Student Performance Predictions

```r
student_por = read.csv('student+performance/student-por.csv', sep = ';',  stringsAsFactors = TRUE)
student_math = read.csv('student+performance/student-mat.csv', sep = ';', stringsAsFactors = TRUE)
all_data = rbind(student_por, student_math)
summary(all_data)
```

```
##  school   sex          age        address famsize   Pstatus      Medu
##  GP:772   F:591   Min.   :15.00   R:285   GT3:738   A:121   Min.   :0.000
##  MS:272   M:453   1st Qu.:16.00   U:759   LE3:306   T:923   1st Qu.:2.000
##                   Median :17.00                             Median :3.000
##                   Mean   :16.73                             Mean   :2.603
##                   3rd Qu.:18.00                             3rd Qu.:4.000
##                   Max.   :22.00                             Max.   :4.000
##       Fedu             Mjob           Fjob            reason      guardian
##  Min.   :0.000   at_home :194   at_home : 62   course    :430   father:243
##  1st Qu.:1.000   health  : 82   health  : 41   home      :258   mother:728
##  Median :2.000   other   :399   other   :584   other     :108   other : 73
##  Mean   :2.388   services:239   services:292   reputation:248
##  3rd Qu.:3.000   teacher :130   teacher : 65
##  Max.   :4.000
##    traveltime      studytime       failures      schoolsup famsup     paid
##  Min.   :1.000   Min.   :1.00   Min.   :0.0000   no :925   no :404   no :824
##  1st Qu.:1.000   1st Qu.:1.00   1st Qu.:0.0000   yes:119   yes:640   yes:220
##  Median :1.000   Median :2.00   Median :0.0000
##  Mean   :1.523   Mean   :1.97   Mean   :0.2644
##  3rd Qu.:2.000   3rd Qu.:2.00   3rd Qu.:0.0000
##  Max.   :4.000   Max.   :4.00   Max.   :3.0000
##  activities nursery   higher    internet  romantic      famrel
##  no :528    no :209   no : 89   no :217   no :673   Min.   :1.000
##  yes:516    yes:835   yes:955   yes:827   yes:371   1st Qu.:4.000
##                                                     Median :4.000
##                                                     Mean   :3.936
##                                                     3rd Qu.:5.000
##                                                     Max.   :5.000
##     freetime         goout           Dalc            Walc
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:3.000   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :3.000   Median :3.000   Median :1.000   Median :2.000
##  Mean   :3.201   Mean   :3.156   Mean   :1.494   Mean   :2.284
##  3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:2.000   3rd Qu.:3.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
##      health         absences           G1              G2
##  Min.   :1.000   Min.   : 0.000   Min.   : 0.00   Min.   : 0.00
##  1st Qu.:3.000   1st Qu.: 0.000   1st Qu.: 9.00   1st Qu.: 9.00
##  Median :4.000   Median : 2.000   Median :11.00   Median :11.00
##  Mean   :3.543   Mean   : 4.435   Mean   :11.21   Mean   :11.25
##  3rd Qu.:5.000   3rd Qu.: 6.000   3rd Qu.:13.00   3rd Qu.:13.00
##  Max.   :5.000   Max.   :75.000   Max.   :19.00   Max.   :19.00
```

```
##        G3
##  Min.   : 0.00
##  1st Qu.:10.00
##  Median :11.00
##  Mean   :11.34
##  3rd Qu.:14.00
##  Max.   :20.00
```

```r
head(all_data)
```

```
##   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        0       yes     no   no         no
## 4   mother          1         3        0        no    yes   no        yes
## 5   father          1         2        0        no    yes   no         no
## 6   mother          1         2        0        no    yes   no        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        4  0 11 11
## 2        2  9 11 11
## 3        6 12 13 12
## 4        0 14 14 14
## 5        0 11 13 13
## 6        6 12 12 13
```

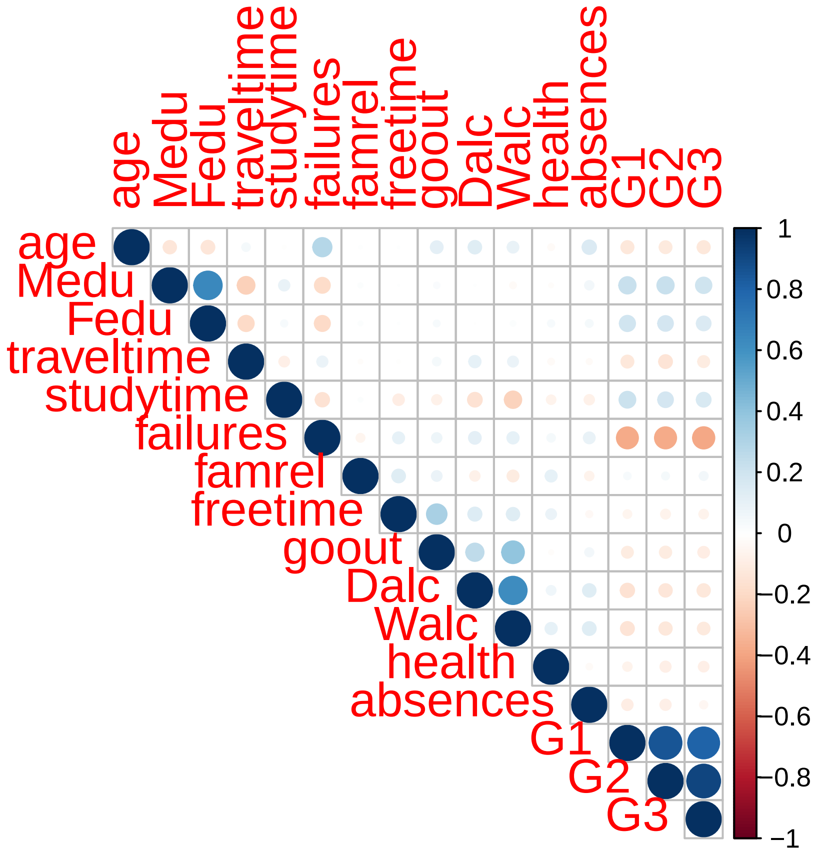
```r
write.csv(all_data, 'student+performance/merged_data.csv', row.names = FALSE)
```

Covariance

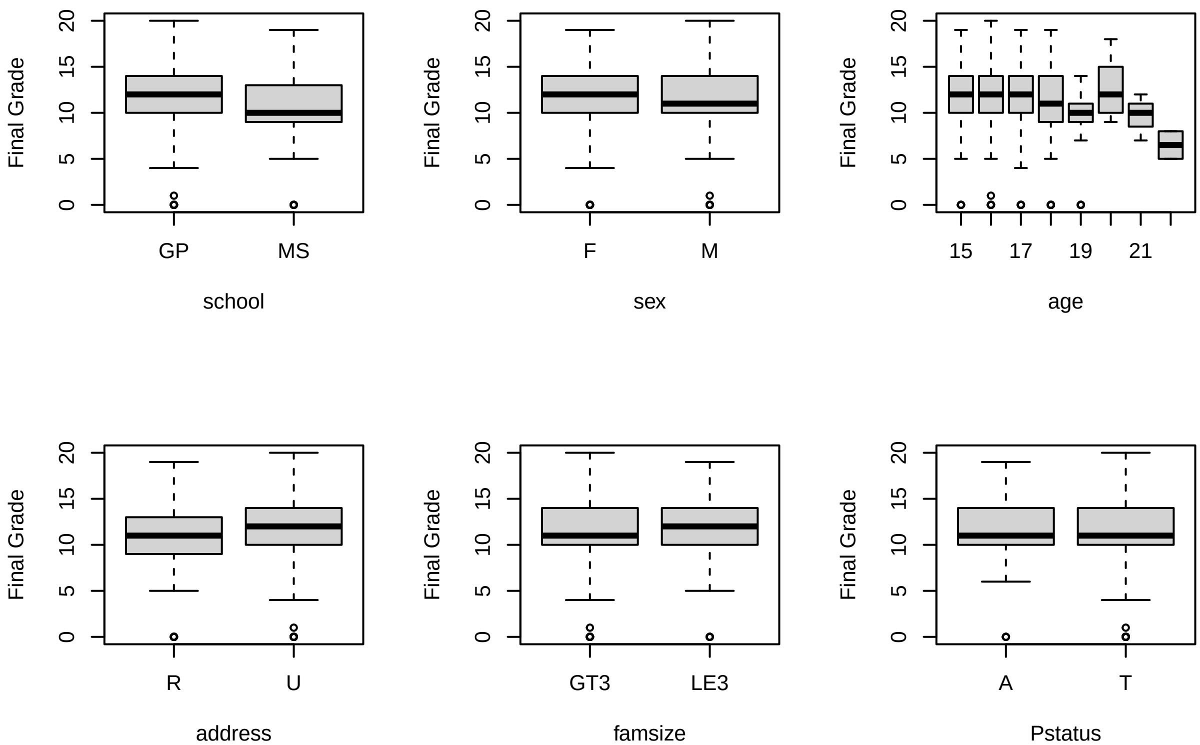
```r
library(corrplot)
```

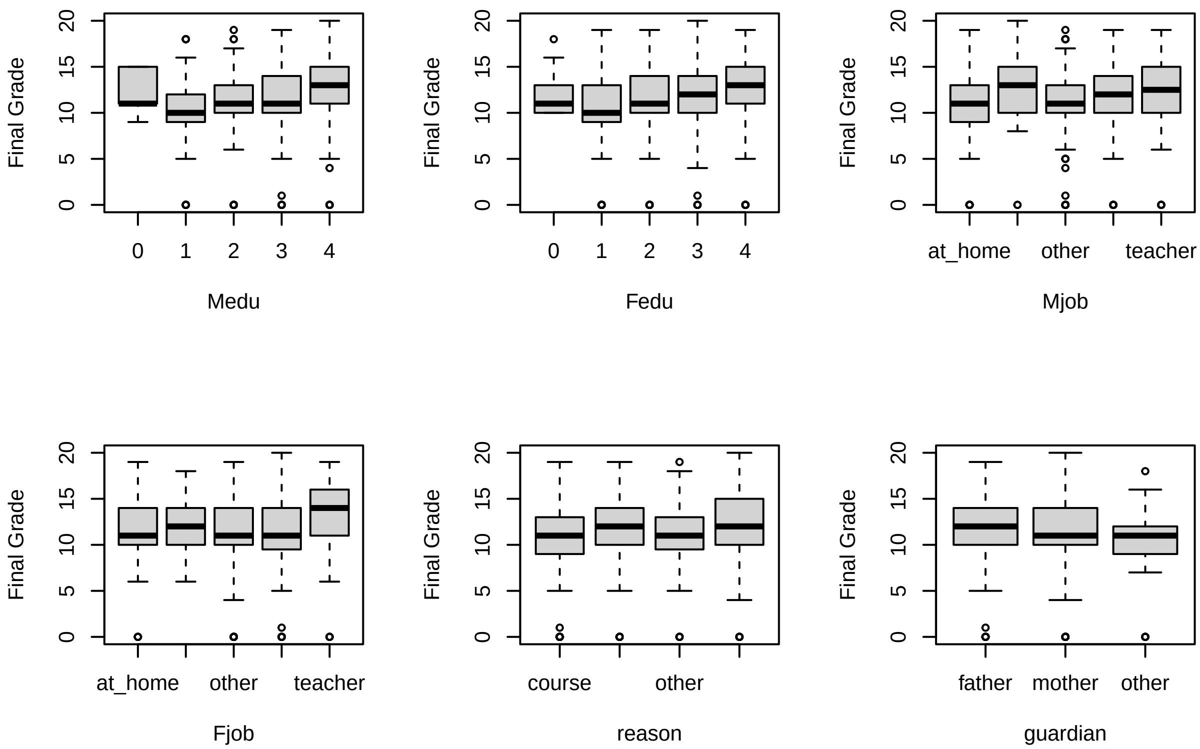
```
## corrplot 0.95 loaded
```

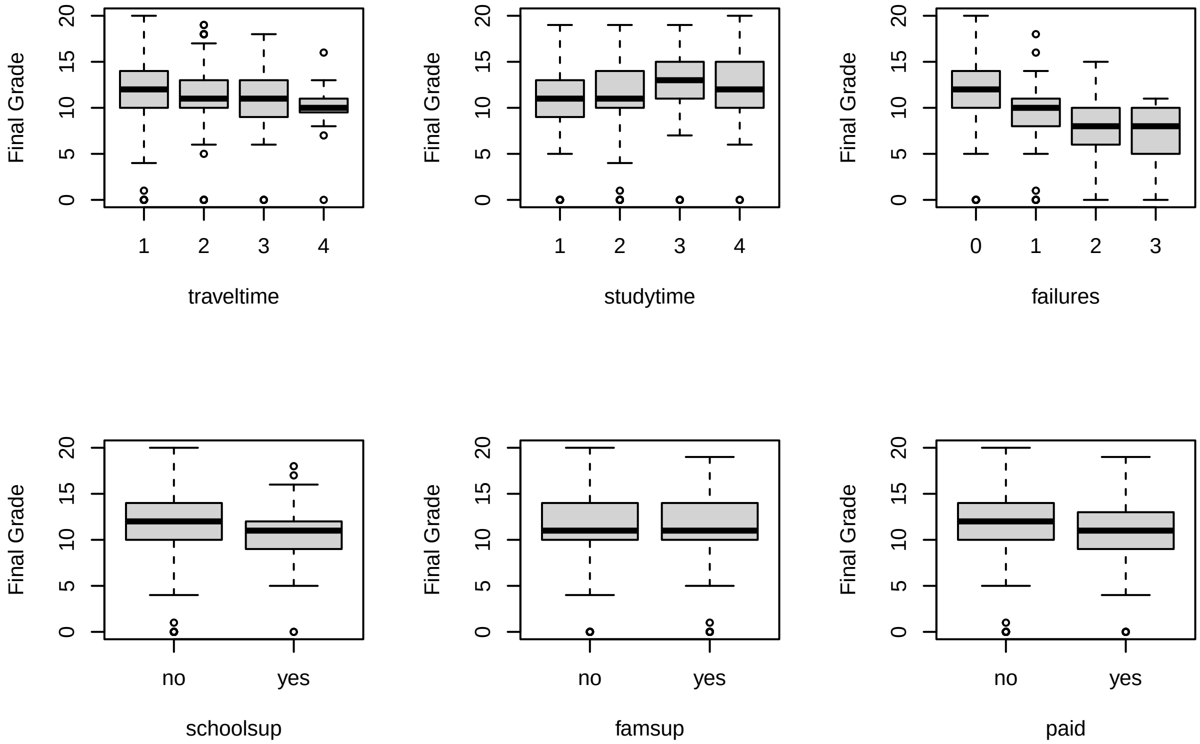
```r
numAll = all_data[ ,sapply(all_data, is.numeric)]
corrMat = cor(numAll)
corrplot(corrMat, type = "upper", number.cex = 1.5, tl.cex = 1.5)
```

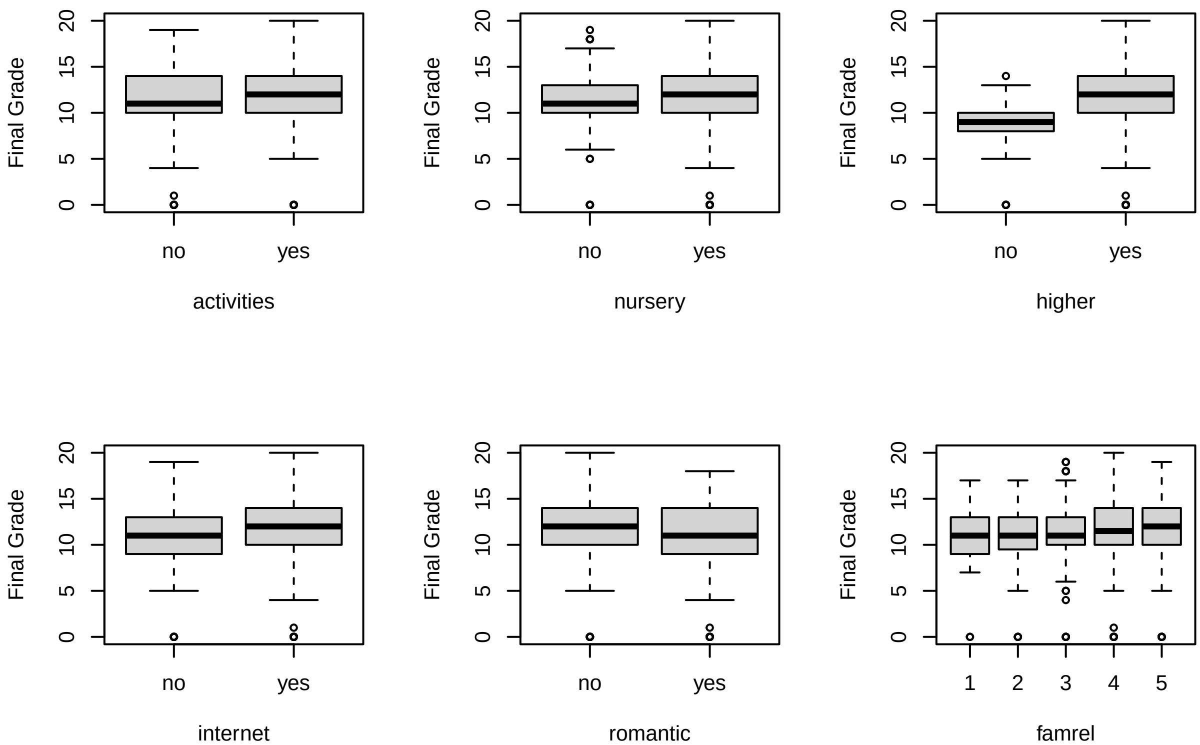


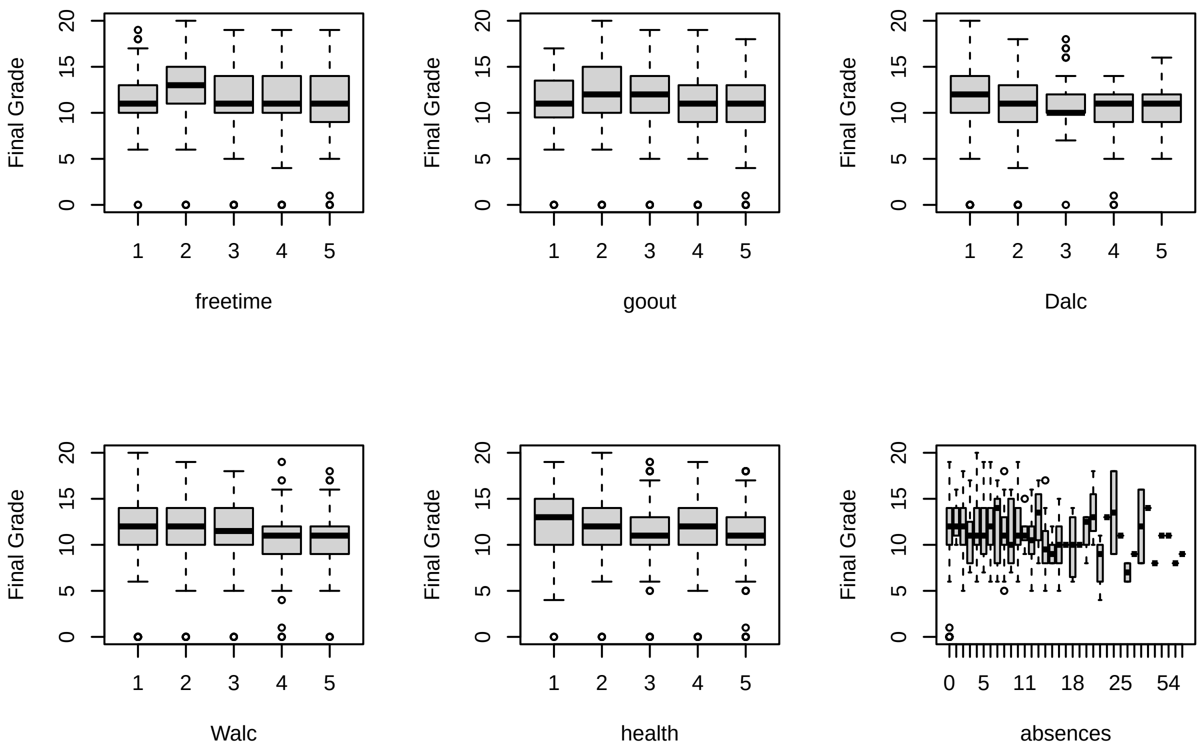

```r
par(mfrow = c(2, 3))
namesInData = names(all_data)
for (i in namesInData){
  if (i != "G3") {
    boxplot(all_data$G3 ~ all_data[, i], xlab = i, ylab = "Final Grade")
  }

}
```

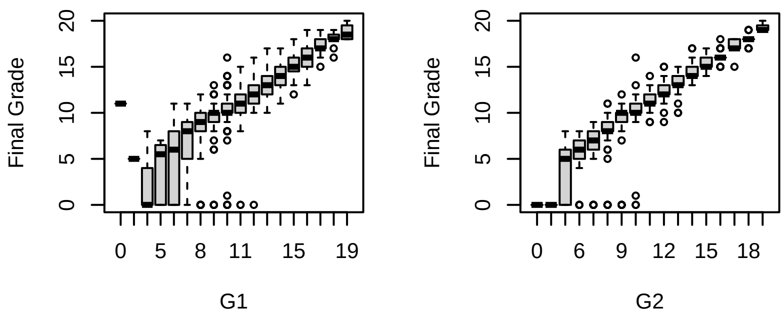

Final Grade
0
5
10
15
20
GP
MS
school
F
M
sex
15
17
19
21
age
R
U
address
GT3
LE3
famsize
A
T
Pstatus


Final Grade
0
5
10
15
20
0 1 2 3 4
Medu
Final Grade
0 1 2 3 4
Fedu
Final Grade
at_home other teacher
Mjob
Final Grade
at_home other teacher
Fjob
Final Grade
course other
reason
Final Grade
father mother other
guardian


Final Grade
0
5
10
15
20
1
2
3
4
traveltime
studytime
0
1
2
3
failures
no
yes
schoolsup
famsup
paid

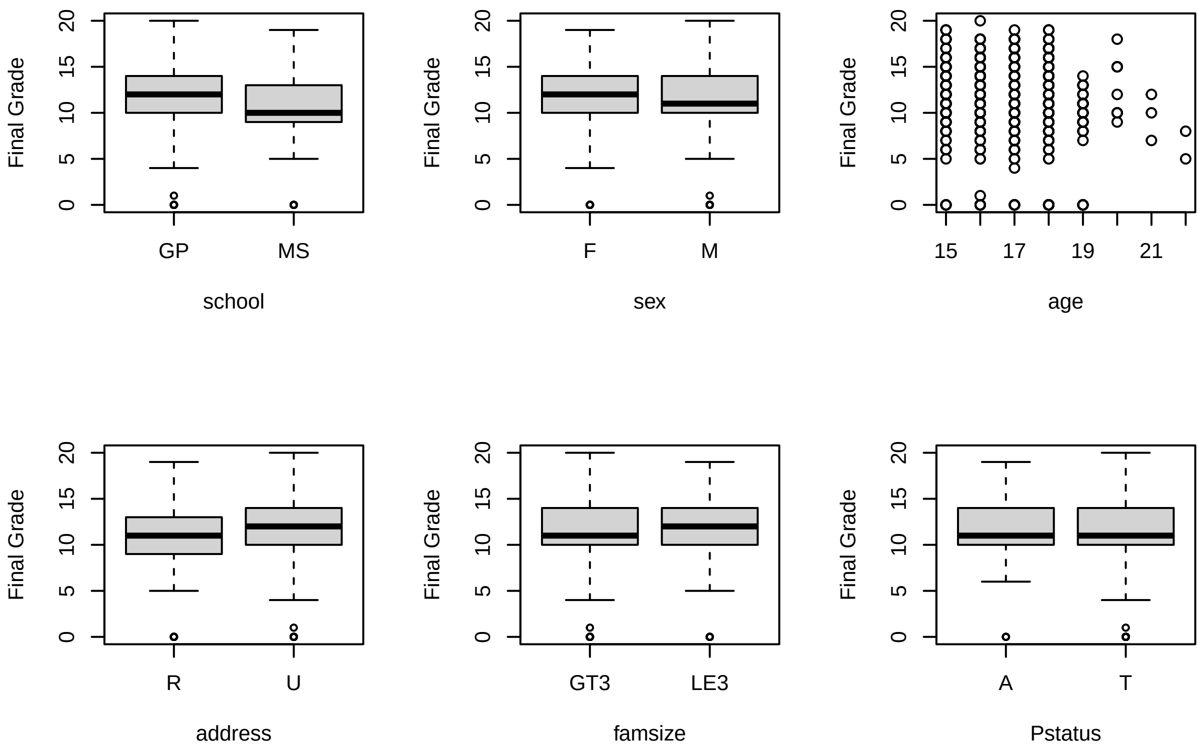

Final Grade
0
5
10
15
20
no
yes
activities
nursery
higher
internet
romantic
1
2
3
4
5
famrel

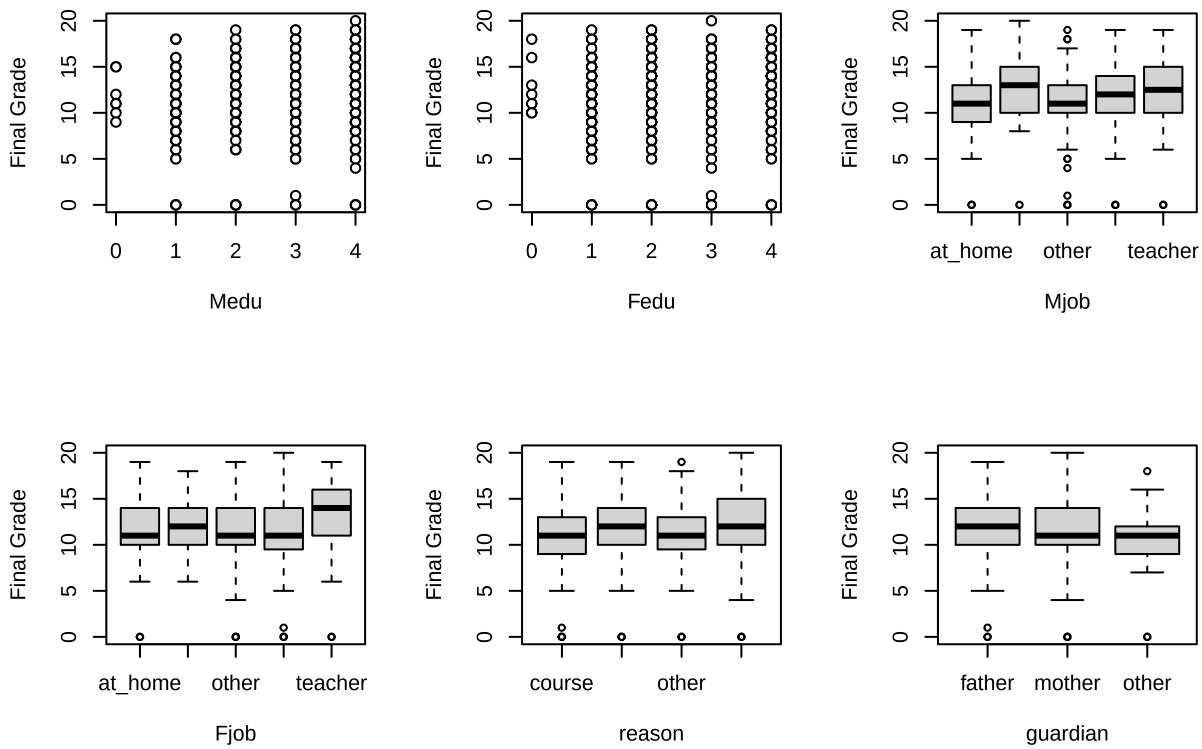

Final Grade
freetime
goout
Dalc
Walc
health
absences
0 5 10 15 20
1 2 3 4 5
0 5 11 18 25 54

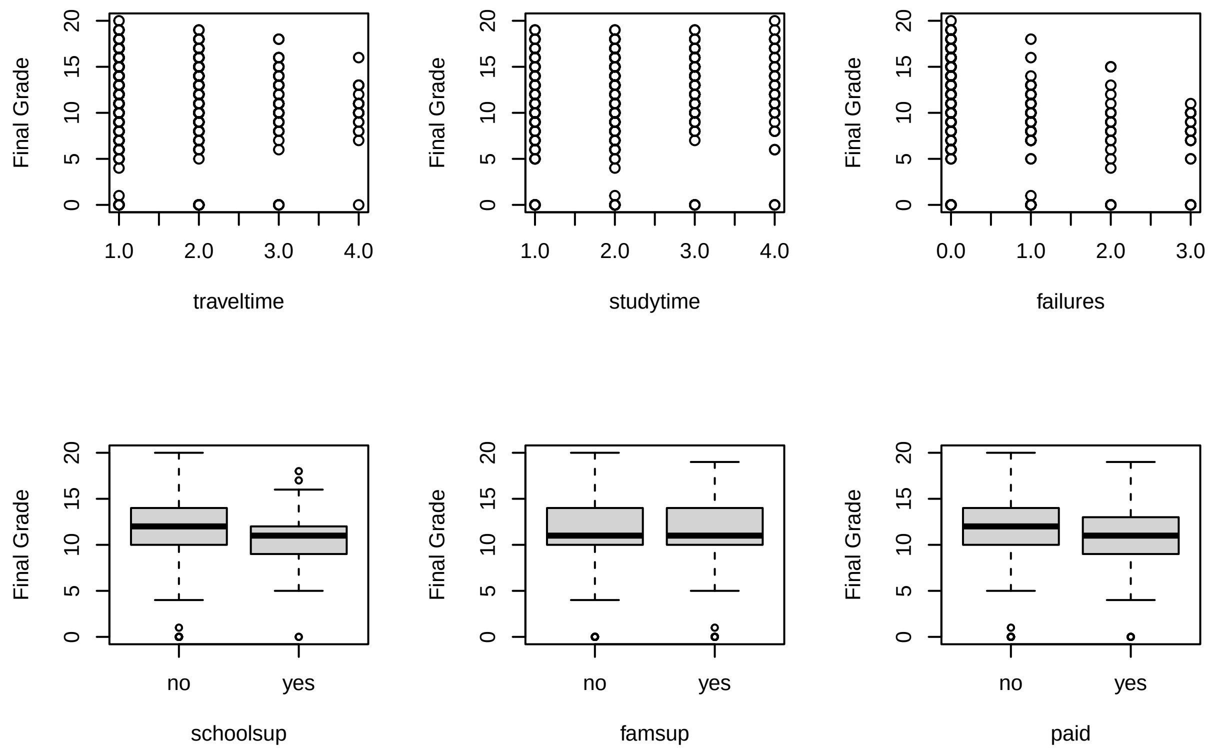


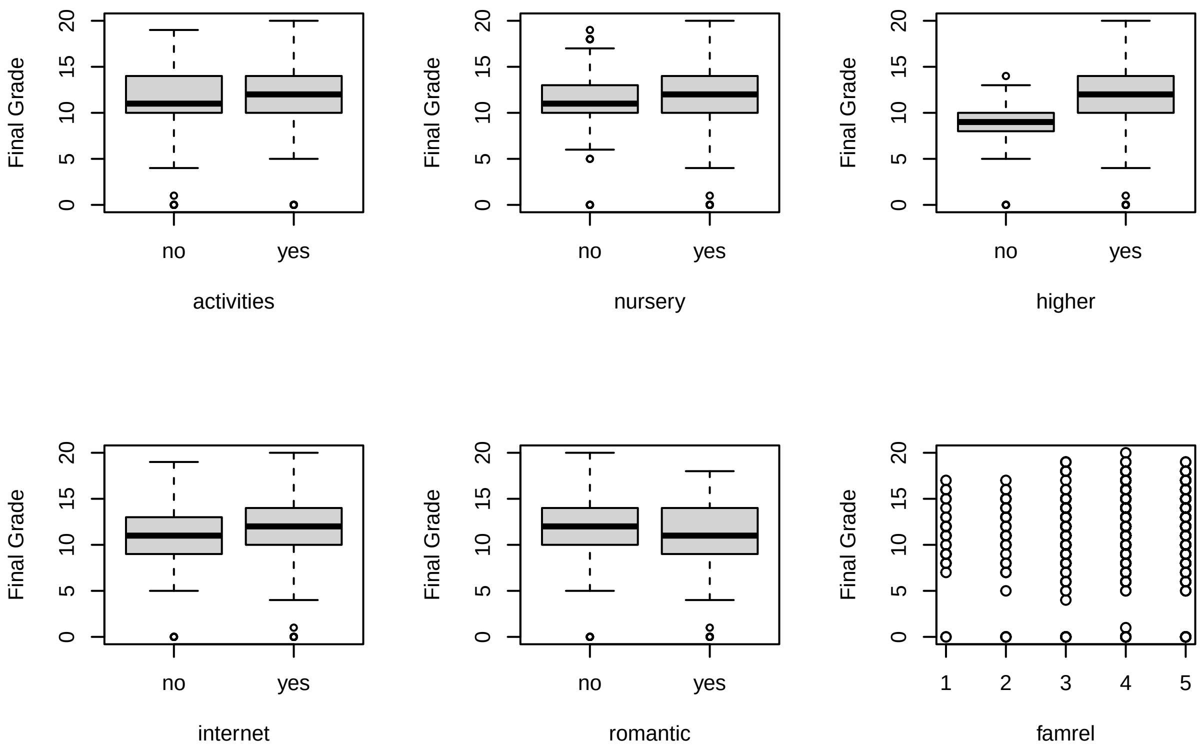
- We see that study time gives a higher final grade along with G1 and G2, which are first and second period grades for the class. We notice the the number of failures the student has had gives and impact for the final grade of there current class.

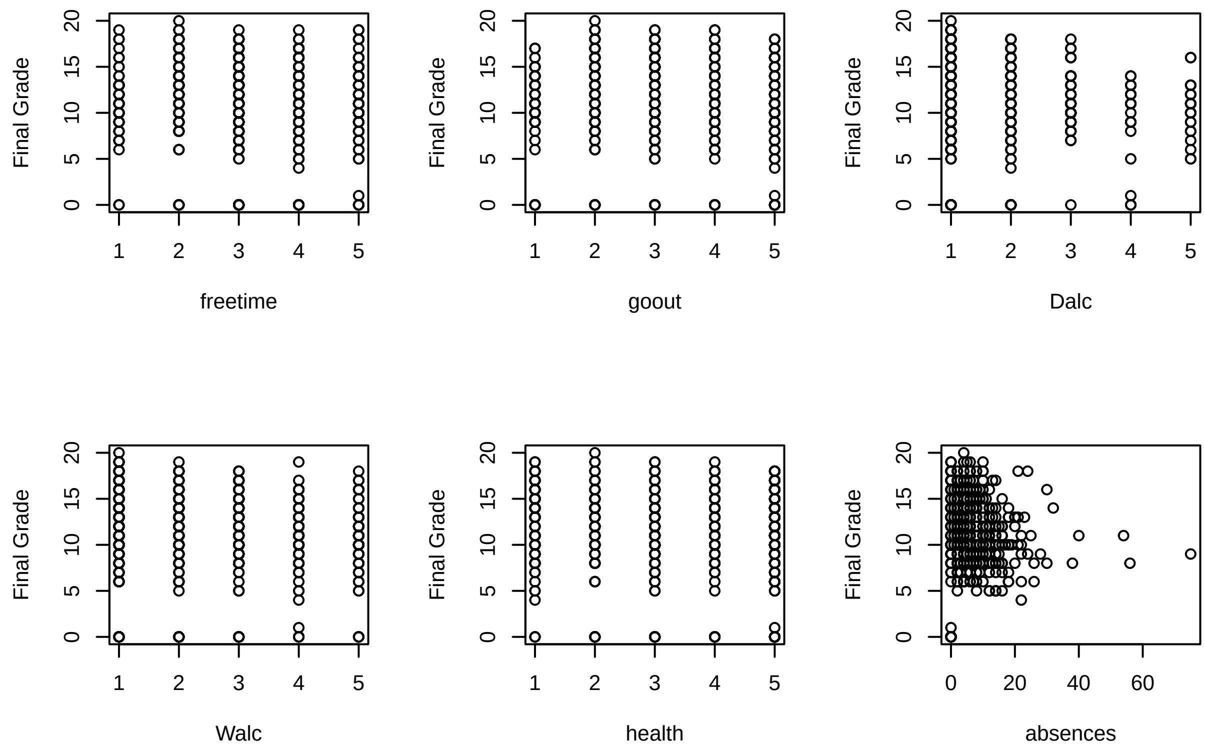
```
par(mfrow = c(2, 3))
namesInData = names(all_data)
for (i in namesInData){
  if (i != "G3") {
      plot(all_data$G3 ~ all_data[, i], xlab = i, ylab = "Final Grade")
  }

}
```

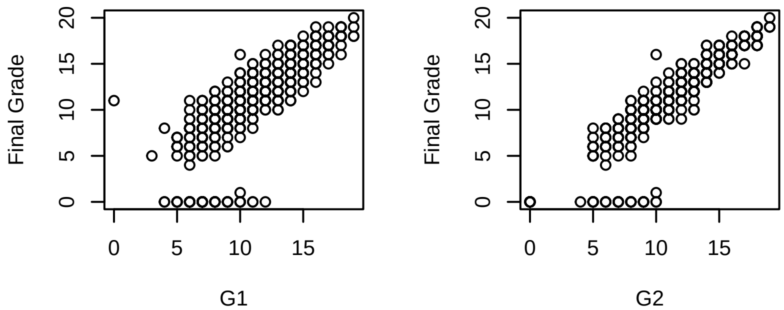

Final Grade
0
5
10
15
20
GP
MS
school
F
M
sex
15
17
19
21
age
R
U
address
GT3
LE3
famsize
A
T
Pstatus


Final Grade
0
5
10
15
20
0
1
2
3
4
Medu
Fedu
at_home
other
teacher
Mjob
Fjob
course
other
reason
father
mother
other
guardian


Final Grade
traveltime
studytime
failures
schoolsup
famsup
paid
no
yes


Final Grade
0
5
10
15
20
no
yes
activities
nursery
higher
internet
romantic
1
2
3
4
5
famrel


Final Grade
freetime
goout
Dalc
Walc
health
absences
0
5
10
15
20
1
2
3
4
5
40
60



- We need to convert some of the data to factor for example study and travel time since it just classifying a range. In studytime 1 is for less then 2 hours, 2 is for 2 to 5 hours, 3 is for 5 to 10 hours, and 4 is for 10 hours or more. We would have to change these variables so it doesn't mess with our models.

```
nameOfNumericCols = names(all_data)[sapply(all_data, is.numeric)]
nameOfNumericCols
```

```
##  [1] "age"        "Medu"       "Fedu"       "traveltime" "studytime"
##  [6] "failures"   "famrel"     "freetime"   "goout"      "Dalc"
## [11] "Walc"       "health"     "absences"   "G1"         "G2"
## [16] "G3"
```

- These are the values that are considered numeric, but we need to change the ones that are just numeric values to define a category. We could change the values for health, Walc, Dalc, goout, and freetime to factors since it just stating a category.

```
set.seed(1)
n = nrow(all_data)
nFeatures= ncol(all_data) - 1
percentTrain = .7
nTrain = n*percentTrain
trainIdx = sample(1:n, nTrain)
trainData = all_data[trainIdx, ]
allLinearFit = lm(G3 ~ ., trainData)
testData = all_data[-trainIdx,]
summary(allLinearFit)
```

```
##
## Call:
```

```
## lm(formula = G3 ~ ., data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5947 -0.4877  0.1208  0.8346  5.2967
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -0.225640   1.170716  -0.193  0.84722
## schoolMS          0.116010   0.167918   0.691  0.48988
## sexM              0.059368   0.145215   0.409  0.68279
## age              -0.007457   0.058443  -0.128  0.89851
## addressU          0.172259   0.153825   1.120  0.26317
## famsizeLE3       -0.078917   0.144376  -0.547  0.58483
## PstatusT         -0.333724   0.213764  -1.561  0.11894
## Medu              0.060033   0.092483   0.649  0.51647
## Fedu             -0.097227   0.082065  -1.185  0.23653
## Mjobhealth       -0.009827   0.307394  -0.032  0.97451
## Mjobother        -0.075830   0.185939  -0.408  0.68353
## Mjobservices      0.103891   0.220099   0.472  0.63706
## Mjobteacher       0.155882   0.295804   0.527  0.59838
## Fjobhealth       -0.353970   0.448716  -0.789  0.43047
## Fjobother        -0.363086   0.269014  -1.350  0.17756
## Fjobservices     -0.636535   0.281288  -2.263  0.02395 *
## Fjobteacher      -0.657323   0.370821  -1.773  0.07674 .
## reasonhome       -0.155302   0.165694  -0.937  0.34894
## reasonother      -0.273730   0.220905  -1.239  0.21572
## reasonreputation -0.179130   0.169080  -1.059  0.28977
## guardianmother   -0.103299   0.152419  -0.678  0.49817
## guardianother    -0.090838   0.285108  -0.319  0.75012
## traveltime        0.184141   0.091817   2.006  0.04530 *
## studytime        -0.001513   0.081798  -0.019  0.98524
## failures         -0.352703   0.110436  -3.194  0.00147 **
## schoolsupyes      0.161154   0.207622   0.776  0.43790
## famsupyes         0.304814   0.134382   2.268  0.02362 *
## paidyes          -0.360163   0.162406  -2.218  0.02690 *
## activitiesyes    -0.157399   0.129481  -1.216  0.22455
## nurseryyes       -0.073752   0.160084  -0.461  0.64515
## higheryes         0.046221   0.239576   0.193  0.84707
## internetyes      -0.005898   0.166950  -0.035  0.97183
## romanticyes      -0.081575   0.133812  -0.610  0.54231
## famrel            0.065754   0.069494   0.946  0.34438
## freetime          0.030135   0.065734   0.458  0.64679
## goout            -0.056127   0.063422  -0.885  0.37647
## Dalc             -0.061533   0.091475  -0.673  0.50138
## Walc              0.066053   0.072318   0.913  0.36137
## health           -0.049724   0.044910  -1.107  0.26860
## absences          0.026853   0.010356   2.593  0.00971 **
## G1                0.109881   0.040467   2.715  0.00679 **
## G2                0.970555   0.035448  27.380  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.637 on 688 degrees of freedom
```

```
## Multiple R-squared:  0.8373, Adjusted R-squared:  0.8276
## F-statistic: 86.36 on 41 and 688 DF,  p-value: < 2.2e-16
```

```
preds = predict(allLinearFit, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.454816
```

```
linearFit = lm(G3 ~ G1 + G2 + studytime + failures, trainData)
summary(linearFit)
```

```
##
## Call:
## lm(formula = G3 ~ G1 + G2 + studytime + failures, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7815 -0.3544  0.0070  0.8234  5.8788
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.55278    0.28167  -1.962  0.05009 .
## G1           0.09832    0.03900   2.521  0.01191 *
## G2           0.97226    0.03475  27.980  < 2e-16 ***
## studytime   -0.03187    0.07539  -0.423  0.67261
## failures    -0.30785    0.09722  -3.166  0.00161 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.644 on 725 degrees of freedom
## Multiple R-squared:  0.8269, Adjusted R-squared:  0.826
## F-statistic:   866 on 4 and 725 DF,  p-value: < 2.2e-16
```

```
preds = predict(linearFit, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.419153
```

-best Subset selection

```
#install.packages("leaps")
library(leaps)
regfit.full = regsubsets(G3 ~ ., trainData, nvmax = nFeatures)
regfit.summary = summary(regfit.full)
regfit.summary
```

```
## Subset selection object
## Call: regsubsets.formula(G3 ~ ., trainData, nvmax = nFeatures)
## 41 Variables  (and intercept)
##                   Forced in Forced out
## schoolMS              FALSE      FALSE
## sexM                  FALSE      FALSE
## age                   FALSE      FALSE
## addressU              FALSE      FALSE
## famsizeLE3            FALSE      FALSE
## PstatusT              FALSE      FALSE
## Medu                  FALSE      FALSE
```

```
## Fedu                   FALSE      FALSE
## Mjobhealth             FALSE      FALSE
## Mjobother              FALSE      FALSE
## Mjobservices           FALSE      FALSE
## Mjobteacher            FALSE      FALSE
## Fjobhealth             FALSE      FALSE
## Fjobother              FALSE      FALSE
## Fjobservices           FALSE      FALSE
## Fjobteacher            FALSE      FALSE
## reasonhome             FALSE      FALSE
## reasonother            FALSE      FALSE
## reasonreputation       FALSE      FALSE
## guardianmother         FALSE      FALSE
## guardianother          FALSE      FALSE
## traveltime             FALSE      FALSE
## studytime              FALSE      FALSE
## failures               FALSE      FALSE
## schoolsupyes           FALSE      FALSE
## famsupyes              FALSE      FALSE
## paidyes                FALSE      FALSE
## activitiesyes          FALSE      FALSE
## nurseryyes             FALSE      FALSE
## higheryes              FALSE      FALSE
## internetyes            FALSE      FALSE
## romanticyes            FALSE      FALSE
## famrel                 FALSE      FALSE
## freetime               FALSE      FALSE
## goout                  FALSE      FALSE
## Dalc                   FALSE      FALSE
## Walc                   FALSE      FALSE
## health                 FALSE      FALSE
## absences               FALSE      FALSE
## G1                     FALSE      FALSE
## G2                     FALSE      FALSE
## 1 subsets of each size up to 32
## Selection Algorithm: exhaustive
##           schoolMS sexM age addressU famsizeLE3 PstatusT Medu Fedu Mjobhealth
## 1  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 2  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 3  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 4  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 5  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 6  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 7  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 8  ( 1 )  " "      " "  " " " "      " "        " "      " "  " "  " "
## 9  ( 1 )  " "      " "  " " " "      " "        "*"      " "  " "  " "
## 10  ( 1 ) " "      " "  " " " "      " "        "*"      " "  " "  " "
## 11  ( 1 ) " "      " "  " " " "      " "        "*"      " "  " "  " "
## 12  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 13  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 14  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 15  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 16  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 17  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
```

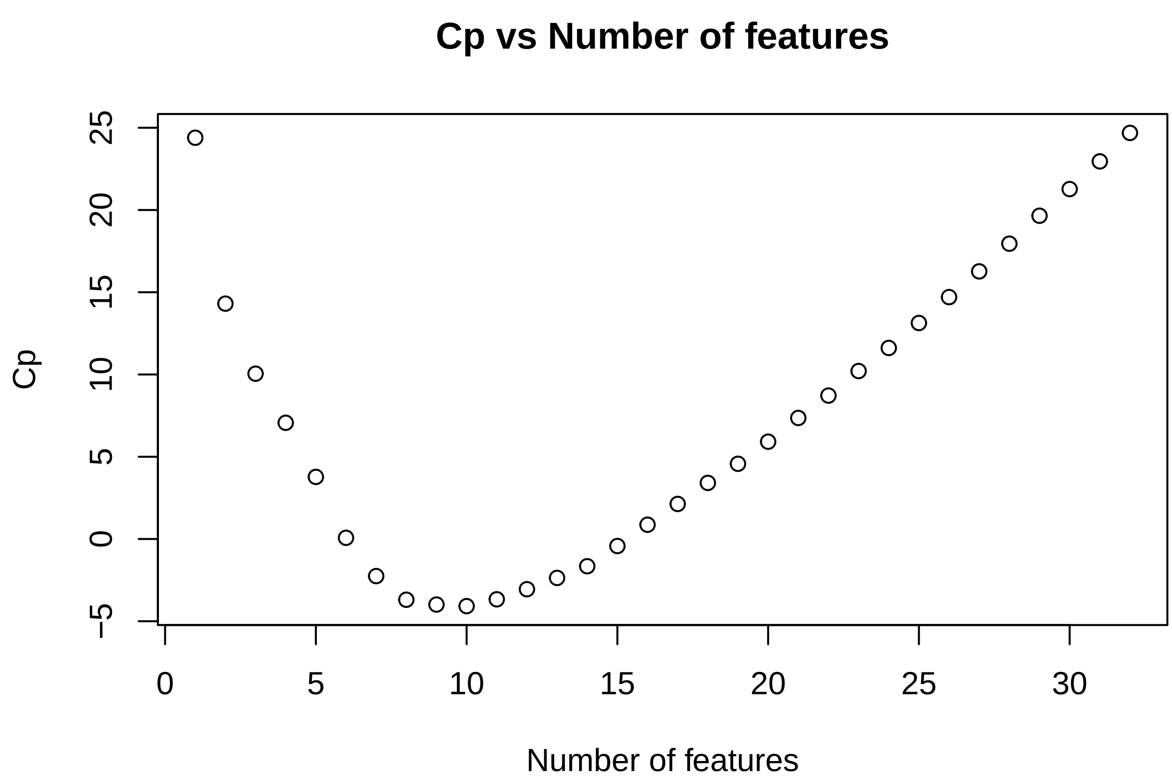
```
## 18  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 19  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 20  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 21  ( 1 ) " "      " "  " " "*"      " "        "*"      " "  " "  " "
## 22  ( 1 ) " "      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 23  ( 1 ) " "      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 24  ( 1 ) " "      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 25  ( 1 ) " "      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 26  ( 1 ) " "      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 27  ( 1 ) "*"      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 28  ( 1 ) "*"      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 29  ( 1 ) "*"      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 30  ( 1 ) "*"      " "  " " "*"      " "        "*"      "*"  "*"  " "
## 31  ( 1 ) "*"      " "  " " "*"      "*"        "*"      "*"  "*"  " "
## 32  ( 1 ) "*"      " "  " " "*"      "*"        "*"      "*"  "*"  " "
##           Mjobother Mjobservices Mjobteacher Fjobhealth Fjobother Fjobservices
## 1  ( 1 )  " "       " "          " "         " "        " "       " "
## 2  ( 1 )  " "       " "          " "         " "        " "       " "
## 3  ( 1 )  " "       " "          " "         " "        " "       " "
## 4  ( 1 )  " "       " "          " "         " "        " "       " "
## 5  ( 1 )  " "       " "          " "         " "        " "       " "
## 6  ( 1 )  " "       " "          " "         " "        " "       " "
## 7  ( 1 )  " "       " "          " "         " "        " "       " "
## 8  ( 1 )  " "       " "          " "         " "        " "       "*"
## 9  ( 1 )  " "       " "          " "         " "        " "       "*"
## 10  ( 1 ) " "       " "          " "         " "        " "       "*"
## 11  ( 1 ) " "       " "          " "         " "        " "       "*"
## 12  ( 1 ) " "       " "          " "         " "        " "       "*"
## 13  ( 1 ) " "       " "          " "         " "        "*"       "*"
## 14  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 15  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 16  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 17  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 18  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 19  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 20  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 21  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 22  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 23  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 24  ( 1 ) " "       " "          " "         "*"        "*"       "*"
## 25  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 26  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 27  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 28  ( 1 ) "*"       " "          " "         "*"        "*"       "*"
## 29  ( 1 ) " "       "*"          "*"         "*"        "*"       "*"
## 30  ( 1 ) " "       "*"          "*"         "*"        "*"       "*"
## 31  ( 1 ) " "       "*"          "*"         "*"        "*"       "*"
## 32  ( 1 ) " "       "*"          "*"         "*"        "*"       "*"
##           Fjobteacher reasonhome reasonother reasonreputation guardianmother
## 1  ( 1 )  " "         " "        " "         " "              " "
## 2  ( 1 )  " "         " "        " "         " "              " "
## 3  ( 1 )  " "         " "        " "         " "              " "
## 4  ( 1 )  " "         " "        " "         " "              " "
## 5  ( 1 )  " "         " "        " "         " "              " "
```

```
## 6  ( 1 )  " "         " "        " "         " "              " "
## 7  ( 1 )  " "         " "        " "         " "              " "
## 8  ( 1 )  " "         " "        " "         " "              " "
## 9  ( 1 )  " "         " "        " "         " "              " "
## 10  ( 1 ) "*"         " "        " "         " "              " "
## 11  ( 1 ) "*"         " "        " "         " "              " "
## 12  ( 1 ) "*"         " "        " "         " "              " "
## 13  ( 1 ) "*"         " "        " "         " "              " "
## 14  ( 1 ) "*"         " "        " "         " "              " "
## 15  ( 1 ) "*"         " "        " "         " "              " "
## 16  ( 1 ) "*"         " "        "*"         " "              " "
## 17  ( 1 ) "*"         " "        "*"         " "              " "
## 18  ( 1 ) "*"         " "        "*"         " "              " "
## 19  ( 1 ) "*"         "*"        "*"         "*"              " "
## 20  ( 1 ) "*"         "*"        "*"         "*"              " "
## 21  ( 1 ) "*"         "*"        "*"         "*"              " "
## 22  ( 1 ) "*"         "*"        "*"         "*"              " "
## 23  ( 1 ) "*"         "*"        "*"         "*"              " "
## 24  ( 1 ) "*"         "*"        "*"         "*"              " "
## 25  ( 1 ) "*"         "*"        "*"         "*"              " "
## 26  ( 1 ) "*"         "*"        "*"         "*"              " "
## 27  ( 1 ) "*"         "*"        "*"         "*"              " "
## 28  ( 1 ) "*"         "*"        "*"         "*"              " "
## 29  ( 1 ) "*"         "*"        "*"         "*"              " "
## 30  ( 1 ) "*"         "*"        "*"         "*"              "*"
## 31  ( 1 ) "*"         "*"        "*"         "*"              "*"
## 32  ( 1 ) "*"         "*"        "*"         "*"              "*"
##           guardianother traveltime studytime failures schoolsupyes famsupyes
## 1  ( 1 )  " "           " "        " "       " "      " "          " "
## 2  ( 1 )  " "           " "        " "       "*"      " "          " "
## 3  ( 1 )  " "           " "        " "       "*"      " "          " "
## 4  ( 1 )  " "           " "        " "       "*"      " "          " "
## 5  ( 1 )  " "           " "        " "       "*"      " "          " "
## 6  ( 1 )  " "           " "        " "       "*"      " "          "*"
## 7  ( 1 )  " "           "*"        " "       "*"      " "          "*"
## 8  ( 1 )  " "           "*"        " "       "*"      " "          "*"
## 9  ( 1 )  " "           "*"        " "       "*"      " "          "*"
## 10  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 11  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 12  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 13  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 14  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 15  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 16  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 17  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 18  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 19  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 20  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 21  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 22  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 23  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 24  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 25  ( 1 ) " "           "*"        " "       "*"      " "          "*"
## 26  ( 1 ) " "           "*"        " "       "*"      "*"          "*"
```

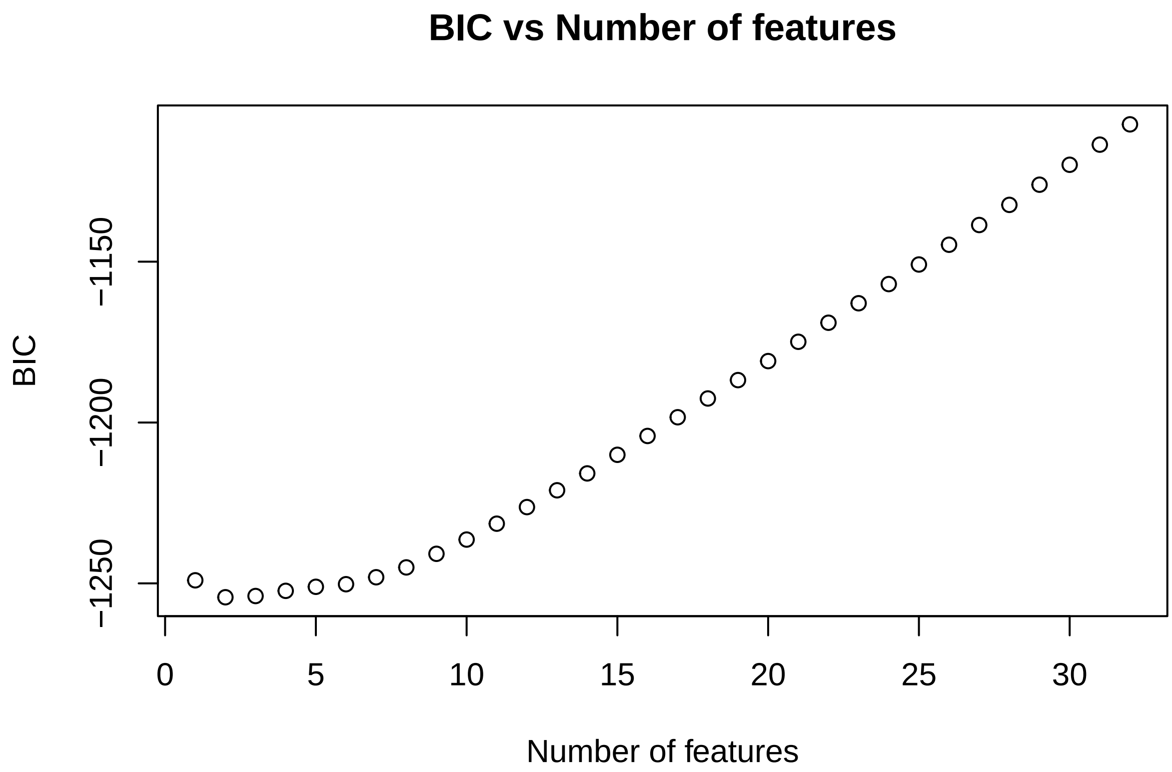
```
## 27  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
## 28  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
## 29  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
## 30  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
## 31  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
## 32  ( 1 ) " "           "*"         " "        "*"      "*"          "*"
##           paidyes activitiesyes nurseryyes higheryes internetyes romanticyes
## 1  ( 1 )  " "     " "           " "        " "       " "         " "
## 2  ( 1 )  " "     " "           " "        " "       " "         " "
## 3  ( 1 )  " "     " "           " "        " "       " "         " "
## 4  ( 1 )  " "     " "           " "        " "       " "         " "
## 5  ( 1 )  "*"     " "           " "        " "       " "         " "
## 6  ( 1 )  "*"     " "           " "        " "       " "         " "
## 7  ( 1 )  "*"     " "           " "        " "       " "         " "
## 8  ( 1 )  "*"     " "           " "        " "       " "         " "
## 9  ( 1 )  "*"     " "           " "        " "       " "         " "
## 10  ( 1 ) "*"     " "           " "        " "       " "         " "
## 11  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 12  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 13  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 14  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 15  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 16  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 17  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 18  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 19  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 20  ( 1 ) "*"     "*"           " "        " "       " "         " "
## 21  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 22  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 23  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 24  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 25  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 26  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 27  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 28  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 29  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 30  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 31  ( 1 ) "*"     "*"           " "        " "       " "         "*"
## 32  ( 1 ) "*"     "*"           " "        " "       " "         "*"
##           famrel freetime goout Dalc Walc health absences G1  G2
## 1  ( 1 )  " "    " "      " "   " "  " "  " "    " "      " " "*"
## 2  ( 1 )  " "    " "      " "   " "  " "  " "    " "      " " "*"
## 3  ( 1 )  " "    " "      " "   " "  " "  " "    " "      "*" "*"
## 4  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 5  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 6  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 7  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 8  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 9  ( 1 )  " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 10  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 11  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 12  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 13  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 14  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
```

```
## 15  ( 1 ) " "    " "      " "   " "  " "  " "    "*"      "*" "*"
## 16  ( 1 ) "*"    " "      " "   " "  " "  " "    "*"      "*" "*"
## 17  ( 1 ) "*"    " "      " "   " "  " "  " "    "*"      "*" "*"
## 18  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 19  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 20  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 21  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 22  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 23  ( 1 ) "*"    " "      " "   " "  " "  "*"    "*"      "*" "*"
## 24  ( 1 ) "*"    " "      "*"   " "  "*"  "*"    "*"      "*" "*"
## 25  ( 1 ) "*"    " "      "*"   " "  "*"  "*"    "*"      "*" "*"
## 26  ( 1 ) "*"    " "      "*"   " "  "*"  "*"    "*"      "*" "*"
## 27  ( 1 ) "*"    " "      "*"   " "  "*"  "*"    "*"      "*" "*"
## 28  ( 1 ) "*"    "*"      "*"   " "  "*"  "*"    "*"      "*" "*"
## 29  ( 1 ) "*"    " "      "*"   "*"  "*"  "*"    "*"      "*" "*"
## 30  ( 1 ) "*"    " "      "*"   "*"  "*"  "*"    "*"      "*" "*"
## 31  ( 1 ) "*"    " "      "*"   "*"  "*"  "*"    "*"      "*" "*"
## 32  ( 1 ) "*"    "*"      "*"   "*"  "*"  "*"    "*"      "*" "*"
```

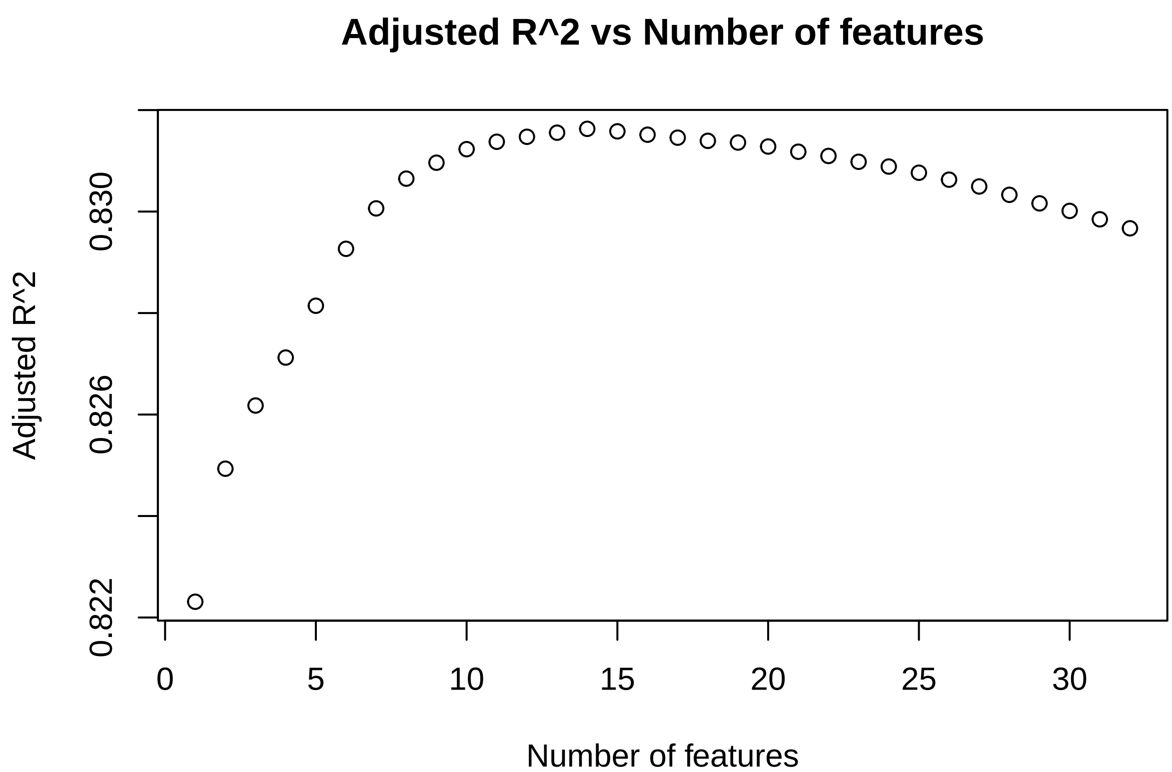
```r
regfit.adjr2 = regfit.summary$adjr2
plot(regfit.summary$cp, xlab = "Number of features", ylab = "Cp", main = "Cp vs Number of features")
```




```r
plot(regfit.summary$bic, xlab = "Number of features", ylab = "BIC", main = "BIC vs Number of features")
```




```
plot(regfit.adjr2, xlab = "Number of features", ylab = "Adjusted R^2", main = "Adjusted R^2 vs Number of
```




```r
which.max(regfit.adjr2)
```

```
## [1] 14
```

```r
which.min(regfit.summary$cp)
```

```
## [1] 10
```

```r
which.min(regfit.summary$bic)
```

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

```r
regfit.adjr2[14]
```

```
## [1] 0.83163
```

```r
coef(regfit.full, id=14)
```

```
##   (Intercept)      addressU      PstatusT    Fjobhealth     Fjobother
##   -0.55602684    0.16446304   -0.27599634   -0.47996236   -0.41070208
##  Fjobservices   Fjobteacher    traveltime      failures     famsupyes
##   -0.62719616   -0.74383734    0.20491426   -0.34479517    0.31277034
##       paidyes activitiesyes      absences            G1            G2
##   -0.37572559   -0.16167604    0.02279686    0.10364214    0.97581309
```

```r
coef(regfit.full, id=10)
```

```
##  (Intercept)     PstatusT Fjobservices  Fjobteacher   traveltime     failures
##  -0.81012477  -0.31331663  -0.25521576  -0.35933059   0.17347400  -0.33761632
##    famsupyes      paidyes     absences           G1           G2
```

```
##   0.32706672  -0.38763466   0.02279829   0.10161936   0.97692118
```

```
coef(regfit.full, id=2)
```

```
## (Intercept)    failures          G2
##  -0.3073145  -0.3336223   1.0433859
```

- The best model according to the training data is with 14 variables. The variables are address, Pstatus, Fjob, traveltime, failures, famsup, paid, activities, absences, G1, and G2. This is not a total of 14 because it has dummy variables within Fjob, and famsup. We can try to make a model with these variables. This turns out to be worse then forward selection so we should try to find the model that has the best CV.

```
bestSub14fit = lm(G3 ~ address + Pstatus + Fjob + traveltime + failures + famsup + paid + activities + a
summary(bestSub14fit)
```

```
##
## Call:
## lm(formula = G3 ~ address + Pstatus + Fjob + traveltime + failures +
##     famsup + paid + activities + absences + G1 + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5110 -0.4667  0.1350  0.8348  5.5295
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.55603    0.44635  -1.246 0.213271
## addressU       0.16446    0.14031   1.172 0.241536
## PstatusT      -0.27600    0.19825  -1.392 0.164306
## Fjobhealth    -0.47996    0.41776  -1.149 0.250980
## Fjobother     -0.41070    0.25690  -1.599 0.110337
## Fjobservices  -0.62720    0.26609  -2.357 0.018686 *
## Fjobteacher   -0.74384    0.33711  -2.207 0.027665 *
## traveltime     0.20491    0.08581   2.388 0.017201 *
## failures      -0.34480    0.09614  -3.587 0.000358 ***
## famsupyes      0.31277    0.12724   2.458 0.014202 *
## paidyes       -0.37573    0.15476  -2.428 0.015437 *
## activitiesyes -0.16168    0.12099  -1.336 0.181879
## absences       0.02280    0.00950   2.400 0.016660 *
## G1             0.10364    0.03848   2.693 0.007245 **
## G2             0.97581    0.03447  28.313  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.617 on 715 degrees of freedom
## Multiple R-squared:  0.8349, Adjusted R-squared:  0.8316
## F-statistic: 258.2 on 14 and 715 DF,  p-value: < 2.2e-16
```

```
preds = predict(bestSub14fit, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.440578
```

```
bestSub10fit = lm(G3 ~ Pstatus + Fjob + traveltime + failures + paid + famsup + absences + G1 + G2, tra
summary(bestSub10fit)
```

```
##
## Call:
## lm(formula = G3 ~ Pstatus + Fjob + traveltime + failures + paid +
##     famsup + absences + G1 + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5557 -0.4579  0.1305  0.8408  5.5619
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.473595   0.419284  -1.130 0.259051
## PstatusT     -0.318071   0.196979  -1.615 0.106806
## Fjobhealth   -0.440912   0.417212  -1.057 0.290956
## Fjobother    -0.393856   0.256652  -1.535 0.125325
## Fjobservices -0.612581   0.266094  -2.302 0.021614 *
## Fjobteacher  -0.717692   0.337035  -2.129 0.033559 *
## traveltime    0.180461   0.082118   2.198 0.028298 *
## failures     -0.341023   0.096196  -3.545 0.000418 ***
## paidyes      -0.377849   0.154392  -2.447 0.014630 *
## famsupyes     0.318290   0.127282   2.501 0.012618 *
## absences      0.022881   0.009506   2.407 0.016331 *
## G1            0.101446   0.038498   2.635 0.008593 **
## G2            0.978772   0.034441  28.419  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.619 on 717 degrees of freedom
## Multiple R-squared:  0.8341, Adjusted R-squared:  0.8313
## F-statistic: 300.4 on 12 and 717 DF,  p-value: < 2.2e-16
```

```r
preds = predict(bestSub10fit, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.433748
```

- Using validation set for best subset selection

```r
set.seed(1)
percentVal = .2
nValidation = percentVal*n
valD = sample(1:nTrain, nValidation)
subSetVal = trainData[valD,]
subSetTrain = trainData[-valD,]
```

```r
regfit.best.validate = regsubsets(G3 ~ ., subSetTrain, nvmax = nFeatures)
```

- the model with the lowest validation error is one that uses 7 variables. They are traveltime, failures, famsup, paid, absences, G1, and G2. Now we should train a model using the full training set and check the testing error.

```r
val.mat = model.matrix(G3 ~ ., subSetVal)
val.errors = rep(NA, nFeatures)
for (i in 1:(ncol(trainData) - 1)) {
  coefi = coef(regfit.best.validate, id = i)
  pred = val.mat[ , names(coefi)] %*% coefi
  val.errors[i] = mean((subSetVal$G3 - pred)^2)
```

```r
}
which.min(val.errors)
```

```
## [1] 5
```

```r
coef(regfit.best.validate, which.min(val.errors))
```

```
## (Intercept)  traveltime    failures    absences          G1          G2
## -1.07793860  0.23087825 -0.36886219  0.02389852  0.09505954  0.98074559
```

```r
coef(regfit.full, 5)
```

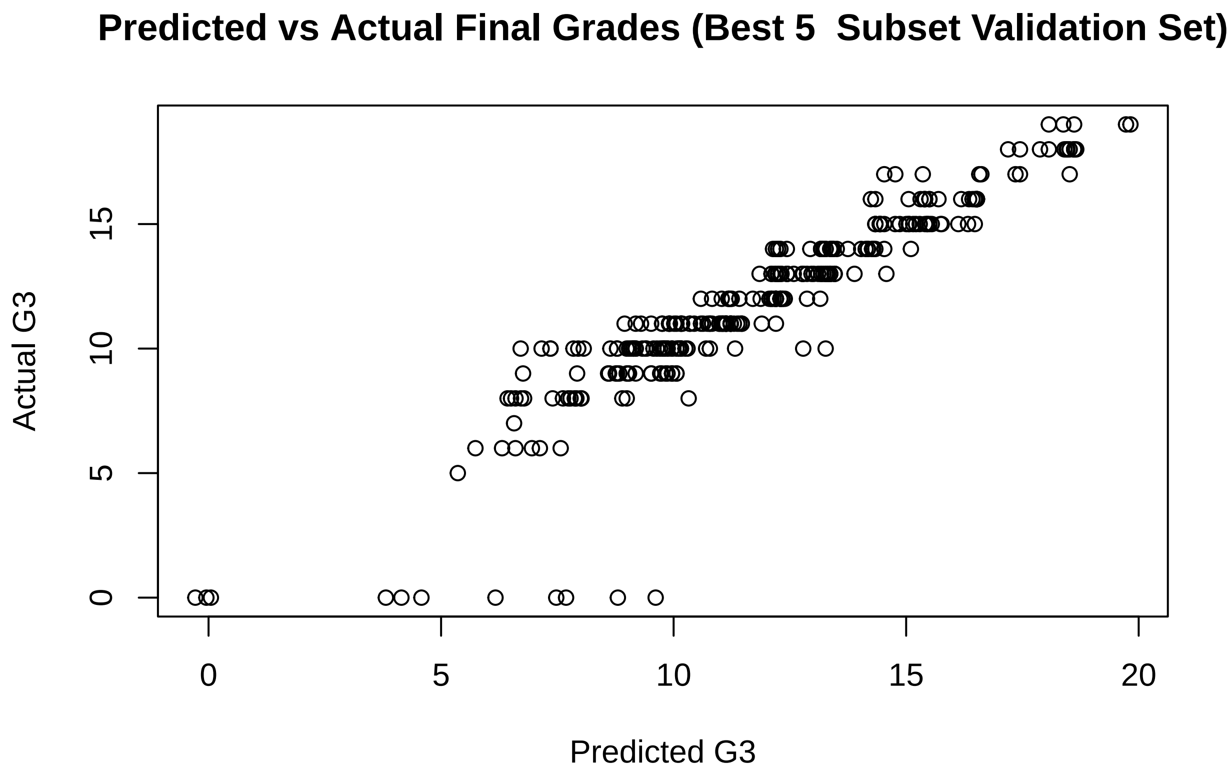
```
## (Intercept)    failures     paidyes    absences          G1          G2
## -0.62551739 -0.33412055 -0.35222242  0.02373891  0.09653667  0.97228413
```

– This has the lowest RMSE

```r
bestSub5Fit = lm(G3 ~ failures + paid + absences + G1 + G2, trainData)
summary(bestSub5Fit)
```

```
##
## Call:
## lm(formula = G3 ~ failures + paid + absences + G1 + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7286 -0.3855  0.0801  0.8270  5.9373
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.625517   0.270215  -2.315 0.020898 *
## failures    -0.334121   0.096900  -3.448 0.000597 ***
## paidyes     -0.352222   0.152885  -2.304 0.021514 *
## absences     0.023739   0.009546   2.487 0.013114 *
## G1           0.096537   0.038586   2.502 0.012574 *
## G2           0.972284   0.034538  28.151  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.634 on 724 degrees of freedom
## Multiple R-squared:  0.8293, Adjusted R-squared:  0.8281
## F-statistic: 703.6 on 5 and 724 DF,  p-value: < 2.2e-16
```

```r
pred = predict(bestSub5Fit, testData)
plot(pred, testData$G3, xlab = "Predicted G3", ylab = "Actual G3", main = "Predicted vs Actual Final Gr
```




```r
sqrt(mean((testData$G3 - pred)^2))
```

```
## [1] 1.398733
```

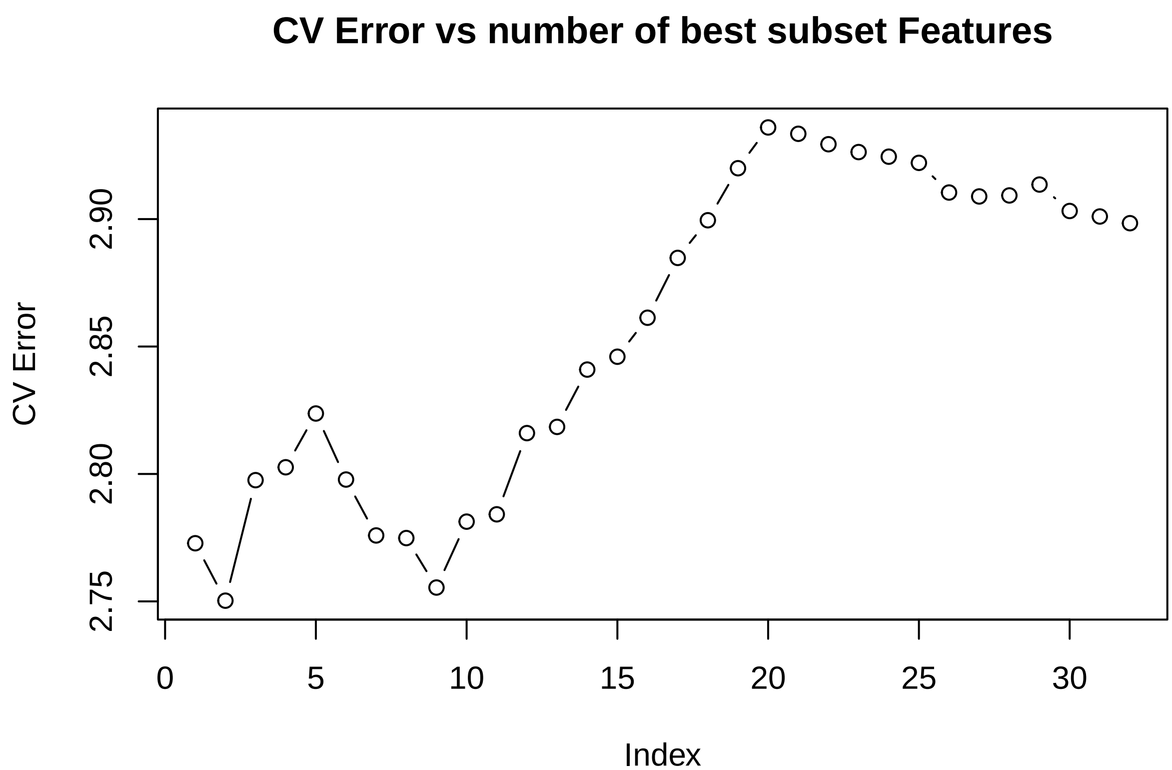
- Try doing CV with best subset. Must make prediction function. According to the output we should use 2 features for the lowest mean cv error

```r
predict.regsubsets = function(object, newData, id, ...) {
  form = as.formula(object$call[[2]])
  mat = model.matrix(form, newData)
  coefi = coef(object, id = id)
  xvars = names(coefi)
  mat[ , xvars] %*% coefi
}
# 10 fold CV
k = 10
set.seed(1)
folds = sample(rep(1:k, length = nTrain))
cv.errors = matrix(NA, k, nFeatures, dimnames = list(NULL, paste(1:nFeatures)))

for (j in 1:k) {
  best.fit = regsubsets(G3 ~ ., trainData[folds != j, ], nvmax = nFeatures)
  for (i in 1:nFeatures) {
    pred = predict(best.fit, trainData[folds == j, ], nvmax = nFeatures, id = i)
    cv.errors[j, i] = mean((trainData$G3[folds == j] - pred)^2)
  }
}
```

```r
mean.cv.errors = apply(cv.errors, 2, mean)
mean.cv.errors
```

```
##        1        2        3        4        5        6        7        8
## 2.772813 2.750289 2.797568 2.802625 2.823720 2.797807 2.775868 2.774846
##        9       10       11       12       13       14       15       16
## 2.755433 2.781296 2.784160 2.816028 2.818463 2.840950 2.845994 2.861294
##       17       18       19       20       21       22       23       24
## 2.884765 2.899538 2.919952 2.935950 2.933457 2.929371 2.926291 2.924485
##       25       26       27       28       29       30       31       32
## 2.922064 2.910421 2.908880 2.909273 2.913549 2.903176 2.901019 2.898393
```

```r
par(mfrow = c(1,1))
plot(mean.cv.errors, type= "b", ylab = "CV Error", main = "CV Error vs number of best subset Features")
```




```r
which.min(mean.cv.errors)
```

```
## 2
## 2
```

```r
coef(regfit.full, which.min(mean.cv.errors))
```

```
## (Intercept)    failures          G2
##  -0.3073145  -0.3336223   1.0433859
```

```r
bestSub2fit = lm(G3 ~ failures + G2, trainData)
summary(bestSub2fit)
```

```
##
## Call:
## lm(formula = G3 ~ failures + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7929 -0.3001 -0.0398  0.8301  5.8735
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.30731    0.23818   -1.29 0.197378
## failures    -0.33362    0.09670   -3.45 0.000593 ***
## G2           1.04339    0.01976   52.82  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.649 on 727 degrees of freedom
## Multiple R-squared:  0.8254, Adjusted R-squared:  0.8249
## F-statistic:  1719 on 2 and 727 DF,  p-value: < 2.2e-16
```

```r
pred = predict(bestSub2fit, testData)
sqrt(mean((testData$G3 - pred)^2))
```

```
## [1] 1.441658
```

```r
coef(regfit.full, 9)
```

```
##  (Intercept)     PstatusT Fjobservices   traveltime     failures    famsupyes
##  -0.80204721  -0.30128171  -0.22236350   0.17420908  -0.33391953   0.30884703
##      paidyes     absences           G1           G2
##  -0.38027783   0.02280147   0.09592451   0.97846994
```

```r
bestSub9fit <- lm(G3 ~ Pstatus + Fjob + traveltime + failures + famsup + paid + absences + G1 + G2, data
summary(bestSub9fit)
```

```
##
## Call:
## lm(formula = G3 ~ Pstatus + Fjob + traveltime + failures + famsup +
##     paid + absences + G1 + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5557 -0.4579  0.1305  0.8408  5.5619
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.473595   0.419284  -1.130 0.259051
## PstatusT     -0.318071   0.196979  -1.615 0.106806
## Fjobhealth   -0.440912   0.417212  -1.057 0.290956
## Fjobother    -0.393856   0.256652  -1.535 0.125325
## Fjobservices -0.612581   0.266094  -2.302 0.021614 *
## Fjobteacher  -0.717692   0.337035  -2.129 0.033559 *
## traveltime    0.180461   0.082118   2.198 0.028298 *
## failures     -0.341023   0.096196  -3.545 0.000418 ***
## famsupyes     0.318290   0.127282   2.501 0.012618 *
## paidyes      -0.377849   0.154392  -2.447 0.014630 *
```

```
## absences      0.022881   0.009506   2.407 0.016331 *
## G1            0.101446   0.038498   2.635 0.008593 **
## G2            0.978772   0.034441  28.419  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.619 on 717 degrees of freedom
## Multiple R-squared:  0.8341, Adjusted R-squared:  0.8313
## F-statistic: 300.4 on 12 and 717 DF,  p-value: < 2.2e-16
```

```
pred = predict(bestSub9fit, newdata = testData)
sqrt(mean((testData$G3 - pred)^2))
```

```
## [1] 1.433748
```

- Backward Selection

```
fullFit = lm(G3 ~ ., trainData)
backward_model = step(fullFit, direction = "backward", trace = F)
summary(backward_model)
```

```
##
## Call:
## lm(formula = G3 ~ Pstatus + traveltime + failures + famsup +
##     paid + absences + G1 + G2, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7161 -0.4484  0.1183  0.7976  5.4141
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.845936   0.358734  -2.358  0.01863 *
## PstatusT    -0.337606   0.196082  -1.722  0.08554 .
## traveltime   0.176224   0.081627   2.159  0.03119 *
## failures    -0.341178   0.096272  -3.544  0.00042 ***
## famsupyes    0.318997   0.126149   2.529  0.01166 *
## paidyes     -0.388332   0.154445  -2.514  0.01214 *
## absences     0.022916   0.009527   2.405  0.01640 *
## G1           0.098088   0.038365   2.557  0.01077 *
## G2           0.976470   0.034464  28.333  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.623 on 721 degrees of freedom
## Multiple R-squared:  0.8324, Adjusted R-squared:  0.8305
## F-statistic: 447.6 on 8 and 721 DF,  p-value: < 2.2e-16
```

```
preds = predict(backward_model, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.419982
```

- Forward Selection

```
nullFit = lm(G3 ~ 1, trainData)
fullFit = lm(G3 ~ ., trainData)
```

```r
forward_model = step(nullFit, scope = list(lower = nullFit, upper = fullFit), direction = "forward", tra
summary(forward_model)
```

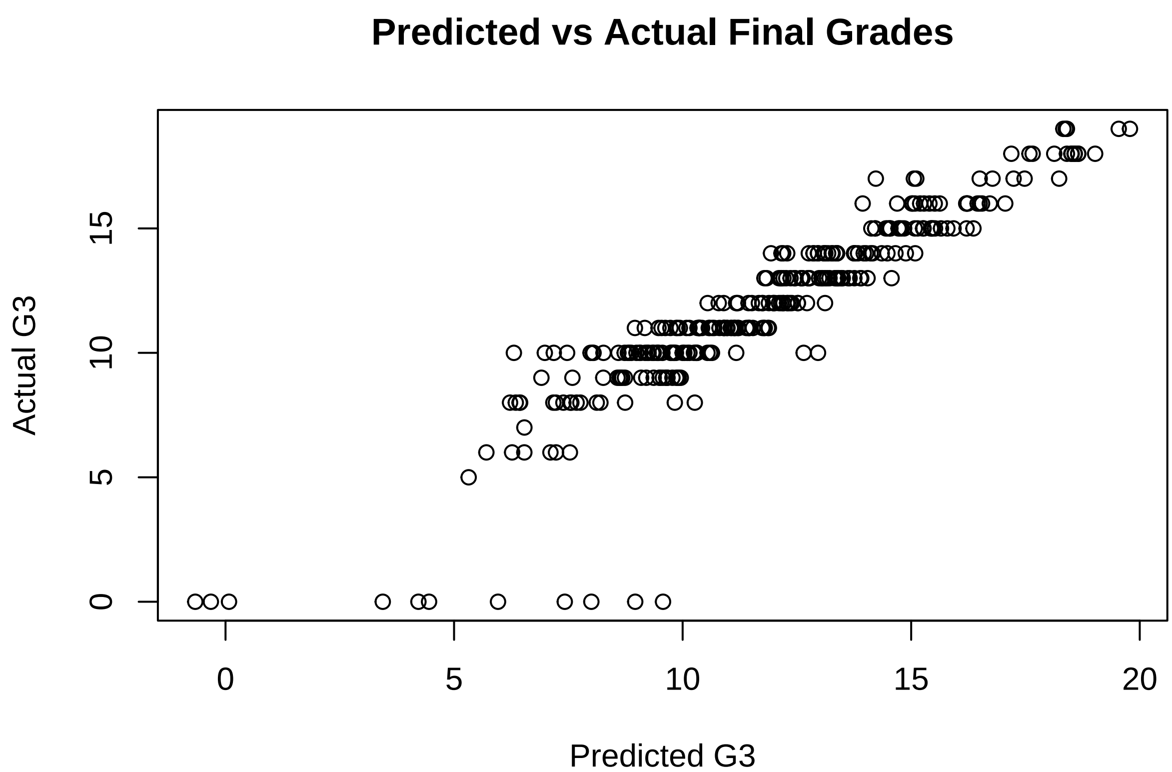
```
##
## Call:
## lm(formula = G3 ~ G2 + failures + G1 + absences + paid + famsup +
##     traveltime + Pstatus, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7161 -0.4484  0.1183  0.7976  5.4141
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.845936   0.358734  -2.358  0.01863 *
## G2           0.976470   0.034464  28.333  < 2e-16 ***
## failures    -0.341178   0.096272  -3.544  0.00042 ***
## G1           0.098088   0.038365   2.557  0.01077 *
## absences     0.022916   0.009527   2.405  0.01640 *
## paidyes     -0.388332   0.154445  -2.514  0.01214 *
## famsupyes    0.318997   0.126149   2.529  0.01166 *
## traveltime   0.176224   0.081627   2.159  0.03119 *
## PstatusT    -0.337606   0.196082  -1.722  0.08554 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.623 on 721 degrees of freedom
## Multiple R-squared:  0.8324, Adjusted R-squared:  0.8305
## F-statistic: 447.6 on 8 and 721 DF,  p-value: < 2.2e-16
```

```r
preds = predict(forward_model, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.419982
```

- The forward and backward selection give the same output so it would be the same rmse.

```r
plot(preds, testData$G3, xlab = "Predicted G3", ylab = "Actual G3", main = "Predicted vs Actual Final G
```



- Noticed that the number of absences increases the predicted final grade, which should not be the case, so we should a model without the absences.

```
forward_model_noAbsence = lm(formula = G3 ~ G2 + failures + G1 + paid + famsup +
    traveltime + Pstatus, data = trainData)
summary(forward_model_noAbsence)
```

```
##
## Call:
## lm(formula = G3 ~ G2 + failures + G1 + paid + famsup + traveltime +
##     Pstatus, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8243 -0.4128  0.1273  0.8095  5.3394
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.68230    0.35339  -1.931 0.053910 .
## G2           0.97703    0.03458  28.256  < 2e-16 ***
## failures    -0.32195    0.09626  -3.345 0.000866 ***
## G1           0.09522    0.03847   2.475 0.013557 *
## paidyes     -0.34518    0.15391  -2.243 0.025214 *
## famsupyes    0.31719    0.12656   2.506 0.012425 *
## traveltime   0.17145    0.08187   2.094 0.036602 *
## PstatusT    -0.38252    0.19584  -1.953 0.051177 .
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.628 on 722 degrees of freedom
## Multiple R-squared:  0.831,  Adjusted R-squared:  0.8294
## F-statistic: 507.3 on 7 and 722 DF,  p-value: < 2.2e-16
```

```
preds = predict(forward_model_noAbsence, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.430864
```

- Try ridge regression

```
#install.packages("glmnet")
library(glmnet)
```

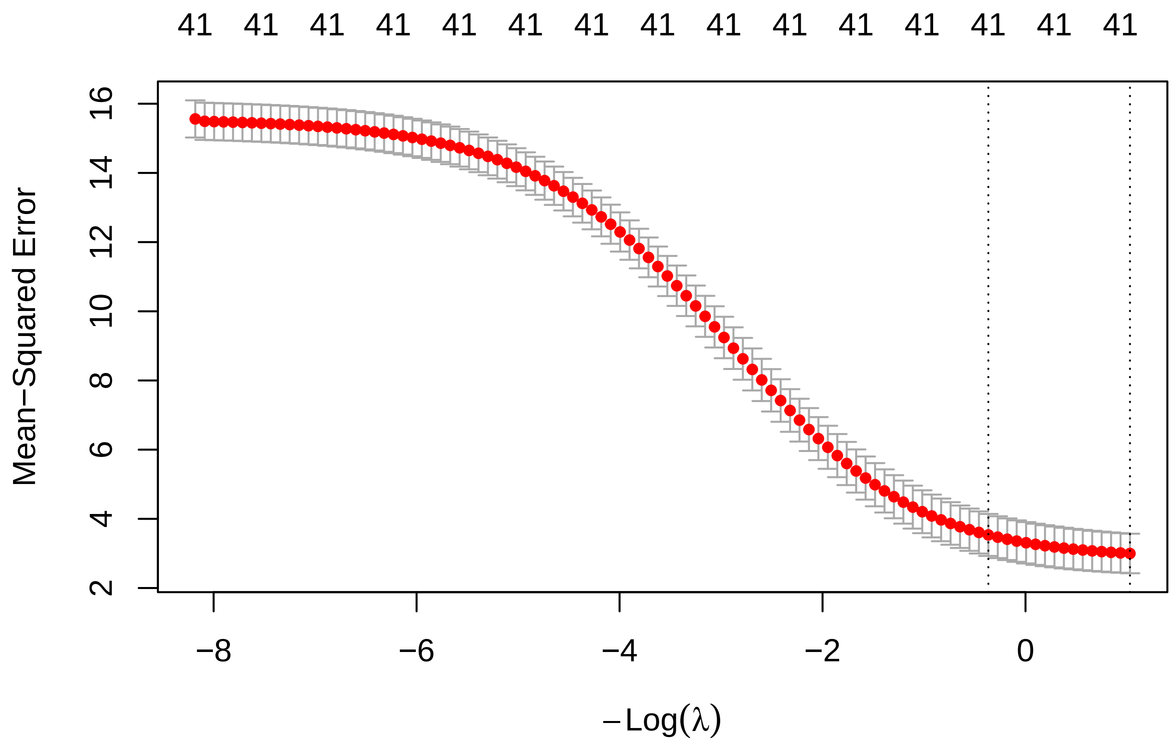
```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-10
```

```
x = model.matrix(G3 ~ ., all_data)[, -1]
y = all_data$G3
grid = 10^seq(10, -2, length = 100)
ridge.mod = glmnet(x, y, alpha = 0, lambda = grid)
dim(coef(ridge.mod))
```

```
## [1]  42 100
```

```
set.seed(1)
cv.out = cv.glmnet(x[trainIdx, ], y[trainIdx], alpha = 0)
plot(cv.out)
```



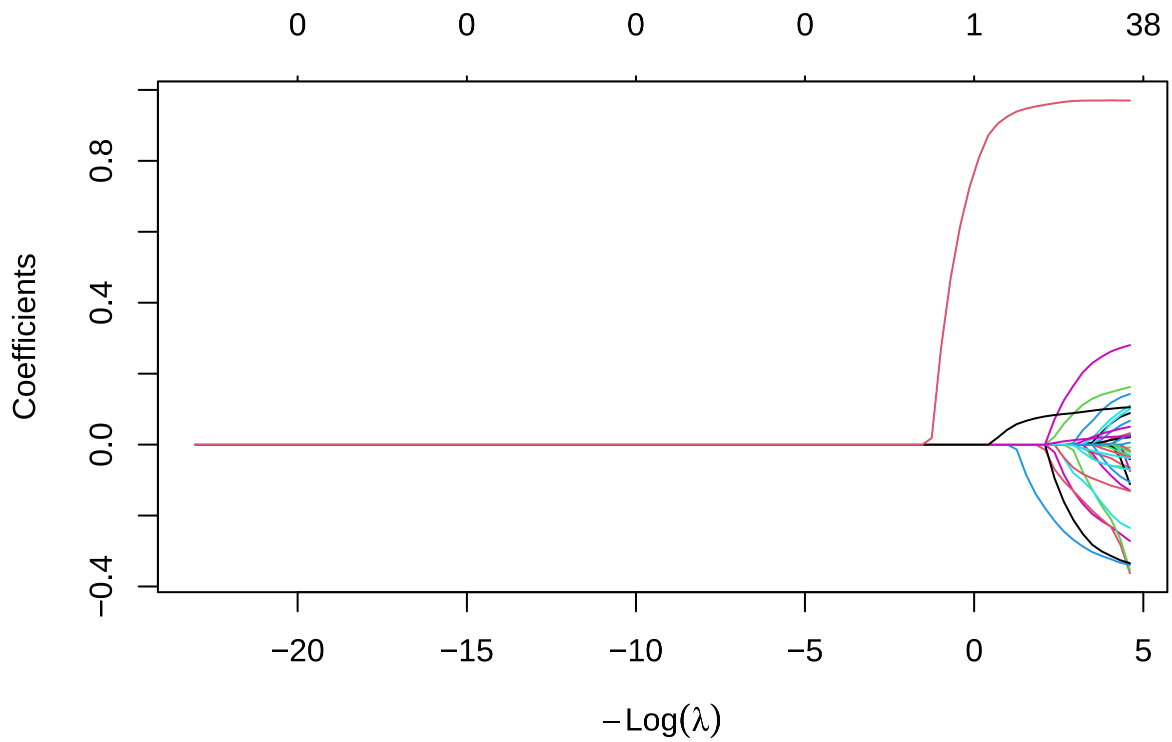

```r
bestlam = cv.out$lambda.min
bestlam
```

```
## [1] 0.3572372
```

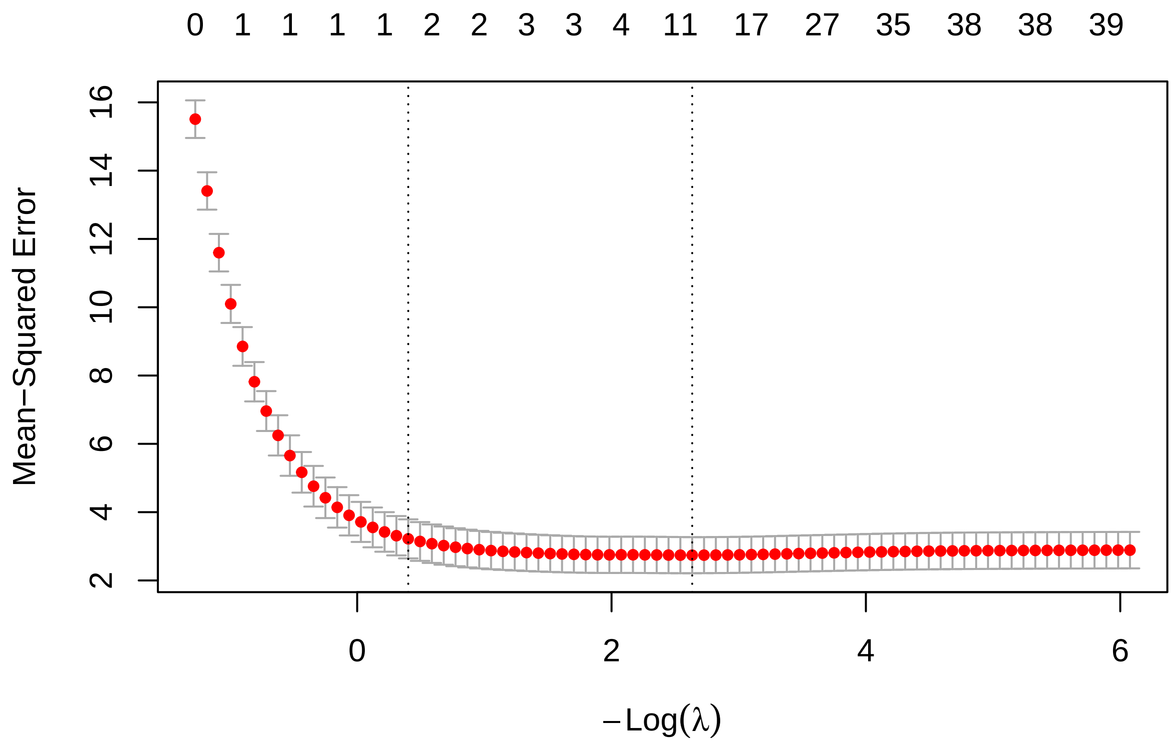
```r
ridge.pred = predict(ridge.mod, s = bestlam, newx = x[-trainIdx, ])
sqrt(mean((ridge.pred - y[-trainIdx])^2))
```

```
## [1] 1.408071
```

```r
lasso.mod = glmnet(x[trainIdx, ], y[trainIdx], alpha = 1, lambda = grid)
plot(lasso.mod)
```




```
set.seed(1)
cv.out = cv.glmnet(x[trainIdx, ], y[trainIdx], alpha = 1)
plot(cv.out)
```




```r
bestlam = cv.out$lambda.min
bestlam
```

```
## [1] 0.07177728
```

```r
lasso.pred = predict(lasso.mod, s = bestlam, newx = x[-trainIdx, ])
sqrt(mean((lasso.pred - y[-trainIdx])^2))
```

```
## [1] 1.413576
```

- lasso gives a close rsme to the best 5 subset but it is still worse then it. We will now try different models.

```r
out = glmnet(x, y, alpha = 1, lambda = grid)
lasso.coef = predict(out, type = "coefficients", s = bestlam)[1:(nFeatures + 1), ]
lasso.coef
```

```
##      (Intercept)        schoolMS            sexM             age
##      -0.60825163      0.00000000      0.00000000      0.00000000
##         addressU       famsizeLE3        PstatusT            Medu
##       0.00000000      0.00000000     -0.01148485      0.00000000
##             Fedu      Mjobhealth       Mjobother    Mjobservices
##       0.00000000      0.00000000      0.00000000      0.00000000
##      Mjobteacher      Fjobhealth       Fjobother    Fjobservices
##       0.00000000      0.00000000      0.00000000     -0.03553269
##      Fjobteacher      reasonhome     reasonother reasonreputation
##       0.00000000      0.00000000      0.00000000      0.00000000
##   guardianmother   guardianother      traveltime       studytime
##       0.00000000      0.00000000      0.02242678      0.00000000
```

```
##         failures     schoolsupyes        famsupyes          paidyes
##      -0.17726388       0.00000000       0.01082511      -0.16640612
##    activitiesyes       nurseryyes        higheryes      internetyes
##       0.00000000       0.00000000       0.00000000       0.00000000
##      romanticyes
##       0.00000000
```

```
lasso.coef[lasso.coef != 0]
```

```
##  (Intercept)     PstatusT Fjobservices   traveltime     failures    famsupyes
## -0.60825163  -0.01148485  -0.03553269   0.02242678  -0.17726388   0.01082511
##      paidyes
## -0.16640612
```

```
xTest = model.matrix(G3 ~ ., testData)[ , -1]
pred = predict(out, s = bestlam, newx = xTest)
sqrt(mean((testData$G3 - pred)^2))
```

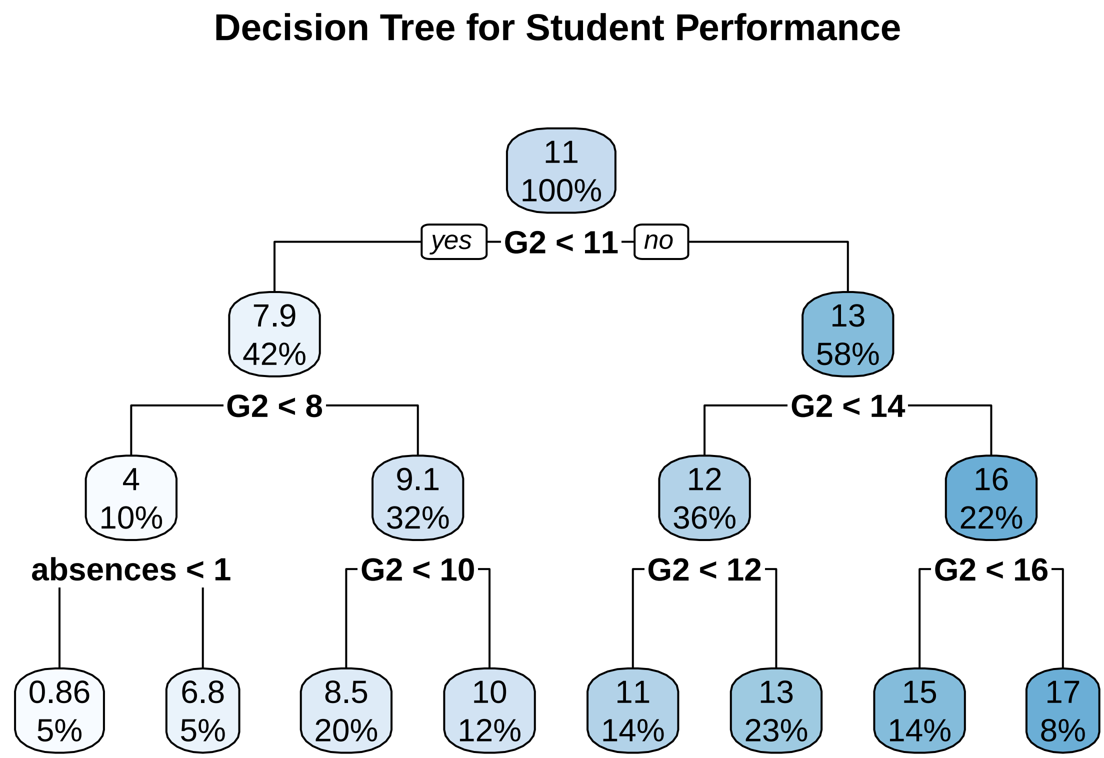
```
## [1] 1.400898
```

-Decision Tree

```
library(rpart)
library(rpart.plot)
tree_model = rpart(G3 ~ ., data = trainData, method = "anova")
print(tree_model)
```

```
## n= 730
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 730 11325.87000 11.1178100
##    2) G2< 10.5 306  3781.67300  7.8562090
##      4) G2< 7.5 76   937.94740  3.9736840
##        8) absences< 1 36   214.30560  0.8611111 *
##        9) absences>=1 40    60.97500  6.7750000 *
##      5) G2>=7.5 230  1319.54800  9.1391300
##       10) G2< 9.5 146   894.49320  8.5068490 *
##       11) G2>=9.5 84   265.23810 10.2381000 *
##    3) G2>=10.5 424  1939.66000 13.4717000
##      6) G2< 13.5 264   384.35980 12.1325800
##       12) G2< 11.5 99    66.02020 11.1414100 *
##       13) G2>=11.5 165   162.72730 12.7272700 *
##      7) G2>=13.5 160   300.74370 15.6812500
##       14) G2< 15.5 101    73.80198 14.8910900 *
##       15) G2>=15.5 59    55.93220 17.0339000 *
```

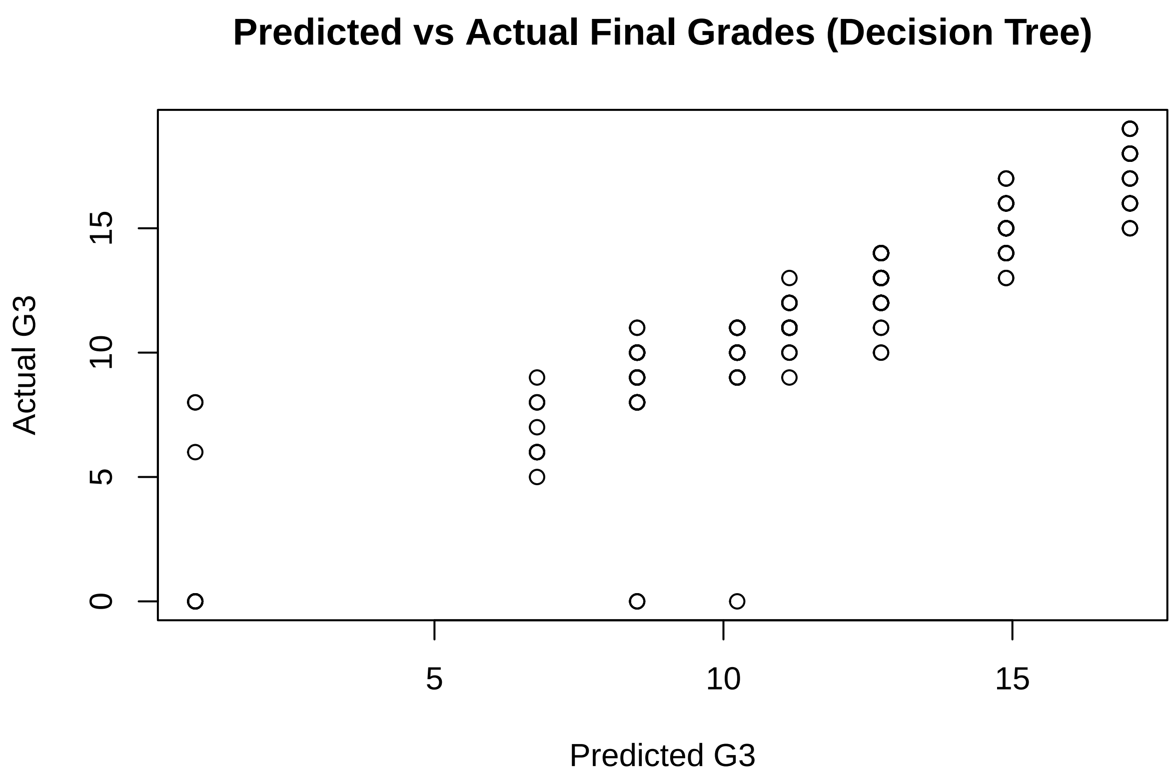
```
rpart.plot(tree_model, main = "Decision Tree for Student Performance")
```




```
tree_pred = predict(tree_model, newdata = testData)
tree_rmse = sqrt(mean((testData$G3 - tree_pred)^2))
cat("Decision Tree RMSE:", tree_rmse, "\n")
```

```
## Decision Tree RMSE: 1.532331
```

```
plot(tree_pred, testData$G3, xlab = "Predicted G3", ylab = "Actual G3", main = "Predicted vs Actual Fina
```




```r
printcp(tree_model)
```

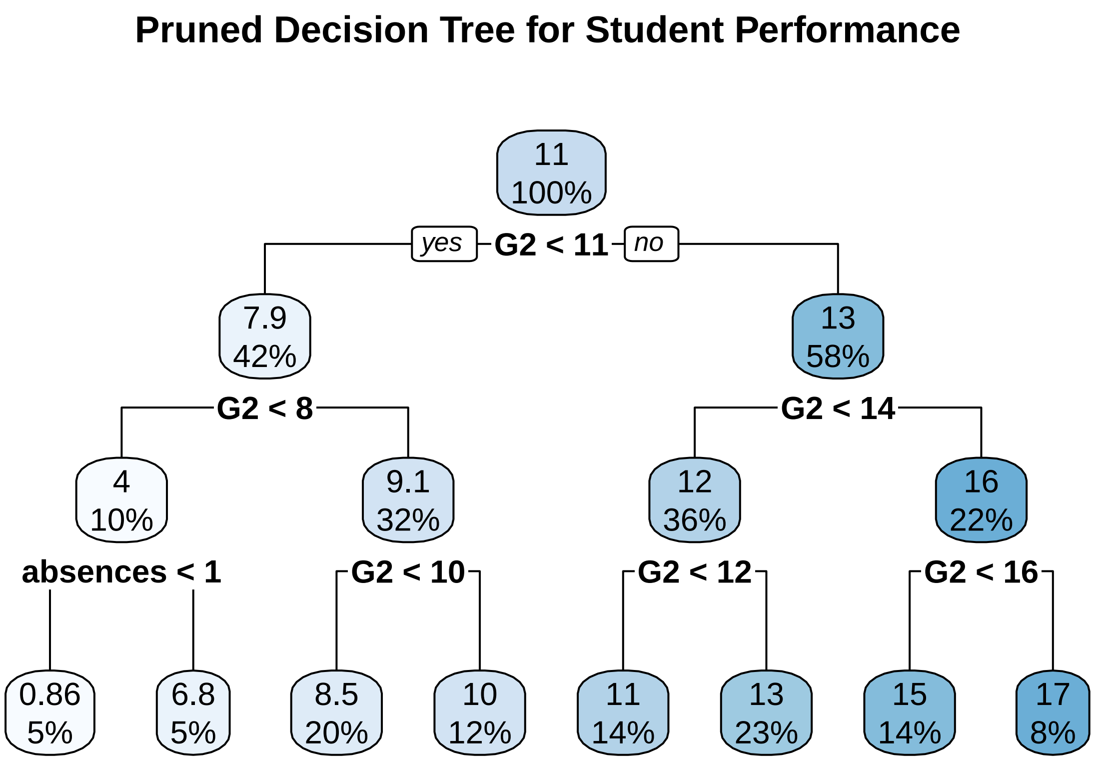
```
##
## Regression tree:
## rpart(formula = G3 ~ ., data = trainData, method = "anova")
##
## Variables actually used in tree construction:
## [1] absences G2
##
## Root node error: 11326/730 = 15.515
##
## n= 730
##
##          CP nsplit rel error  xerror     xstd
## 1 0.494844      0   1.00000 1.00197 0.069863
## 2 0.134575      1   0.50516 0.51508 0.036031
## 3 0.110769      2   0.37058 0.39734 0.028796
## 4 0.058509      3   0.25981 0.26677 0.026359
## 5 0.015099      4   0.20130 0.20851 0.026481
## 6 0.014111      5   0.18620 0.19409 0.026306
## 7 0.013740      6   0.17209 0.18688 0.026532
## 8 0.010000      7   0.15835 0.16998 0.025799
```

```r
printcp(tree_model)
```

```
##
## Regression tree:
```

```
## rpart(formula = G3 ~ ., data = trainData, method = "anova")
##
## Variables actually used in tree construction:
## [1] absences G2
##
## Root node error: 11326/730 = 15.515
##
## n= 730
##
##          CP nsplit rel error  xerror     xstd
## 1 0.494844      0   1.00000 1.00197 0.069863
## 2 0.134575      1   0.50516 0.51508 0.036031
## 3 0.110769      2   0.37058 0.39734 0.028796
## 4 0.058509      3   0.25981 0.26677 0.026359
## 5 0.015099      4   0.20130 0.20851 0.026481
## 6 0.014111      5   0.18620 0.19409 0.026306
## 7 0.013740      6   0.17209 0.18688 0.026532
## 8 0.010000      7   0.15835 0.16998 0.025799
```

```r
optimal_cp = tree_model$cptable[which.min(tree_model$cptable[, "xerror"]), "CP"]
pruned_tree = prune(tree_model, cp = optimal_cp)
rpart.plot(pruned_tree, main = "Pruned Decision Tree for Student Performance")
```




```r
pruned_pred = predict(pruned_tree, newdata = testData)
pruned_rmse = sqrt(mean((testData$G3 - pruned_pred)^2))
cat("Pruned Decision Tree RMSE:", pruned_rmse, "\n")
```

```
## Pruned Decision Tree RMSE: 1.532331
```

Lets try using the two the two features that the decision tree used for a linear regression It was worse then the best

```r
twoFeaturelm = lm(G3 ~ absences + G2, data = trainData)

twoFeaturePreds = predict(twoFeaturelm, testData)
sqrt(mean((testData$G3 - twoFeaturePreds)^2))
```

```
## [1] 1.424654
```

```r
#install.packages("randomForest")
library(randomForest)
```

```
## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.
```

```r
set.seed(1)
bag.grade = randomForest(G3 ~ ., trainData, mtry = nFeatures, importance = TRUE)
bag.grade
```

```
##
## Call:
##  randomForest(formula = G3 ~ ., data = trainData, mtry = nFeatures,      importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 32
##
##           Mean of squared residuals: 2.578824
##                     % Var explained: 83.38
```

```r
summary(bag.grade)
```

```
##                 Length Class  Mode
## call              5    -none- call
## type              1    -none- character
## predicted       730    -none- numeric
## mse             500    -none- numeric
## rsq             500    -none- numeric
## oob.times       730    -none- numeric
## importance       64    -none- numeric
## importanceSD     32    -none- numeric
## localImportance   0    -none- NULL
## proximity         0    -none- NULL
## ntree             1    -none- numeric
## mtry              1    -none- numeric
## forest           11    -none- list
## coefs             0    -none- NULL
## y               730    -none- numeric
## test              0    -none- NULL
## inbag             0    -none- NULL
## terms             3    terms  call
```

```r
bag.preds = predict(bag.grade, testData)
sqrt(mean((testData$G3 - bag.preds)^2))
```

```
## [1] 1.410638
```

```
nFeatures
```

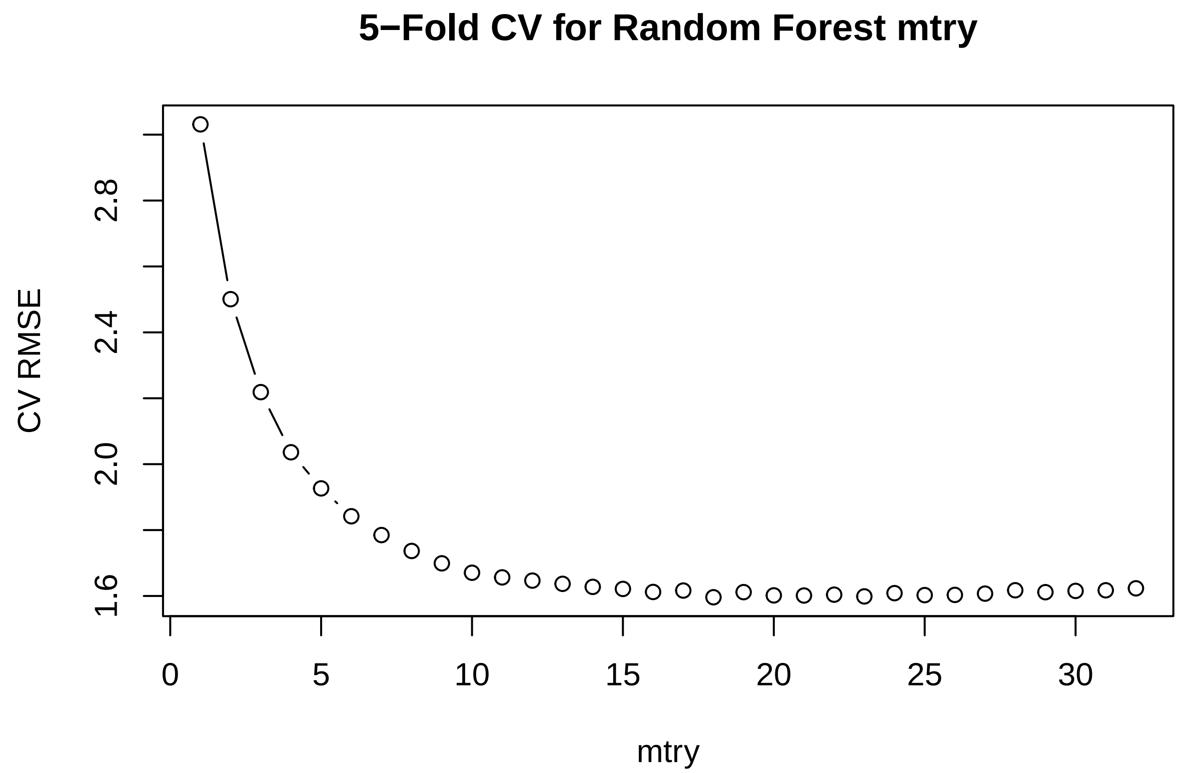
```
## [1] 32
```

Do CV to find the best number of variables to consider

```
k = 5
set.seed(1)

folds = sample(rep(1:k, length = nTrain))
cv.errors = matrix(NA, k, nFeatures, dimnames = list(NULL, paste(1:nFeatures)))

for (j in 1:k) {
  train_fold = trainData[folds != j, ]
  val_fold = trainData[folds == j, ]
  for (i in 1:nFeatures) {
    cvModel = randomForest(G3 ~ ., data = train_fold, mtry = i, importance = TRUE)
    preds = predict(cvModel, val_fold)
    cv.errors[j, i] = sqrt(mean((val_fold$G3 - preds)^2))
  }
}

mean.cv.errors = apply(cv.errors, 2, mean)
mean.cv.errors
```

```
##        1        2        3        4        5        6        7        8
## 3.031255 2.500724 2.218642 2.036103 1.926500 1.841903 1.784860 1.736578
##        9       10       11       12       13       14       15       16
## 1.699038 1.670485 1.656379 1.646655 1.636822 1.627610 1.621393 1.612126
##       17       18       19       20       21       22       23       24
## 1.616449 1.596153 1.611678 1.601829 1.601387 1.603941 1.598686 1.608574
##       25       26       27       28       29       30       31       32
## 1.602603 1.603243 1.607095 1.617302 1.611540 1.615390 1.617281 1.623264
```

```
par(mfrow = c(1,1))
plot(1:nFeatures, mean.cv.errors, type = "b", xlab = "mtry", ylab = "CV RMSE", main = "5-Fold CV for Ra
```




```
best_mtry = which.min(mean.cv.errors)
best_mtry
```

```
## 18
## 18
```

```
best_mtry_model = randomForest(G3 ~ ., data = trainData, mtry = best_mtry, importance = TRUE)
preds = predict(best_mtry_model, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.376701
```

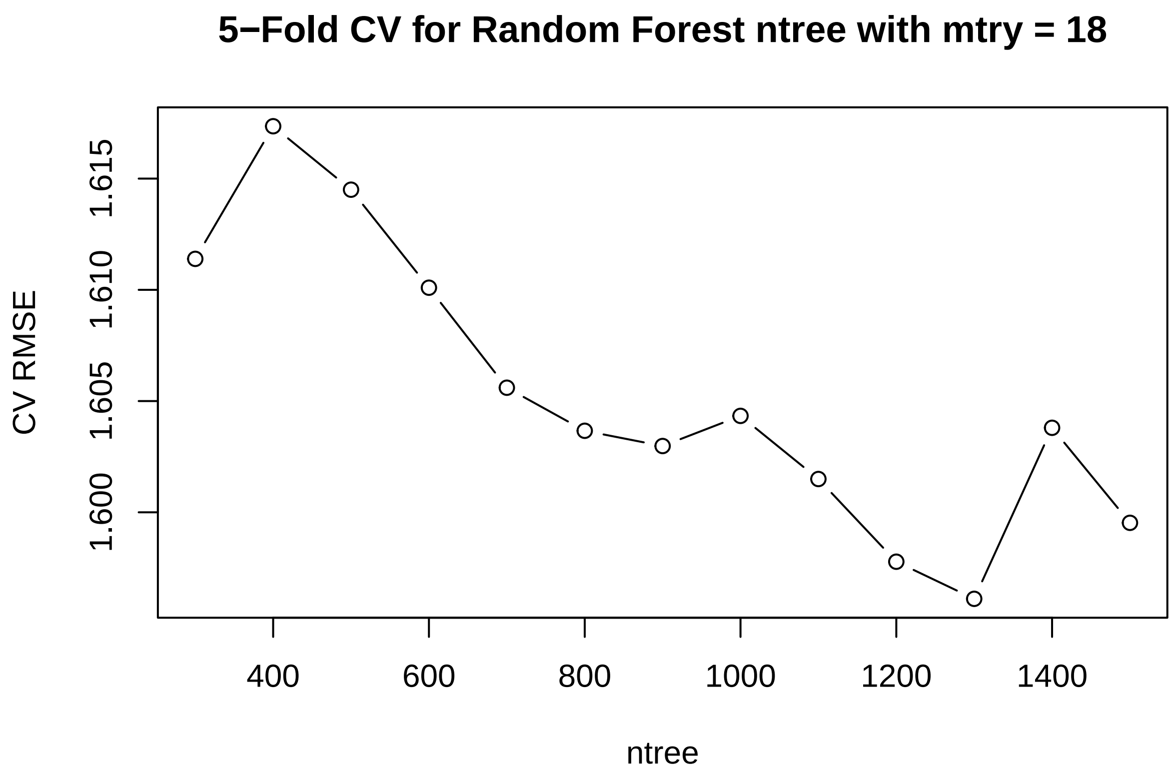
5-fold cv for ntree

```
k = 5
set.seed(1)

num_tree = c(300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500)
folds = sample(rep(1:k, length = nTrain))
cv.errors = matrix(NA, k, length(num_tree), dimnames = list(NULL, paste(num_tree)))

for (j in 1:k) {
  train_fold = trainData[folds != j, ]
  val_fold = trainData[folds == j, ]
  for (i in 1:length(num_tree)) {
    cvModel = randomForest(G3 ~ ., data = train_fold, mtry = best_mtry, ntree = num_tree[i], importance
    preds = predict(cvModel, val_fold)
    cv.errors[j, i] = sqrt(mean((val_fold$G3 - preds)^2))
```

```r
  }
}

mean.cv.errors = apply(cv.errors, 2, mean)
mean.cv.errors
```

```
##      300      400      500      600      700      800      900     1000
## 1.611392 1.617354 1.614502 1.610096 1.605600 1.603665 1.602978 1.604334
##     1100     1200     1300     1400     1500
## 1.601495 1.597779 1.596110 1.603798 1.599526
```

```r
set.seed(1)
best_mtry = 18
par(mfrow = c(1,1))
plot(num_tree, mean.cv.errors, type = "b", xlab = "ntree", ylab = "CV RMSE", main = "5-Fold CV for Rand
```




```r
best_ntree = num_tree[which.min(mean.cv.errors)]
best_ntree
```

```
## [1] 1300
```

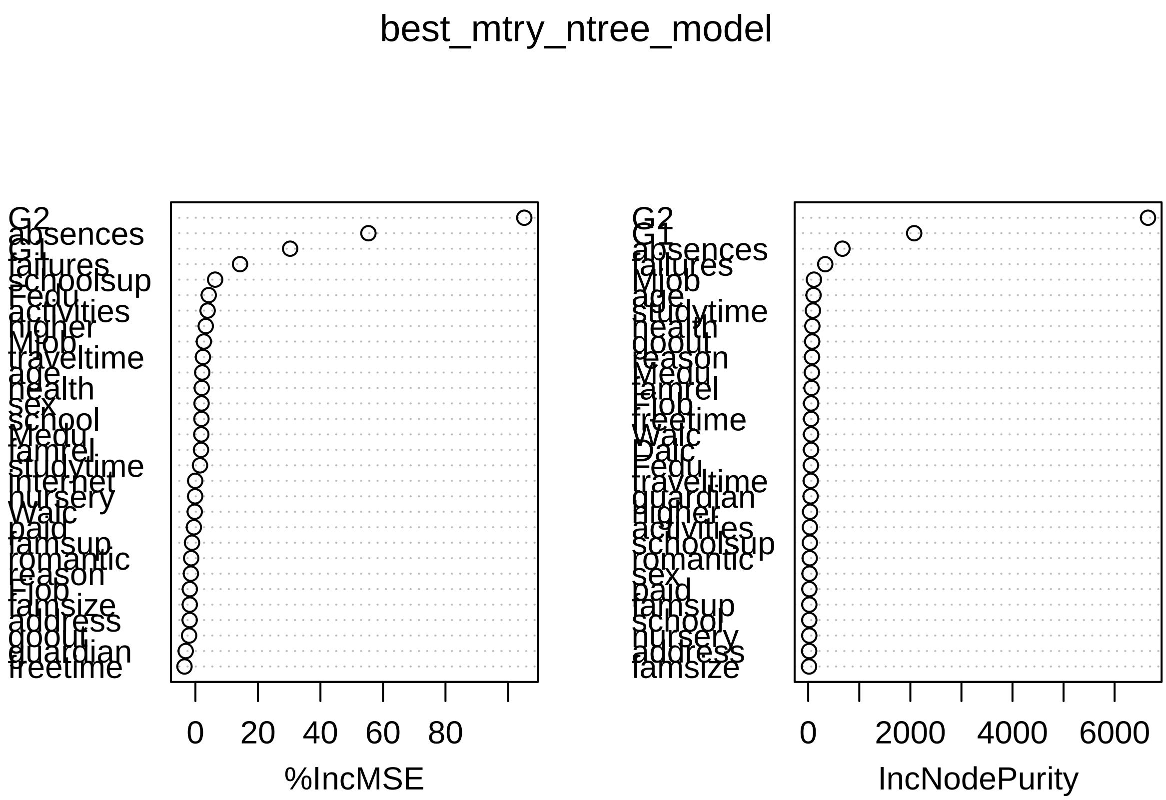
```r
best_mtry_ntree_model = randomForest(G3 ~ ., data = trainData, mtry = best_mtry, ntree = best_ntree, im
preds = predict(best_mtry_ntree_model, testData)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.366702
```

```r
importance(best_mtry_ntree_model)
```

```
##                %IncMSE IncNodePurity
## school      1.92094201     20.541576
## sex         1.96421389     25.294556
## age         2.17470514    104.078247
## address    -1.83767197     18.487838
## famsize    -1.80755593     15.973303
## Pstatus    -3.76015357      6.642354
## Medu        1.87883590     71.921914
## Fedu        4.25690818     52.998818
## Mjob        2.70042893    112.209146
## Fjob       -1.79803929     57.381035
## reason     -1.45861900     74.740303
## guardian   -3.11023809     46.798700
## traveltime  2.39247331     49.250670
## studytime   1.46361460     92.476565
## failures   14.27141506    333.475364
## schoolsup   6.30013103     32.399816
## famsup     -1.08890145     21.256512
## paid       -0.49984802     22.049340
## activities  3.90476080     33.147660
## nursery    -0.09296337     20.518667
## higher      3.30714037     36.405934
## internet   -0.07230990     14.718405
## romantic   -1.36424888     29.196371
## famrel      1.78780560     64.745881
## freetime   -3.50070337     56.861962
## goout      -2.02397436     77.551666
## Dalc       -5.03100753     54.966561
## Walc       -0.22194110     56.149429
## health      2.02594611     80.793628
## absences   55.34964853    669.920630
## G1         30.29733386   2075.478004
## G2        105.21044518   6654.193589
```

```r
varImpPlot(best_mtry_ntree_model)
```



try only using the best 4 features that are used G2, G1, absences and failures and the current best ntree

```
set.seed(1)
mostImportFeaturesRandomForeset = c('G2','G1','absences','failures','G3')
mostImportTrain = trainData[, mostImportFeaturesRandomForeset]
mostImportTest = testData[, mostImportFeaturesRandomForeset]
mostImportModel = randomForest(G3 ~ ., mostImportTrain, mtry = ncol(mostImportTrain) - 1, ntrees = best_
preds = predict(mostImportModel, mostImportTest)
sqrt(mean((mostImportTest$G3 - preds)^2))
```

```
## [1] 1.544647
```

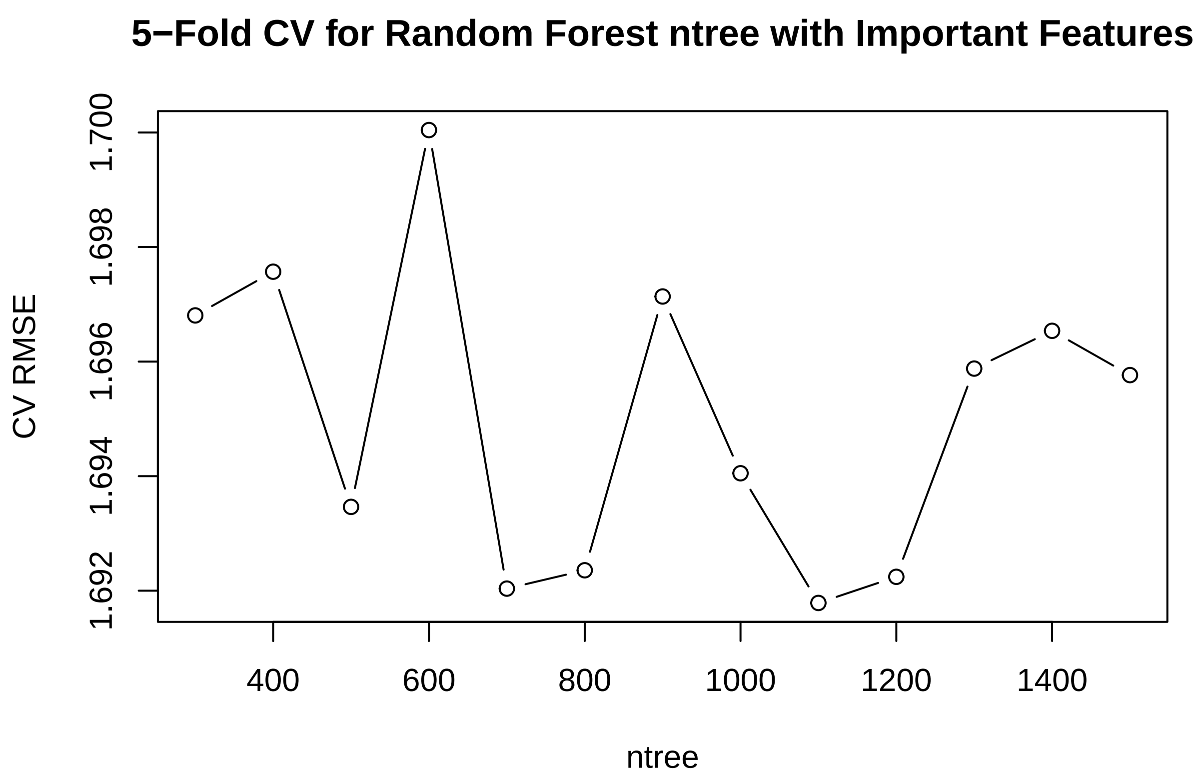
Try k-fold cv for number of trees using this subset when only consider these features for splits

```
k = 5
set.seed(1)

num_tree = c(300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500)
folds = sample(rep(1:k, length = nTrain))
cv.errors = matrix(NA, k, length(num_tree), dimnames = list(NULL, paste(num_tree)))

for (j in 1:k) {
  train_fold = mostImportTrain[folds != j, ]
  val_fold = mostImportTrain[folds == j, ]
  for (i in 1:length(num_tree)) {
    cvModel = randomForest(G3 ~ ., data = train_fold, mtry = 4, ntree = num_tree[i], importance = TRUE)
    preds = predict(cvModel, val_fold)
    cv.errors[j, i] = sqrt(mean((val_fold$G3 - preds)^2))
```

```
  }
}

mean.cv.errors = apply(cv.errors, 2, mean)
mean.cv.errors
```

```
##      300      400      500      600      700      800      900     1000
## 1.696806 1.697569 1.693463 1.700042 1.692037 1.692357 1.697137 1.694051
##     1100     1200     1300     1400     1500
## 1.691786 1.692242 1.695878 1.696537 1.695764
```

```
par(mfrow = c(1,1))
plot(num_tree, mean.cv.errors, type = "b", xlab = "ntree", ylab = "CV RMSE", main = "5-Fold CV for Rando
```




```
best_ntree = num_tree[which.min(mean.cv.errors)]
best_ntree
```

```
## [1] 1100
```

```
best_mtry_ntree_model = randomForest(G3 ~ ., data = mostImportTrain, mtry = 4, ntree = best_ntree, impo
preds = predict(best_mtry_ntree_model, mostImportTest)
sqrt(mean((testData$G3 - preds)^2))
```

```
## [1] 1.526788
```