LM

Table 1: **Table 1. Overall Characteristics**

Characteristic	**N = 948**[1]
Gender	
female	488 / 948 (51%)
male	460 / 948 (49%)
EthnicGroup	
group A	80 / 948 (8.4%)
group B	171 / 948 (18%)
group C	336 / 948 (35%)
group D	237 / 948 (25%)
group E	124 / 948 (13%)
ParentEduc	
some high school	163 / 948 (17%)
high school	176 / 948 (19%)
associate's degree	198 / 948 (21%)
some college	252 / 948 (27%)
bachelor's degree	104 / 948 (11%)
master's degree	55 / 948 (5.8%)
LunchType	
free/reduced	331 / 948 (35%)
standard	617 / 948 (65%)
TestPrep	
completed	322 / 948 (34%)
none	626 / 948 (66%)
ParentMaritalStatus	
divorced	146 / 948 (15%)
married	565 / 948 (60%)
single	213 / 948 (22%)
widowed	24 / 948 (2.5%)
PracticeSport	

[1]n / N (%); Mean (SD)

Table 1: **Table 1. Overall Characteristics**

Characteristic	N = 948[1]
never	112 / 948 (12%)
sometimes	493 / 948 (52%)
regularly	343 / 948 (36%)
IsFirstChild	634 / 948 (67%)
NrSiblings	2 (1)
TransportMeans	
private	337 / 948 (36%)
school_bus	611 / 948 (64%)
WklyStudyHours	
< 5	253 / 948 (27%)
5-10	545 / 948 (57%)
> 10	150 / 948 (16%)
MathScore	66 (16)
ReadingScore	69 (15)
WritingScore	68 (15)

[1] n / N (%); Mean (SD)

`Math`

Break the data into training and testing

```
data_cleaned_math <-
  data_cleaned %>%
  select(c(-ReadingScore,-WritingScore))
train.indices <- sample(nrow(data_cleaned_math), floor(nrow(data_cleaned_math)/1.5), replace = FALSE)
validation.indices <- seq(nrow(data_cleaned_math))[-train.indices]
pred.data.train <- data_cleaned_math[train.indices,]
pred.data.train <- pred.data.train[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
pred.data.validation <- data_cleaned_math[validation.indices,]
pred.data.validation <- pred.data.validation[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
```

```
glmnet.formula <- as.formula(MathScore ~ .)
glmnet.design.matrix <- model.matrix(glmnet.formula, data = pred.data.train)
dim(glmnet.design.matrix)
```

```
## [1] 632  23
```

```
glmnet.cv.data.out <- cv.glmnet(glmnet.design.matrix,
                     y = pred.data.train$MathScore,
                     family = c("gaussian"),
                     type.measure="mse", # the model selection criteria
                     alpha = 1) # The Lasso regression
plot(glmnet.cv.data.out)
```

Table 2: Table 2. Math score backwards stepwise model

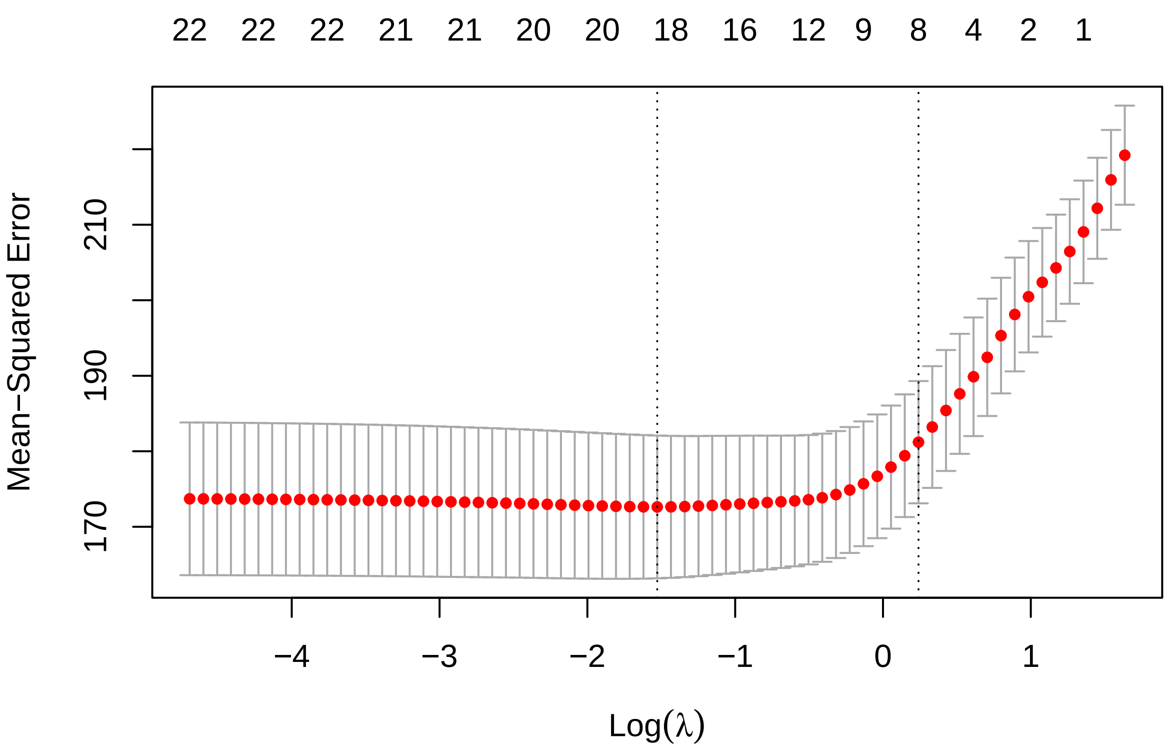
term	estimate	p.value
(Intercept)	55.976	<.001
Gendermale	4.981	<.001
EthnicGroup.L	7.401	<.001
EthnicGroup.Q	2.940	0.011
EthnicGroup.C	0.737	0.482
EthnicGroup^4	-0.985	0.251
ParentEduc.L	6.642	<.001
ParentEduc.Q	-0.045	0.972
ParentEduc.C	-0.300	0.806
ParentEduc^4	1.402	0.213
ParentEduc^5	-2.644	0.006
LunchTypestandard	11.155	<.001
TestPrepnone	-5.582	<.001
ParentMaritalStatusmarried	3.876	0.002
ParentMaritalStatussingle	1.102	0.444
ParentMaritalStatuswidowed	5.066	0.087
PracticeSport.L	2.514	0.015
PracticeSport.Q	-0.518	0.503
IsFirstChildyes	2.368	0.011
WklyStudyHours.L	2.621	0.008
WklyStudyHours.Q	-1.032	0.158

Table 3: Table 3. Reading score backwards stepwise model

term	estimate	p.value
(Intercept)	68.495	<.001
Gendermale	-7.282	<.001
EthnicGroup.L	4.149	0.001
EthnicGroup.Q	1.438	0.203
EthnicGroup.C	-0.471	0.645
EthnicGroup^4	-0.942	0.261
ParentEduc.L	7.638	<.001
ParentEduc.Q	1.535	0.22
ParentEduc.C	0.566	0.635
ParentEduc^4	1.547	0.158
ParentEduc^5	-3.033	0.001
LunchTypestandard	7.494	<.001
TestPrepnone	-6.972	<.001
ParentMaritalStatusmarried	4.113	0.001
ParentMaritalStatussingle	1.275	0.363
ParentMaritalStatuswidowed	4.645	0.106
IsFirstChildyes	2.446	0.007
WklyStudyHours.L	1.431	0.135
WklyStudyHours.Q	-0.933	0.191

Table 4: Table 4. Writing score backwards stepwise model

term	estimate	p.value
(Intercept)	69.347	<.001
Gendermale	-9.209	<.001
EthnicGroup.L	4.673	<.001
EthnicGroup.Q	0.627	0.569
EthnicGroup.C	-1.891	0.058
EthnicGroup^4	-1.649	0.043
ParentEduc.L	9.983	<.001
ParentEduc.Q	1.365	0.265
ParentEduc.C	0.289	0.803
ParentEduc^4	1.715	0.109
ParentEduc^5	-3.106	0.001
LunchTypestandard	8.388	<.001
TestPrepnone	-9.629	<.001
ParentMaritalStatusmarried	4.135	0.001
ParentMaritalStatussingle	1.056	0.44
ParentMaritalStatuswidowed	3.950	0.16
PracticeSport.L	2.251	0.022
PracticeSport.Q	-0.708	0.335
IsFirstChildyes	2.208	0.012
WklyStudyHours.L	1.338	0.152
WklyStudyHours.Q	-0.960	0.167




```
saved.coef <- coef(glmnet.cv.data.out, s=c("lambda.1se"))
chosen.vars <- data.frame(name = saved.coef@Dimnames[[1]][saved.coef@i + 1],
                          coefficient = saved.coef@x)
```

```r
print(paste("The lasso regression chose", dim(chosen.vars)[1]-1,
            "variables and 1 intercept"))
```

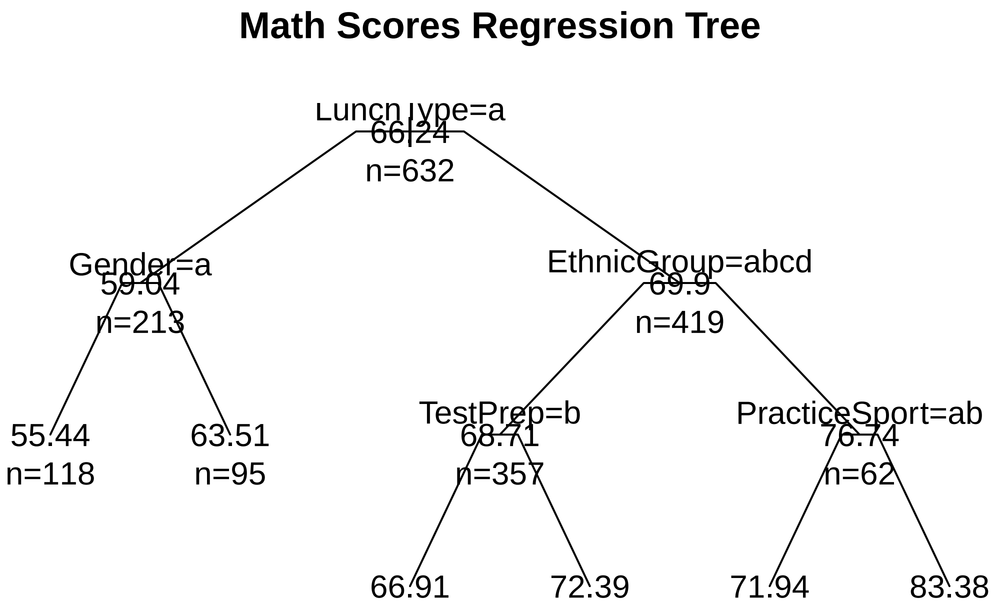
```
## [1] "The lasso regression chose 8 variables and 1 intercept"
```

```r
print(saved.coef)
```

```
## 24 x 1 sparse Matrix of class "dgCMatrix"
##                                      s1
## (Intercept)                  61.4240942
## (Intercept)                   .
## Gendermale                    1.9051110
## EthnicGroup.L                 4.0493431
## EthnicGroup.Q                 0.3564610
## EthnicGroup.C                 .
## EthnicGroup^4                 .
## ParentEduc.L                  1.6718964
## ParentEduc.Q                  .
## ParentEduc.C                  .
## ParentEduc^4                  .
## ParentEduc^5                  .
## LunchTypestandard             8.0257026
## TestPrepnone                 -2.6007855
## ParentMaritalStatusmarried    0.5752321
## ParentMaritalStatussingle     .
## ParentMaritalStatuswidowed    .
## PracticeSport.L               .
## PracticeSport.Q               .
## IsFirstChildyes               .
## NrSiblings                    .
## TransportMeansschool_bus      .
## WklyStudyHours.L              0.6616217
## WklyStudyHours.Q              .
```

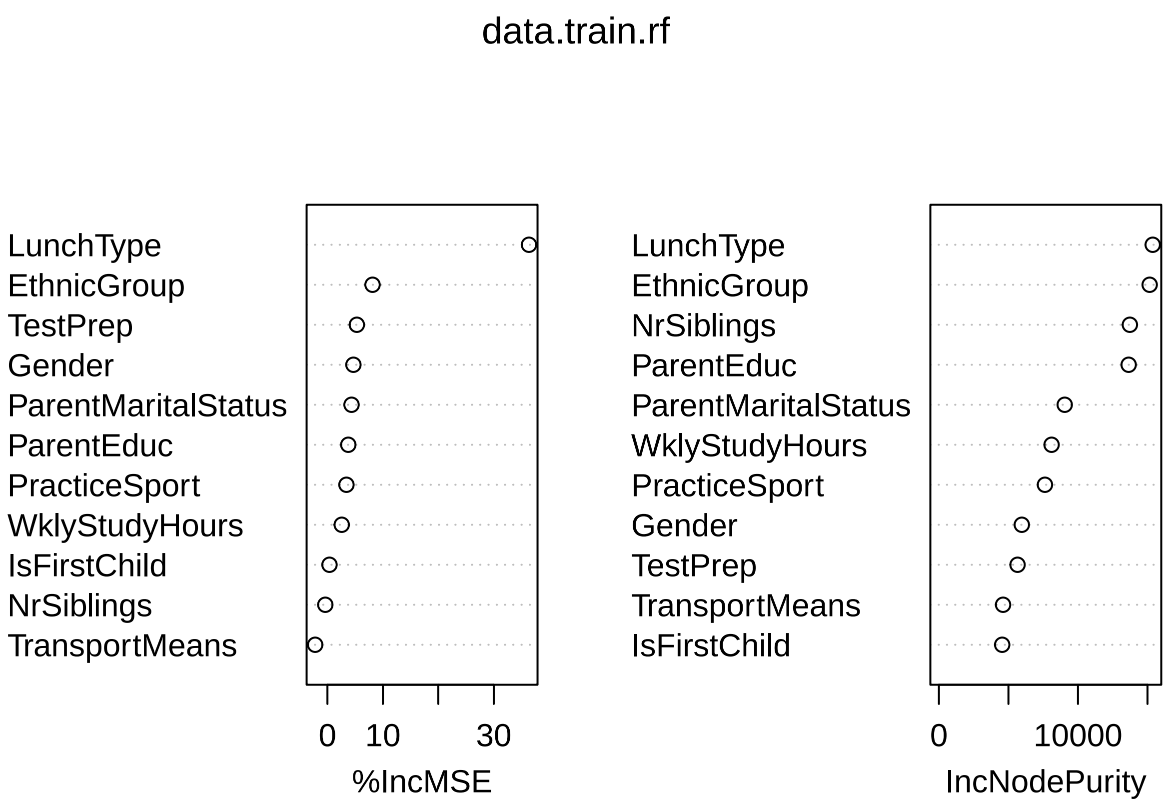
Math Scores Regression Tree

```r
tree.out.1 <-rpart(MathScore ~ ., data = pred.data.train,
                   parms   = list(split="information"),
                   control = rpart.control(minsplit=20))
#Create a plot of the classification tree.
#Code to plot the tree.
plot(tree.out.1, uniform=TRUE, branch=0.2, margin=0.02)
text(tree.out.1, all=TRUE, use.n=TRUE)
title("Math Scores Regression Tree")
```



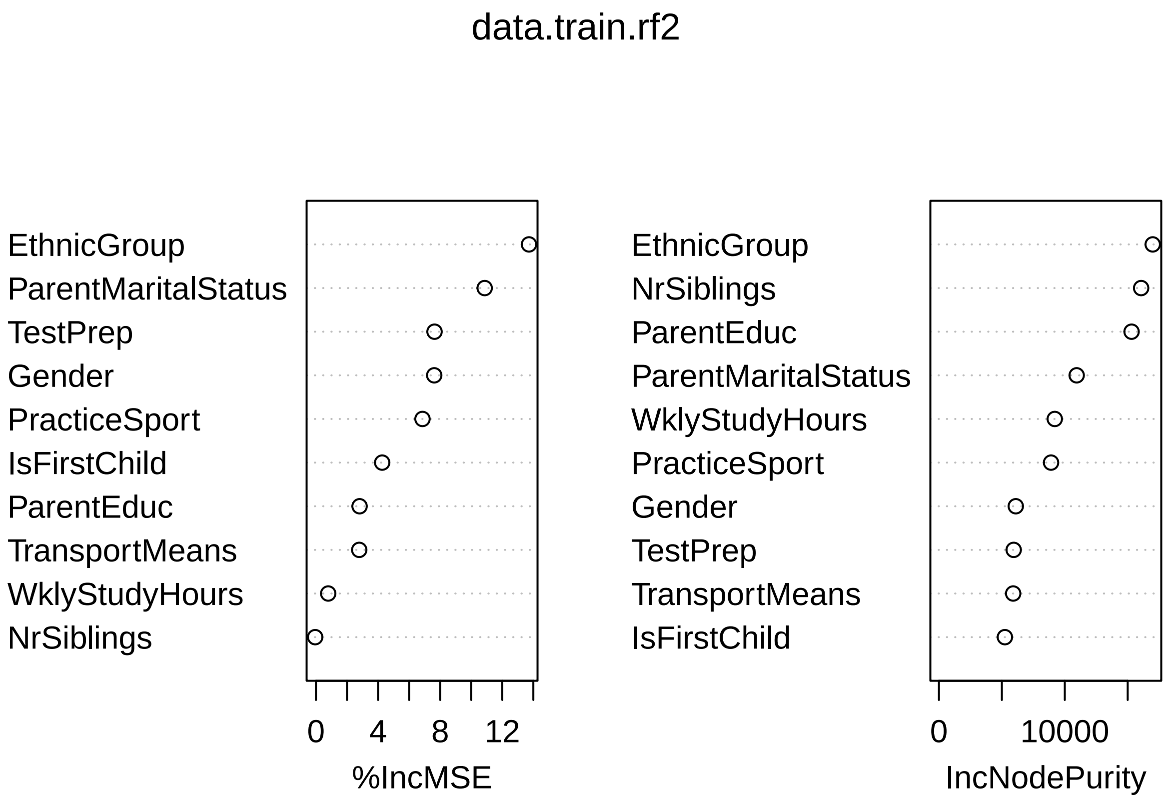
Use Random Forest to see if we can model it better

```
data.train.rf <- randomForest(MathScore ~ .,
                              data = pred.data.train,
                              importance=TRUE)

varImpPlot(data.train.rf)
```



we see lunch type is skewing the data so we need to get rid of it and train the random forest again.

```r
data.train.rf2 <- randomForest(MathScore ~ . -LunchType,
                          data = pred.data.train,
                          importance=TRUE)

varImpPlot(data.train.rf2)
```



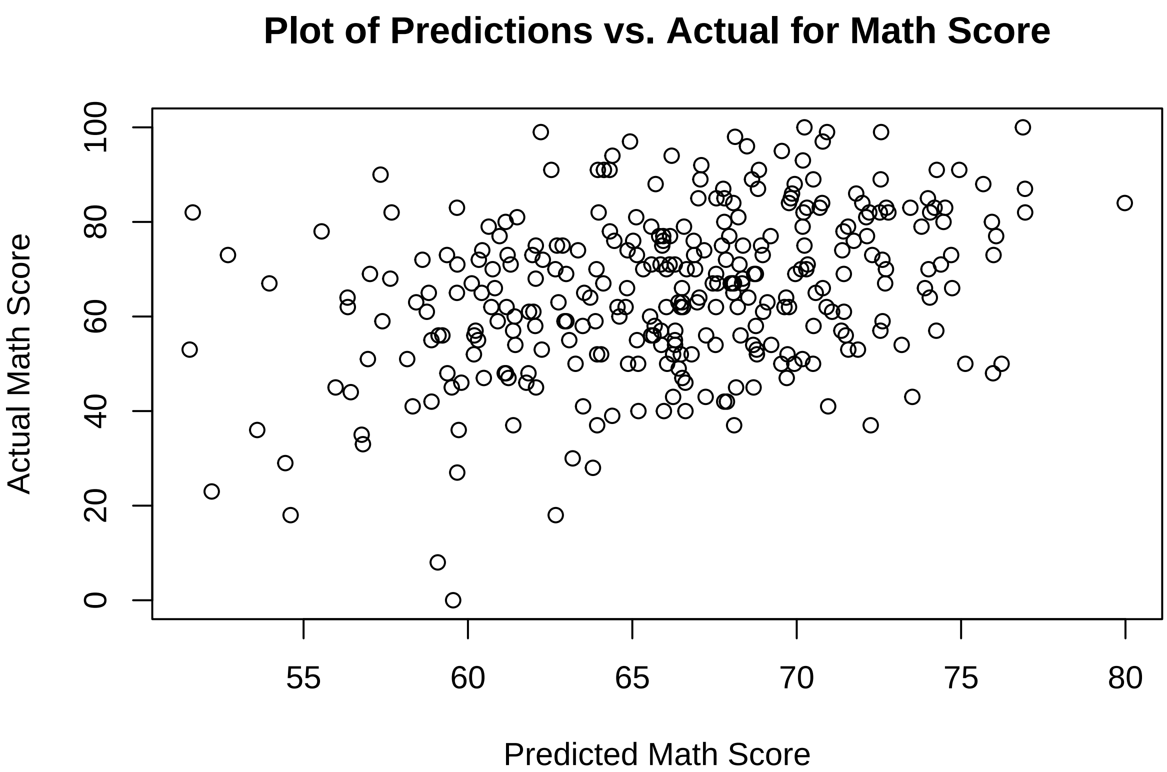

```r
optimum <- which.max(data.train.rf2$importance[,"%IncMSE"])
opt.var <- data.train.rf2$importance[optimum,0,drop=FALSE]
print("The most predictive variable with regard to Math Score is:")
```

```
## [1] "The most predictive variable with regard to Math Score is:"
```

```r
print(opt.var)
```

```
##
## EthnicGroup
```

```r
val.preds.rf <- predict(data.train.rf2, # The forest
newdata = pred.data.validation, # The values of x to do prediction at type = c("response")
)
# Code to plot the predictions against the actual values
plot(val.preds.rf, pred.data.validation$MathScore,
     main = "Plot of Predictions vs. Actual for Math Score",
     xlab = "Predicted Math Score",
     ylab = "Actual Math Score")
```



‘

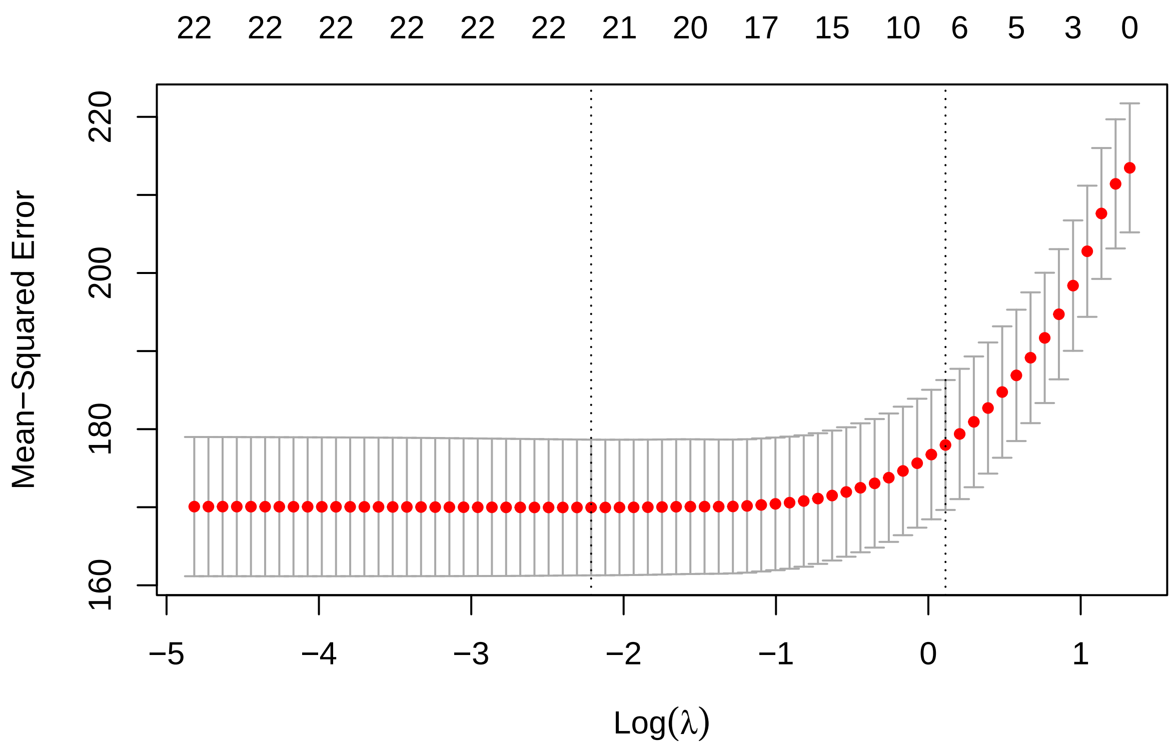
Reading scores

```
data_cleaned_read <-
  data_cleaned %>%
  select(c(-MathScore,-WritingScore))
train.indices <- sample(nrow(data_cleaned_read), floor(nrow(data_cleaned_read)/1.5), replace = FALSE)
validation.indices <- seq(nrow(data_cleaned_read))[-train.indices]
pred.data.train <- data_cleaned_read[train.indices,]
pred.data.train <- pred.data.train[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
pred.data.validation <- data_cleaned_read[validation.indices,]
pred.data.validation <- pred.data.validation[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
```

```
glmnet.formula <- as.formula(ReadingScore ~ .)
glmnet.design.matrix <- model.matrix(glmnet.formula, data = pred.data.train)
dim(glmnet.design.matrix)
```

```
## [1] 632  23
```

```
glmnet.cv.data.out <- cv.glmnet(glmnet.design.matrix,
                     y = pred.data.train$ReadingScore,
                     family = c("gaussian"),
                     type.measure="mse", # the model selection criteria
                     alpha = 1) # The Lasso regression
plot(glmnet.cv.data.out)
```




```r
saved.coef <- coef(glmnet.cv.data.out, s=c("lambda.1se"))
chosen.vars <- data.frame(name = saved.coef@Dimnames[[1]][saved.coef@i + 1],
                          coefficient = saved.coef@x)
print(paste("The lasso regression chose", dim(chosen.vars)[1]-1,
            "variables and 1 intercept"))
```

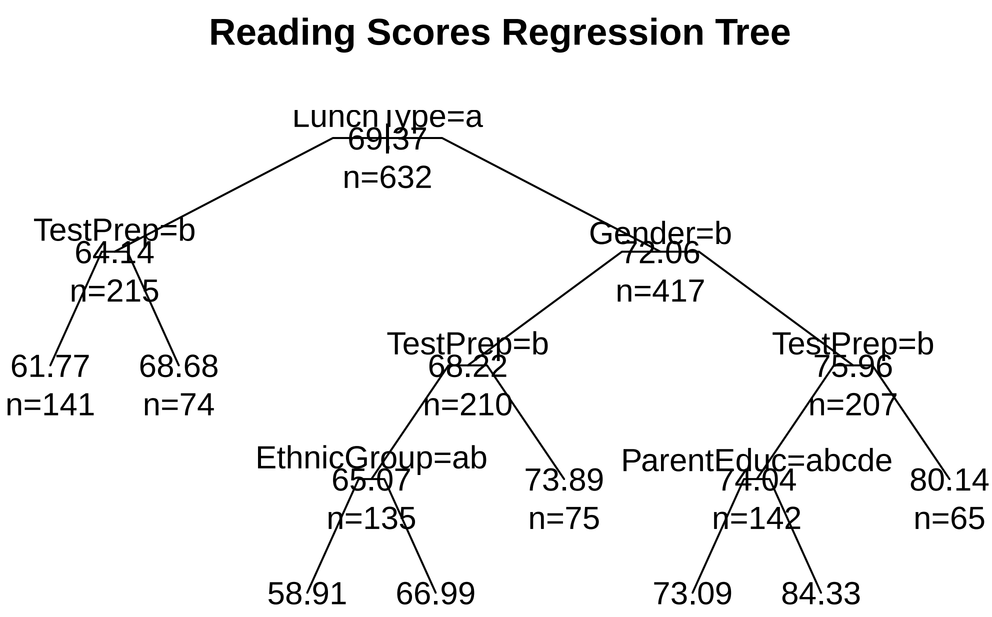
```
## [1] "The lasso regression chose 9 variables and 1 intercept"
```

```r
print(saved.coef)
```

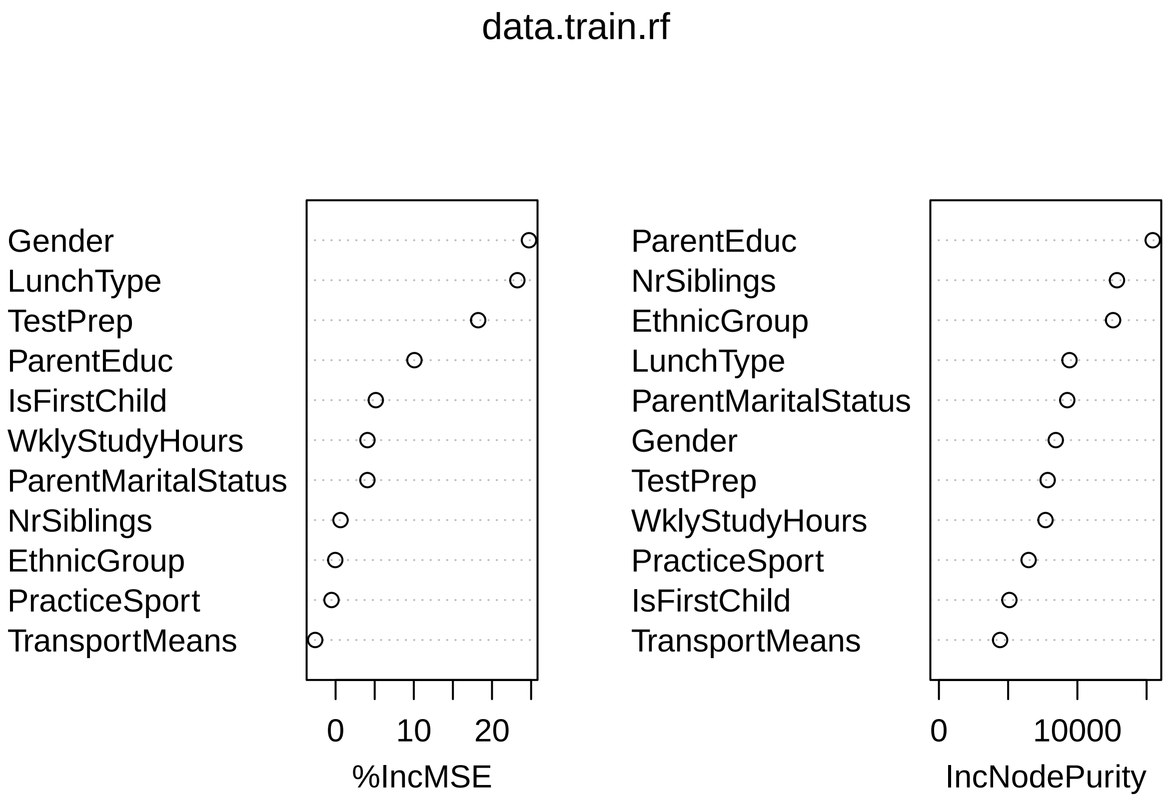
```
## 24 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)                 70.9918283
## (Intercept)                  .
## Gendermale                  -4.6349752
## EthnicGroup.L                1.6778520
## EthnicGroup.Q                .
## EthnicGroup.C                .
## EthnicGroup^4                .
## ParentEduc.L                 2.3182319
## ParentEduc.Q                 .
## ParentEduc.C                 .
## ParentEduc^4                 0.0339427
## ParentEduc^5                -0.9007165
## LunchTypestandard            5.6762097
## TestPrepnone                -4.8034515
## ParentMaritalStatusmarried   0.1754840
## ParentMaritalStatussingle    .
## ParentMaritalStatuswidowed   .
## PracticeSport.L              .
## PracticeSport.Q              .
```

```
## IsFirstChildyes              0.1408015
## NrSiblings                   .
## TransportMeansschool_bus     .
## WklyStudyHours.L             .
## WklyStudyHours.Q             .
```

```r
tree.out.1 <-rpart(ReadingScore ~ ., data = pred.data.train,
                   parms  = list(split="information"),
                   control = rpart.control(minsplit=20))
#Create a plot of the classification tree.
#Code to plot the tree.
plot(tree.out.1, uniform=TRUE, branch=0.2, margin=0.02)
text(tree.out.1, all=TRUE, use.n=TRUE)
title("Reading Scores Regression Tree")
```




```r
data.train.rf <- randomForest(ReadingScore ~ .,
                              data = pred.data.train,
                              importance=TRUE)

varImpPlot(data.train.rf)
```




```r
data.train.rf2 <- randomForest(ReadingScore ~ . -LunchType,
                          data = pred.data.train,
                          importance=TRUE)

varImpPlot(data.train.rf2)
```

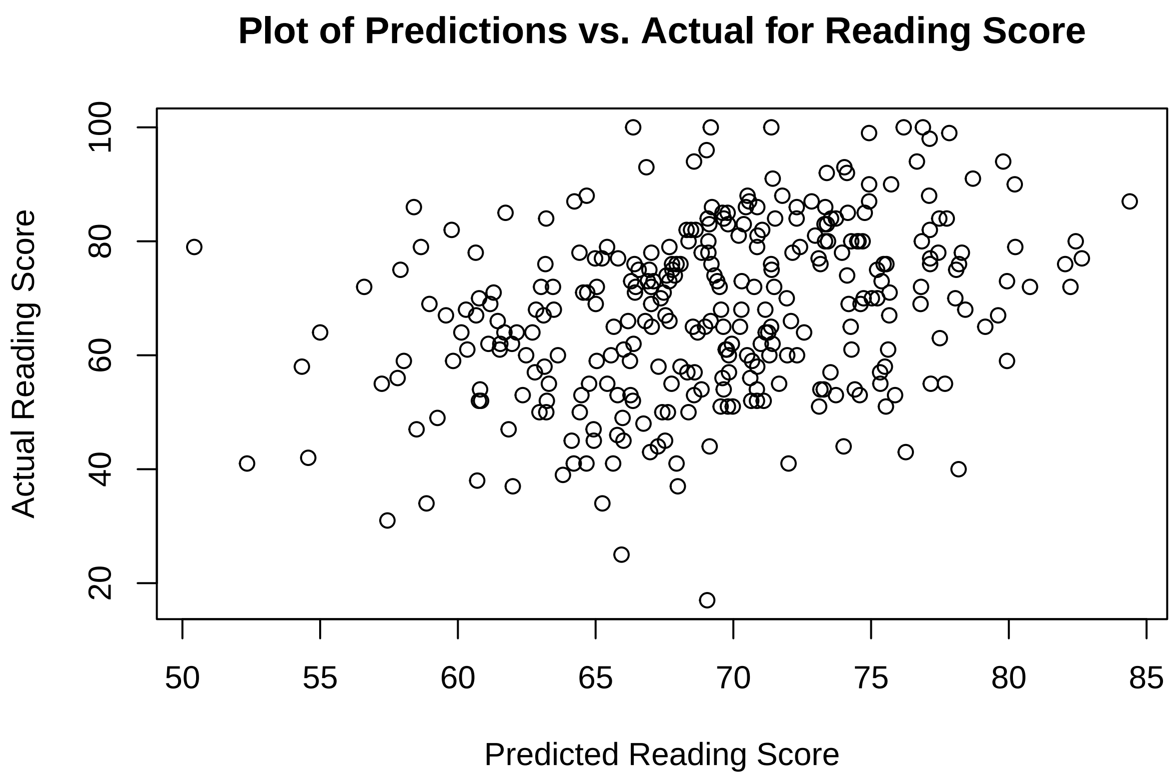

```
optimum <- which.max(data.train.rf2$importance[,"%IncMSE"])
opt.var <- data.train.rf2$importance[optimum,0,drop=FALSE]
print("The most predictive variable with regard to Reading Score is:")
```

```
## [1] "The most predictive variable with regard to Reading Score is:"
```

```
print(opt.var)
```

```
##
## Gender
```

```
val.preds.rf <- predict(data.train.rf2, # The forest
newdata = pred.data.validation, # The values of x to do prediction at type = c("response")
)
# Code to plot the predictions against the actual values
plot(val.preds.rf, pred.data.validation$ReadingScore,
     main = "Plot of Predictions vs. Actual for Reading Score",
     xlab = "Predicted Reading Score",
     ylab = "Actual Reading Score")
```



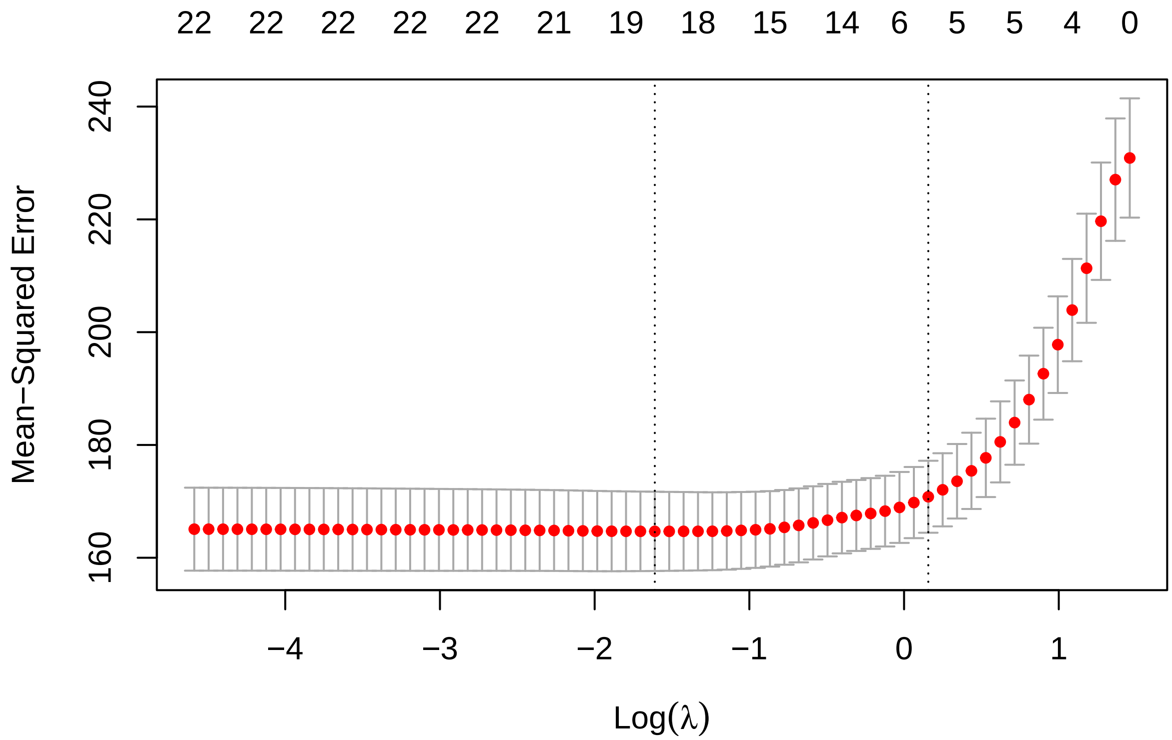
Writing Scores

```
data_cleaned_writing <-
  data_cleaned %>%
  select(c(-MathScore,-ReadingScore))
train.indices <- sample(nrow(data_cleaned_writing), floor(nrow(data_cleaned_writing)/1.5), replace = FA
validation.indices <- seq(nrow(data_cleaned_writing))[-train.indices]
pred.data.train <- data_cleaned_writing[train.indices,]
pred.data.train <- pred.data.train[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
pred.data.validation <- data_cleaned_writing[validation.indices,]
pred.data.validation <- pred.data.validation[,c(1,2,3,4,5,6,7,8,9,10,11,12)]
```

```
glmnet.formula <- as.formula(WritingScore ~ .)
glmnet.design.matrix <- model.matrix(glmnet.formula, data = pred.data.train)
dim(glmnet.design.matrix)
```

```
## [1] 632  23
```

```
glmnet.cv.data.out <- cv.glmnet(glmnet.design.matrix,
                      y = pred.data.train$WritingScore,
                      family = c("gaussian"),
                      type.measure="mse", # the model selection criteria
                      alpha = 1) # The Lasso regression
plot(glmnet.cv.data.out)
```




```
saved.coef <- coef(glmnet.cv.data.out, s=c("lambda.1se"))
chosen.vars <- data.frame(name = saved.coef@Dimnames[[1]][saved.coef@i + 1],
                         coefficient = saved.coef@x)
print(paste("The lasso regression chose", dim(chosen.vars)[1]-1,
            "variables and 1 intercept"))
```

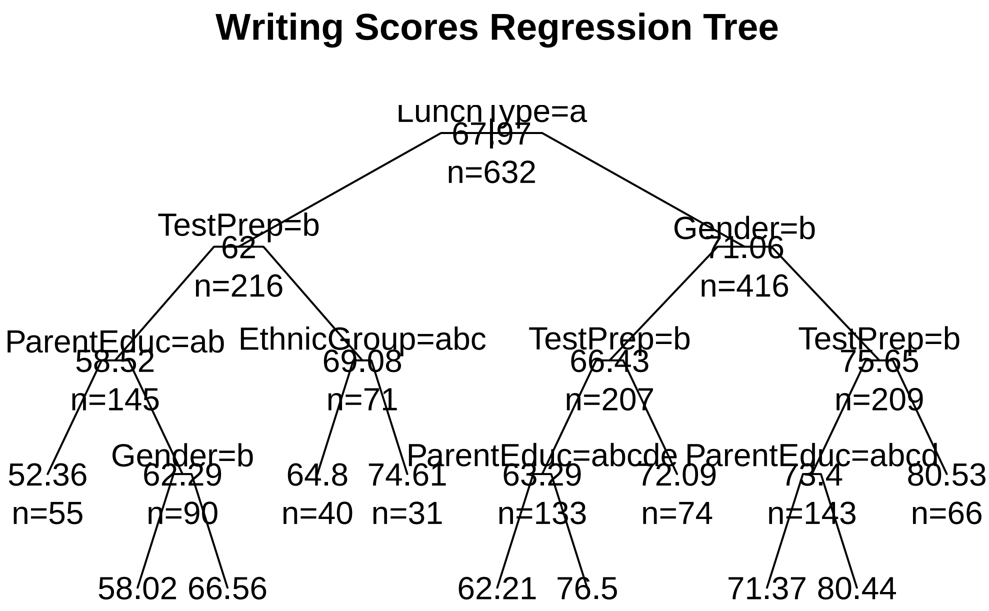
```
## [1] "The lasso regression chose 6 variables and 1 intercept"
```

```
print(saved.coef)
```

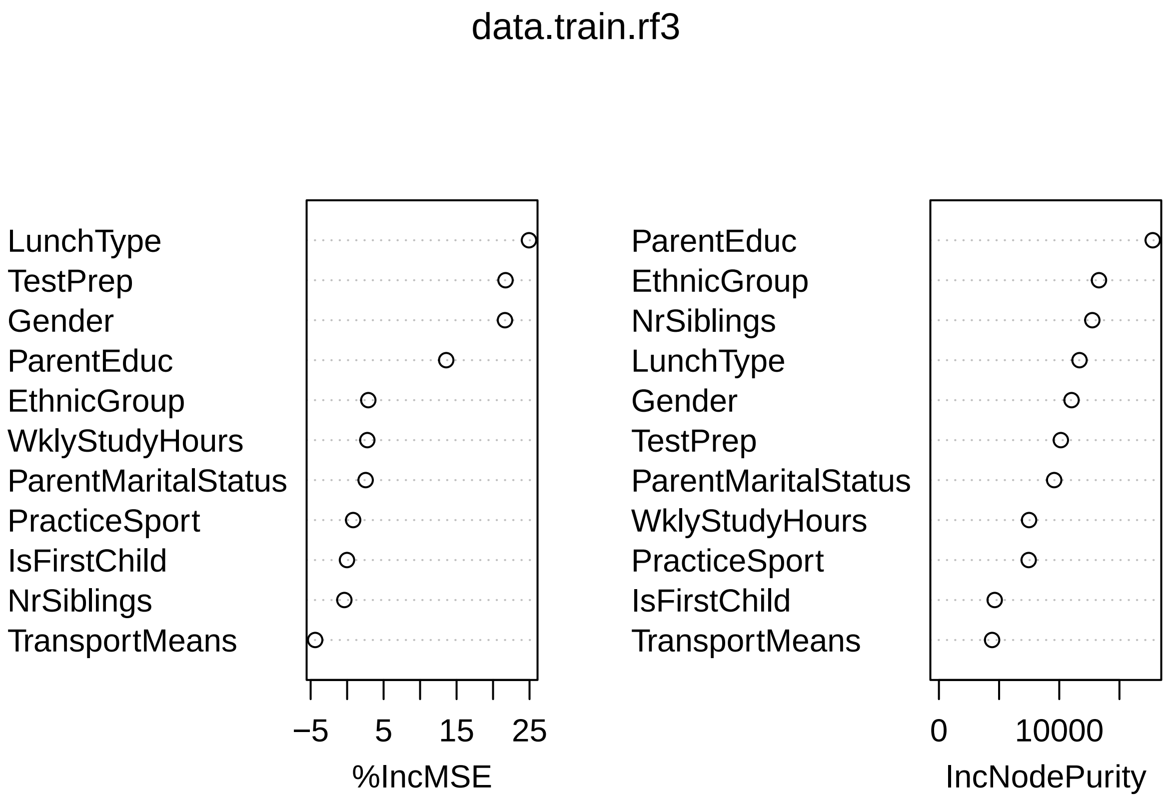
```
## 24 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)                 71.1170761
## (Intercept)                  .
## Gendermale                  -5.9360919
## EthnicGroup.L                3.0335608
## EthnicGroup.Q                .
## EthnicGroup.C                .
## EthnicGroup^4                .
## ParentEduc.L                 6.6879270
## ParentEduc.Q                 .
## ParentEduc.C                 .
## ParentEduc^4                 .
## ParentEduc^5                -0.1873087
## LunchTypestandard            6.5443619
## TestPrepnone                -6.2404584
## ParentMaritalStatusmarried   .
## ParentMaritalStatussingle    .
## ParentMaritalStatuswidowed   .
## PracticeSport.L              .
## PracticeSport.Q              .
```

```
## IsFirstChildyes                .
## NrSiblings                     .
## TransportMeansschool_bus       .
## WklyStudyHours.L               .
## WklyStudyHours.Q               .
```

```r
tree.out.1 <-rpart(WritingScore ~ ., data = pred.data.train,
                   parms  = list(split="information"),
                   control = rpart.control(minsplit=20))
#Create a plot of the classification tree.
#Code to plot the tree.
plot(tree.out.1, uniform=TRUE, branch=0.2, margin=0.02)
text(tree.out.1, all=TRUE, use.n=TRUE)
title("Writing Scores Regression Tree")
```




```r
data.train.rf3 <- randomForest(WritingScore ~ .,
                          data = pred.data.train,
                          importance=TRUE)

varImpPlot(data.train.rf3)
```



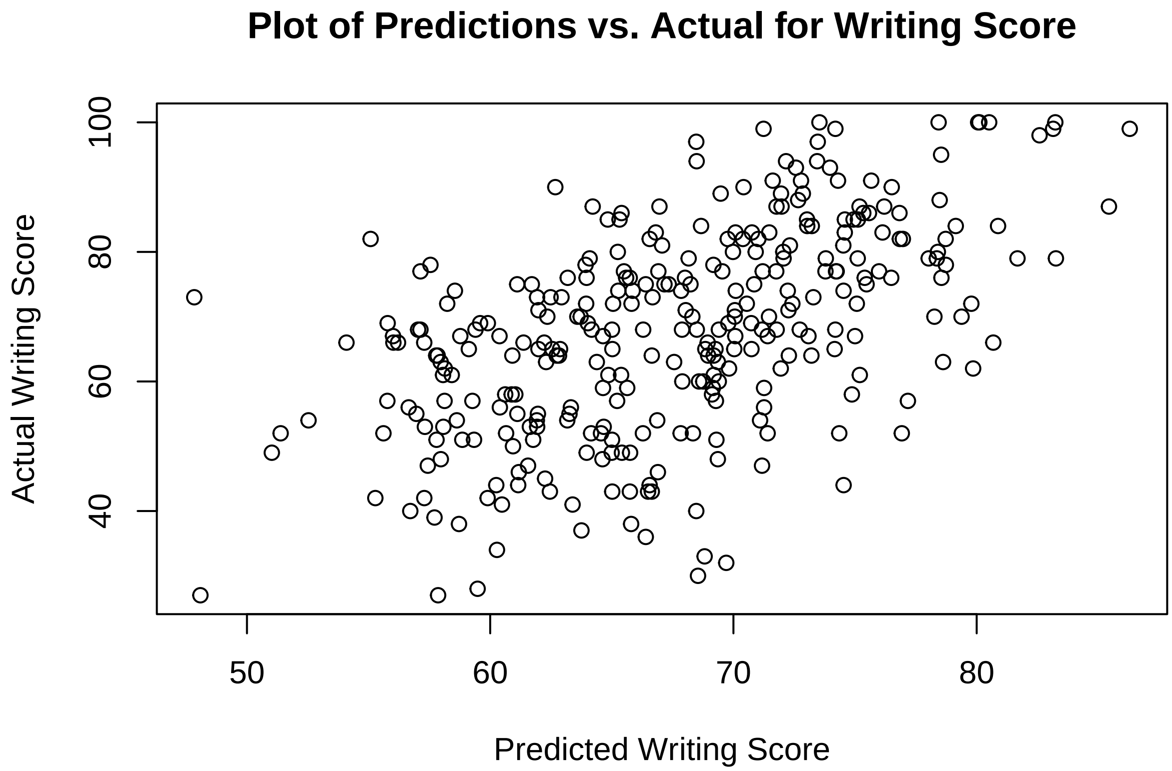

```
optimum <- which.max(data.train.rf3$importance[,"%IncMSE"])
opt.var <- data.train.rf3$importance[optimum,0,drop=FALSE]
print("The most predictive variable with regard to Writing Score is:")
```

```
## [1] "The most predictive variable with regard to Writing Score is:"
```

```
print(opt.var)
```

```
##
## LunchType
```

```
val.preds.rf <- predict(data.train.rf3, # The forest
newdata = pred.data.validation, # The values of x to do prediction at type = c("response")
)
# Code to plot the predictions against the actual values
plot(val.preds.rf, pred.data.validation$WritingScore,
     main = "Plot of Predictions vs. Actual for Writing Score",
     xlab = "Predicted Writing Score",
     ylab = "Actual Writing Score")
```



We can conclude that the most important variables runnning the random forest model for prediction was the Test Prep category to sort the means the of the test scores.