the model is 63.87% and the misclassification rate is 36.13%
```

```
# SVM model 3 - Tuning with a sigmoid kernel

sigmoidModel3 <- svm(Target ~ ., data = train_student, kernel = "sigmoid", cost = 0.95, scale =
FALSE)

# Prediction and accuracy

sigmodel3Pred <- predict(sigmoidModel3, newdata = test_student)

sigmodel3CM <- confusionMatrix(table(sigmodel3Pred, test_student$Target))
sigmodel3CM
```

```
## Confusion Matrix and Statistics
##
##
## sigmodel3Pred  0  1
##             0  0  0
##             1 46 73
##
##                Accuracy : 0.6134
##                  95% CI : (0.5198, 0.7013)
##     No Information Rate : 0.6134
##     P-Value [Acc > NIR] : 0.5403
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 3.247e-11
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.6134
##              Prevalence : 0.3866
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

```
# The accuracy of the model is 61.34% and the misclassification rate is 38.66%
```

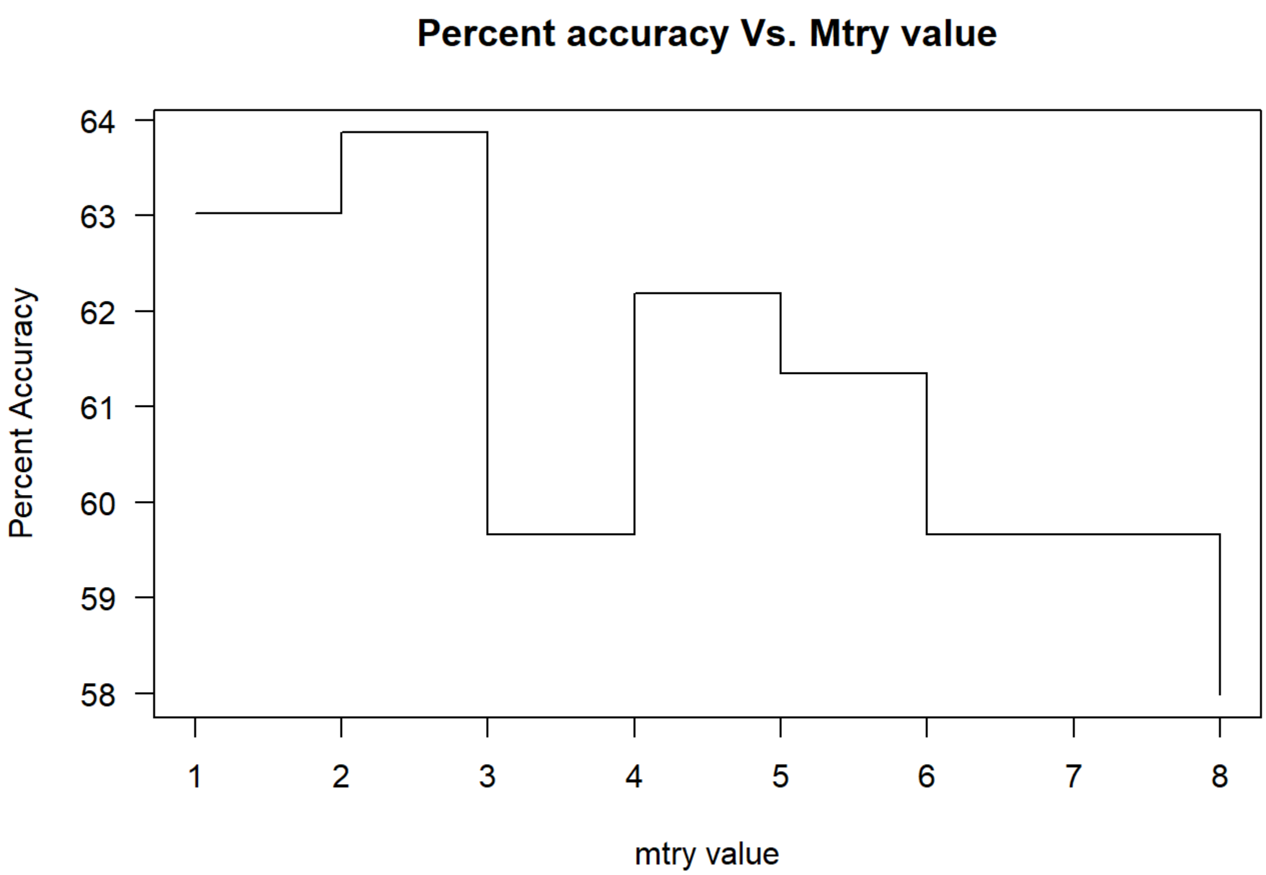
### *Random Forest*

```
# Random Forest model 1 - default parameters

rfModel1 <- randomForest(Target ~ ., data = train_student_1, importance = TRUE)

# Prediction and accuracy

rfmodel1Pred <- predict(rfModel1, newdata = test_student_1)

rfmodel1CM <- confusionMatrix(table(rfmodel1Pred, test_student_1$Target))
rfmodel1CM
```

```
## Confusion Matrix and Statistics
##
##
## rfmodel1Pred   0   1
##            0 115   2
##            1   1 158
##
##                Accuracy : 0.9891
##                  95% CI : (0.9686, 0.9978)
##     No Information Rate : 0.5797
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9777
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9914
##             Specificity : 0.9875
##          Pos Pred Value : 0.9829
##          Neg Pred Value : 0.9937
##              Prevalence : 0.4203
##          Detection Rate : 0.4167
##    Detection Prevalence : 0.4239
##       Balanced Accuracy : 0.9894
##
##        'Positive' Class : 0
##
```

```
# The accuracy of the model is 59.66%
```

```
# Identifying the most appropriate mtry for model 3

# Pre-processing
trainwithoutTarget <- subset(train_student, select = -c(Target))

trainwithTarget <- train_student$Target

testwithoutTarget <- subset(test_student, select = -c(Target))

testwithTarget <- test_student$Target


a = c()
i = 5
for (i in 3:10) {
  idealrfModel <- randomForest(Target ~ ., data = train_student, ntree = 500, mtry = i, importan
ce = TRUE)
  idealrfmodelPred <- predict(idealrfModel, testwithoutTarget, type = "class")
  a[i-2] = mean(idealrfmodelPred == testwithTarget)
}

plot(a*100, type='s', las=1, ylab = "Percent Accuracy", xlab = "mtry value", main = "Percent acc
uracy Vs. Mtry value")
```



```
# The accuracy of mtry increased from 1 to in between 2 and 5. The next increase was in between
7 and 8. Therefore, a mtry value of 4 will be used
```

```
# Random Forest model 3 - Tuning mtry = 4


rfModel3 <- randomForest(Target ~ ., data = train_student_1, ntree = 500, mtry = 4, importance =
TRUE)

# Prediction and accuracy


rfmodel3Pred <- predict(rfModel3, newdata = test_student_1)

rfmodel3CM <- confusionMatrix(table(rfmodel3Pred, test_student_1$Target))
rfmodel3CM
```

```
## Confusion Matrix and Statistics
##
##
## rfmodel3Pred   0   1
##            0 115   1
##            1   1 159
##
##                Accuracy : 0.9928
##                  95% CI : (0.9741, 0.9991)
##     No Information Rate : 0.5797
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9851
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9914
##             Specificity : 0.9938
##          Pos Pred Value : 0.9914
##          Neg Pred Value : 0.9938
##              Prevalence : 0.4203
##          Detection Rate : 0.4167
##    Detection Prevalence : 0.4203
##       Balanced Accuracy : 0.9926
##
##        'Positive' Class : 0
##
```

```
# The accuracy of the model is 61.34%
```

### *Logistic Regression*

```
log_1 <- glm(Target~. , family= binomial, data=test_student_1)
summary(log_1)
```

```
##
## Call:
## glm(formula = Target ~ ., family = binomial, data = test_student_1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1560  -0.9060   0.2839   0.8870   2.1301
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.18636  176.56193   0.001 0.999158
## failures       1.37650    0.36597   3.761 0.000169 ***
## freetime.L    -1.17747    0.65343  -1.802 0.071550 .
## freetime.Q     0.41363    0.55495   0.745 0.456060
## freetime.C    -1.38893    0.44043  -3.154 0.001613 **
## freetime^4     0.80258    0.30510   2.631 0.008524 **
## Fedu.L        -9.84347  558.29609  -0.018 0.985933
## Fedu.Q         7.42671  471.84633   0.016 0.987442
## Fedu.C        -4.87438  279.14817  -0.017 0.986068
## Fedu^4         1.67696  105.50843   0.016 0.987319
## Dalc.L        -0.59442    0.91504  -0.650 0.515943
## Dalc.Q        -0.76375    0.70588  -1.082 0.279256
## Dalc.C        -0.19206    0.72679  -0.264 0.791580
## Dalc^4        -0.12978    0.70220  -0.185 0.853367
## Walc.L         0.80760    0.73237   1.103 0.270153
## Walc.Q         0.18474    0.55029   0.336 0.737091
## Walc.C         0.05596    0.44279   0.126 0.899430
## Walc^4        -0.57787    0.37159  -1.555 0.119911
## schoolsupyes   1.80990    0.54057   3.348 0.000813 ***
## absences       0.03500    0.02817   1.242 0.214121
## age            0.14541    0.13151   1.106 0.268865
## famrel.L       0.89688    0.69419   1.292 0.196361
## famrel.Q      -0.03045    0.60494  -0.050 0.959851
## famrel.C      -0.77025    0.59828  -1.287 0.197940
## famrel^4       0.61239    0.48418   1.265 0.205942
## famsizeLE3    -0.63690    0.33690  -1.890 0.058699 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 375.57  on 275  degrees of freedom
## Residual deviance: 288.51  on 250  degrees of freedom
## AIC: 340.51
##
## Number of Fisher Scoring iterations: 13
```

```
preds_log_1 <- predict(log_1, test_student_1, type="response")
preds_log_1.classes <- ifelse(preds_log_1 > 0.6, 1, 0)
result_1 <- data.frame(preds_log_1.classes, test_student_1$Target)
result_1$Correct <- ifelse(result_1$preds_log_1.classes == result_1$test_student_1.Target, 1, 0)
sum(result_1$Correct)/nrow(result_1)
```

```
## [1] 0.7173913
```

```
log_2 <- glm(Target~ Fedu + failures + schoolsup + freetime + Walc, family= binomial, data=train
_student_1)
summary(log_2)
```

```
##
## Call:
## glm(formula = Target ~ Fedu + failures + schoolsup + freetime +
##     Walc, family = binomial, data = train_student_1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0559  -0.9132   0.3355   0.9028   2.1288
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.98978  176.54883   0.017  0.98649
## Fedu.L        -9.96308  558.29601  -0.018  0.98576
## Fedu.Q         7.54825  471.84627   0.016  0.98724
## Fedu.C        -4.97692  279.14812  -0.018  0.98578
## Fedu^4         1.64632  105.50838   0.016  0.98755
## failures       1.54000    0.36003   4.277 1.89e-05 ***
## schoolsupyes   1.72265    0.52473   3.283  0.00103 **
## freetime.L    -1.12757    0.60401  -1.867  0.06193 .
## freetime.Q     0.31439    0.52722   0.596  0.55096
## freetime.C    -1.33240    0.41824  -3.186  0.00144 **
## freetime^4     0.81827    0.29107   2.811  0.00493 **
## Walc.L         0.61929    0.40140   1.543  0.12288
## Walc.Q        -0.15759    0.38129  -0.413  0.67938
## Walc.C         0.00508    0.36770   0.014  0.98898
## Walc^4        -0.53223    0.34335  -1.550  0.12112
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 375.57  on 275  degrees of freedom
## Residual deviance: 300.81  on 261  degrees of freedom
## AIC: 330.81
##
## Number of Fisher Scoring iterations: 13
```

```
preds_log_2 <- predict(log_2, test_student_1, type="response")
preds_log_2.classes <- ifelse(preds_log_2 > 0.6, 1, 0)
mean(preds_log_2.classes == test_student_1$Target)
```

```
## [1] 0.6992754
```

```
log_3 <- glm(Target~ failures + schoolsup, family= binomial, data=test_student_1)
summary(log_3)
```

```
##
## Call:
## glm(formula = Target ~ failures + schoolsup, family = binomial,
##     data = test_student_1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7582  -1.1085   0.3476   1.2479   1.2479
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.1641     0.1445  -1.136  0.25617
## failures       1.4701     0.3438   4.276  1.9e-05 ***
## schoolsupyes   1.4418     0.4795   3.007  0.00264 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 375.57  on 275  degrees of freedom
## Residual deviance: 330.77  on 273  degrees of freedom
## AIC: 336.77
##
## Number of Fisher Scoring iterations: 5
```

```
preds_log_3 <- predict(log_3, test_student_1, type="response")
preds_log_3.classes <- ifelse(preds_log_3 > 0.6, 1, 0)
mean(preds_log_3.classes == test_student_1$Target)
```

```
## [1] 0.6268116
```

*Naive Bayes*

```
naivebayes_model <-naiveBayes(Target~., data=train_student, na.action = na.pass)
```

```
nb_Pred <- predict(naivebayes_model, test_student)
confusionMatrix(data = nb_Pred, reference = test_student$Target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 35 39
##          1 11 34
##
##                Accuracy : 0.5798
##                  95% CI : (0.4859, 0.6697)
##     No Information Rate : 0.6134
##     P-Value [Acc > NIR] : 0.8020630
##
##                   Kappa : 0.2037
##
##  Mcnemar's Test P-Value : 0.0001343
##
##             Sensitivity : 0.7609
##             Specificity : 0.4658
##          Pos Pred Value : 0.4730
##          Neg Pred Value : 0.7556
##              Prevalence : 0.3866
##          Detection Rate : 0.2941
##    Detection Prevalence : 0.6218
##       Balanced Accuracy : 0.6133
##
##        'Positive' Class : 0
##
```

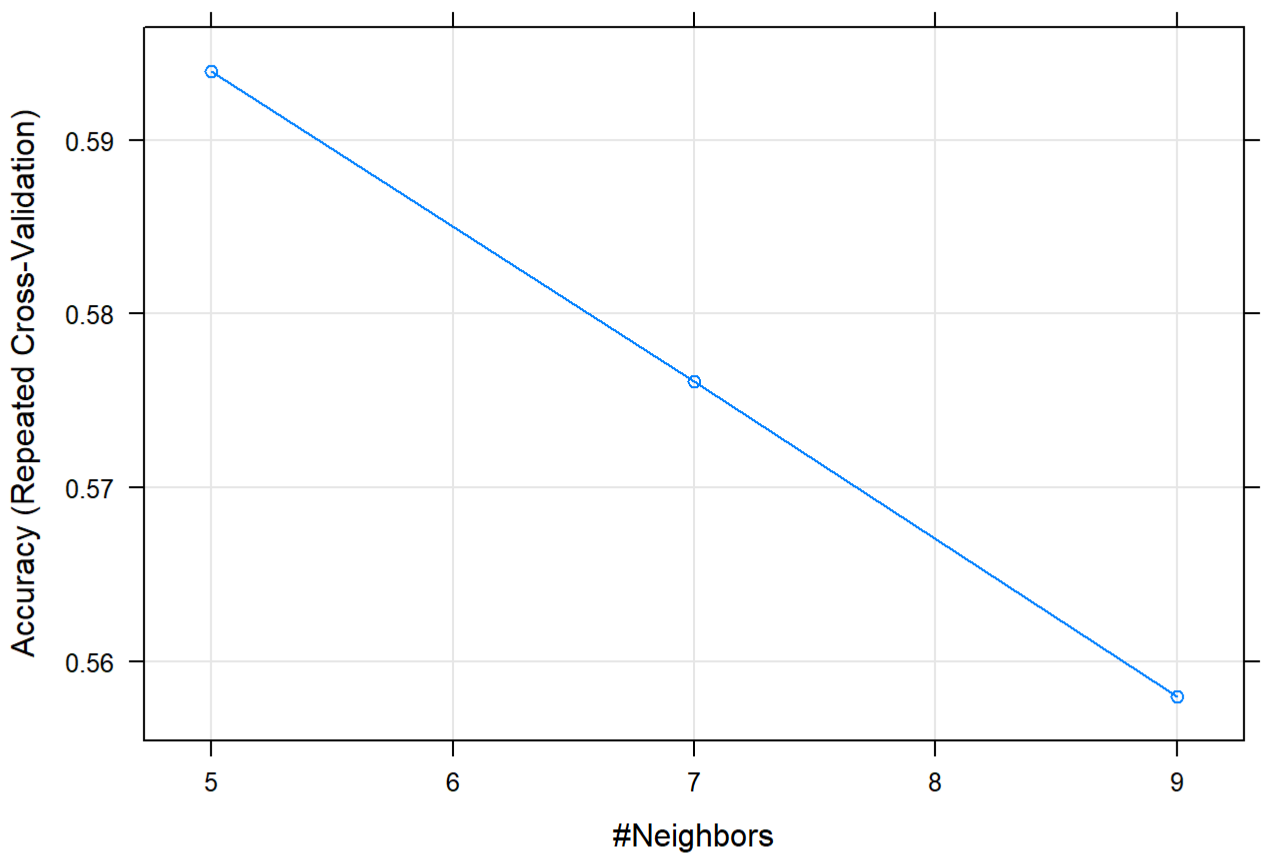
Sensitivity (true positive rate) is the probability of a positive test result, conditioned on the individual truly being positive. Specificity (true negative rate) is the probability of a negative test result, conditioned on the individual truly being negative.

***k-NN***

```
trainControl <- trainControl(method="repeatedcv", number=10, repeats=3)
fit.knn <- train(Target ~., data = train_student, method = 'knn', metric = 'Accuracy', trControl
= trainControl)
knn.k1 <- fit.knn$bestTune
print(fit.knn)
```

```
## k-Nearest Neighbors
##
## 276 samples
##  29 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 249, 249, 248, 248, 248, 249, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.5939594  0.16795933
##   7  0.5761023  0.12819282
##   9  0.5579806  0.08025146
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
plot(fit.knn)
```

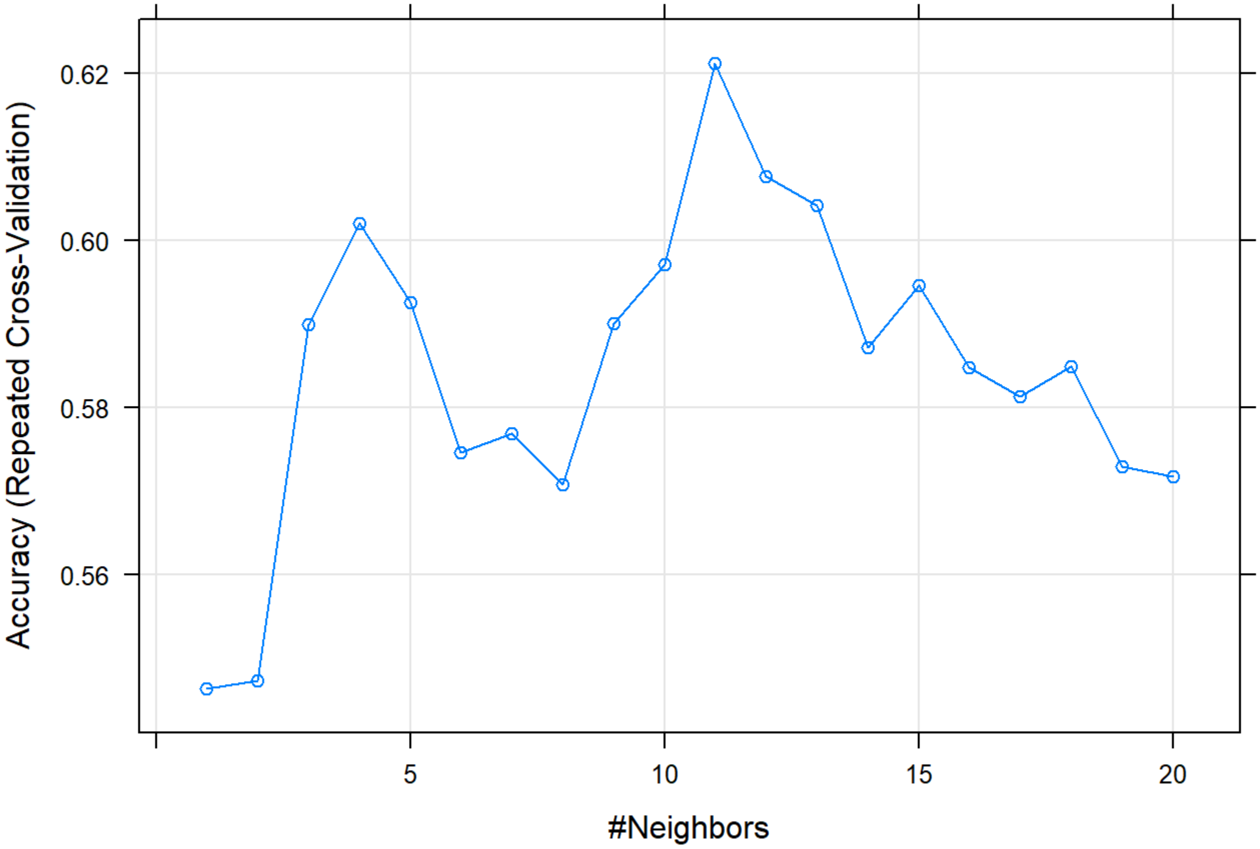


```
grid <- expand.grid(.k=seq(1,20,by=1))
fit.knn <- train(Target~., data=train_student_1, method='knn', metric = 'Accuracy', tuneGrid=gri
d, trControl=trainControl)
knn.k2 <- fit.knn$bestTune
print(fit.knn)
```

```
## k-Nearest Neighbors
##
## 276 samples
##  10 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 248, 249, 248, 248, 248, 249, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.5462522  0.08159795
##    2  0.5472663  0.08076681
##    3  0.5898589  0.15447645
##    4  0.6019841  0.17563048
##    5  0.5925485  0.14964771
##    6  0.5745150  0.11830269
##    7  0.5768519  0.12167089
##    8  0.5707231  0.11375347
##    9  0.5899912  0.14969807
##   10  0.5970459  0.16347018
##   11  0.6212081  0.21071701
##   12  0.6076279  0.18409971
##   13  0.6041005  0.17843263
##   14  0.5871693  0.14251955
##   15  0.5945767  0.15434194
##   16  0.5847002  0.13554529
##   17  0.5812169  0.12540594
##   18  0.5848765  0.13501729
##   19  0.5728395  0.10892024
##   20  0.5716490  0.10390101
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 11.
```

Accuracy is highest when k=4

```
plot(fit.knn)
```

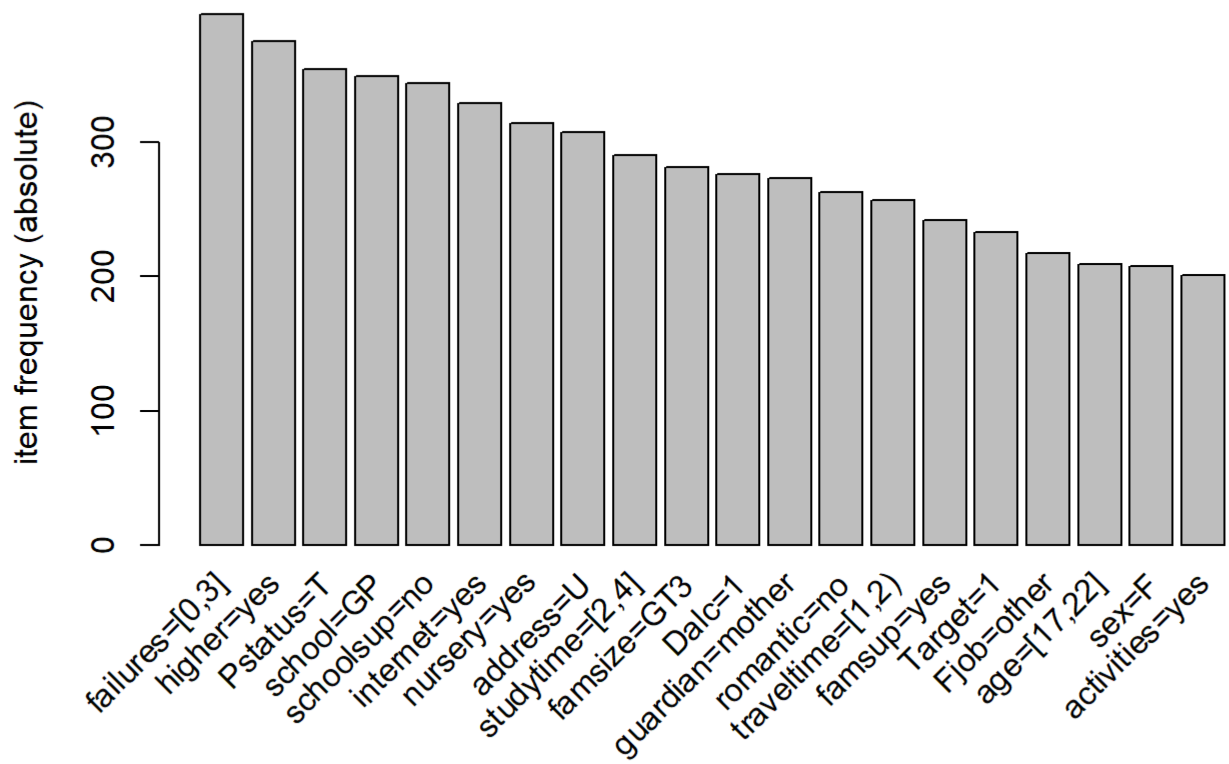



```
prediction <- predict(fit.knn, newdata = test_student_1)
cf <- confusionMatrix(prediction, test_student_1$Target)
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0  67  41
##          1  49 119
##
##                Accuracy : 0.6739
##                  95% CI : (0.6151, 0.7289)
##     No Information Rate : 0.5797
##     P-Value [Acc > NIR] : 0.0008239
##
##                   Kappa : 0.3244
##
##  Mcnemar's Test P-Value : 0.4605966
##
##             Sensitivity : 0.5776
##             Specificity : 0.7438
##          Pos Pred Value : 0.6204
##          Neg Pred Value : 0.7083
##              Prevalence : 0.4203
##          Detection Rate : 0.2428
##    Detection Prevalence : 0.3913
##       Balanced Accuracy : 0.6607
##
##        'Positive' Class : 0
##
```

*Association Rule Mining*

```
transactions <- as(student_df, 'transactions')
itemFrequencyPlot(transactions, topN=20, type='absolute')
```




```
rules <- apriori(transactions, parameter = list(supp = 0.03, conf = 0.8), appearance= list(defau
lt = 'lhs', rhs = 'Target=1'))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5    0.03      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 11
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[94 item(s), 395 transaction(s)] done [0.00s].
## sorting and recoding items ... [89 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 done [6.65s].
## writing ... [95954 rule(s)] done [0.41s].
## creating S4 object  ... done [0.26s].
```

```
rules <- sort(rules, decreasing = TRUE, by = 'lift')
```

```
inspect(rules[1:40])
```

```
##      lhs                      rhs          support confidence   coverage     lift count
## [1]  {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.04556962          1 0.04556962 1.695279    18
## [2]  {Medu=3,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [3]  {reason=home,
##       nursery=yes,
##       Walc=4}            => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [4]  {guardian=mother,
##       goout=5,
##       absences=[0,2)}    => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [5]  {schoolsup=no,
##       goout=5,
##       absences=[0,2)}    => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [6]  {Medu=1,
##       Fedu=1,
##       Mjob=other}        => {Target=1} 0.04810127          1 0.04810127 1.695279    19
## [7]  {sex=F,
##       internet=no,
##       absences=[2,6)}    => {Target=1} 0.03544304          1 0.03544304 1.695279    14
## [8]  {address=R,
##       famsize=GT3,
##       Walc=3}            => {Target=1} 0.03544304          1 0.03544304 1.695279    14
## [9]  {Fedu=1,
##       romantic=yes,
##       freetime=3}        => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [10] {address=R,
##       famsize=GT3,
##       goout=4}           => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [11] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       Walc=1}            => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [12] {sex=F,
##       age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [13] {sex=F,
##       age=[15,16),
##       schoolsup=yes,
##       internet=yes}      => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [14] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       famsup=yes}        => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [15] {age=[15,16),
##       traveltime=[1,2),
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [16] {age=[15,16),
```

```
##       studytime=[2,4],
##       schoolsup=yes,
##       romantic=no}       => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [17] {age=[15,16),
##       guardian=mother,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [18] {age=[15,16),
##       guardian=mother,
##       schoolsup=yes,
##       internet=yes}      => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [19] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       Dalc=1}            => {Target=1} 0.03797468          1 0.03797468 1.695279    15
## [20] {age=[15,16),
##       famsize=GT3,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [21] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       nursery=yes}       => {Target=1} 0.03797468          1 0.03797468 1.695279    15
## [22] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       internet=yes}      => {Target=1} 0.04050633          1 0.04050633 1.695279    16
## [23] {school=GP,
##       age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.04556962          1 0.04556962 1.695279    18
## [24] {age=[15,16),
##       Pstatus=T,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03797468          1 0.03797468 1.695279    15
## [25] {age=[15,16),
##       studytime=[2,4],
##       schoolsup=yes,
##       higher=yes}        => {Target=1} 0.04556962          1 0.04556962 1.695279    18
## [26] {age=[15,16),
##       studytime=[2,4],
##       failures=[0,3],
##       schoolsup=yes}     => {Target=1} 0.04556962          1 0.04556962 1.695279    18
## [27] {school=GP,
##       Medu=3,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [28] {Medu=3,
##       studytime=[2,4],
##       schoolsup=yes,
##       higher=yes}        => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [29] {Medu=3,
```

```
##       studytime=[2,4],
##       failures=[0,3],
##       schoolsup=yes}     => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [30] {studytime=[2,4],
##       schoolsup=yes,
##       famsup=yes,
##       absences=[2,6)}    => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [31] {Pstatus=T,
##       Mjob=other,
##       schoolsup=yes,
##       romantic=no}       => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [32] {Pstatus=T,
##       Mjob=other,
##       studytime=[2,4],
##       schoolsup=yes}     => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [33] {studytime=[2,4],
##       schoolsup=yes,
##       activities=yes,
##       famrel=4}          => {Target=1} 0.03544304          1 0.03544304 1.695279    14
## [34] {address=U,
##       schoolsup=yes,
##       activities=yes,
##       famrel=4}          => {Target=1} 0.03037975          1 0.03037975 1.695279    12
## [35] {schoolsup=yes,
##       activities=yes,
##       internet=yes,
##       famrel=4}          => {Target=1} 0.03544304          1 0.03544304 1.695279    14
## [36] {traveltime=[1,2),
##       studytime=[2,4],
##       schoolsup=yes,
##       famrel=4}          => {Target=1} 0.03797468          1 0.03797468 1.695279    15
## [37] {traveltime=[1,2),
##       schoolsup=yes,
##       internet=yes,
##       famrel=4}          => {Target=1} 0.03291139          1 0.03291139 1.695279    13
## [38] {studytime=[2,4],
##       schoolsup=yes,
##       romantic=no,
##       famrel=4}          => {Target=1} 0.04556962          1 0.04556962 1.695279    18
## [39] {address=U,
##       schoolsup=yes,
##       romantic=no,
##       famrel=4}          => {Target=1} 0.03797468          1 0.03797468 1.695279    15
## [40] {schoolsup=yes,
##       internet=yes,
##       romantic=no,
##       famrel=4}          => {Target=1} 0.03797468          1 0.03797468 1.695279    15
```