Tree/Random Forest Lab

Nathan Sandford, Mike Chaykowsky, and Chris Donnay

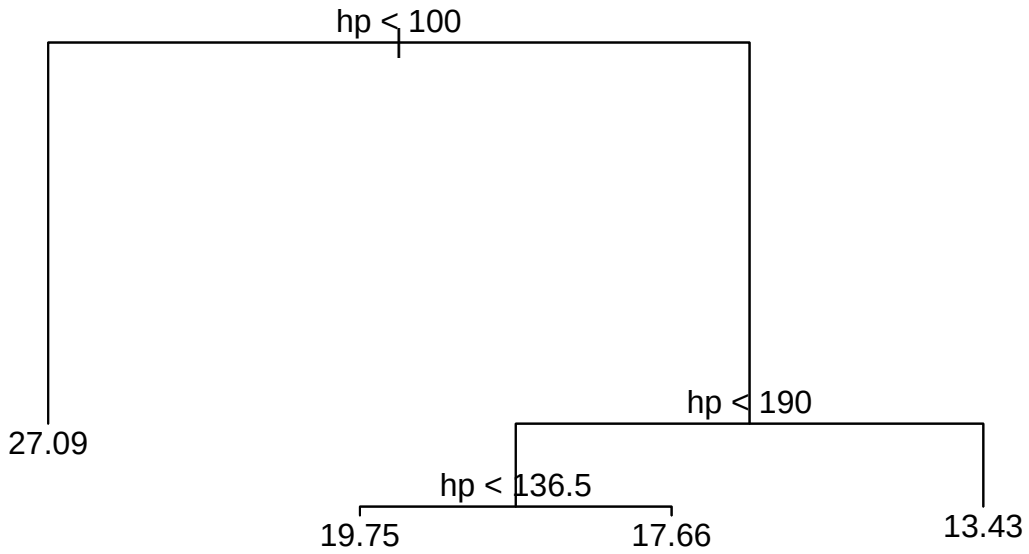
```
# Required Libraries
library(tree)
library(randomForest)
library(hydroGOF) # for easy calculation of MSE of model predictions on test data
```

We begin our exploration of regression trees and random forests by splitting our data set, mtcars, into a training data set (24 observations) and a test data set (8 observations). We wish to predict the mpg of each car given its other characteristics.

```
# Read in Data
data(mtcars)
attach(mtcars)

N_OBS <- length(mtcars[,1]) # Number of observations (32)

index.train <- sample(1:N_OBS,N_OBS*0.75) # Randomly sample of 24 observations for training data
mpg.train <- mtcars[index.train,]

mpg.test <-mtcars[-index.train,] # Remaining observations make up test data
```

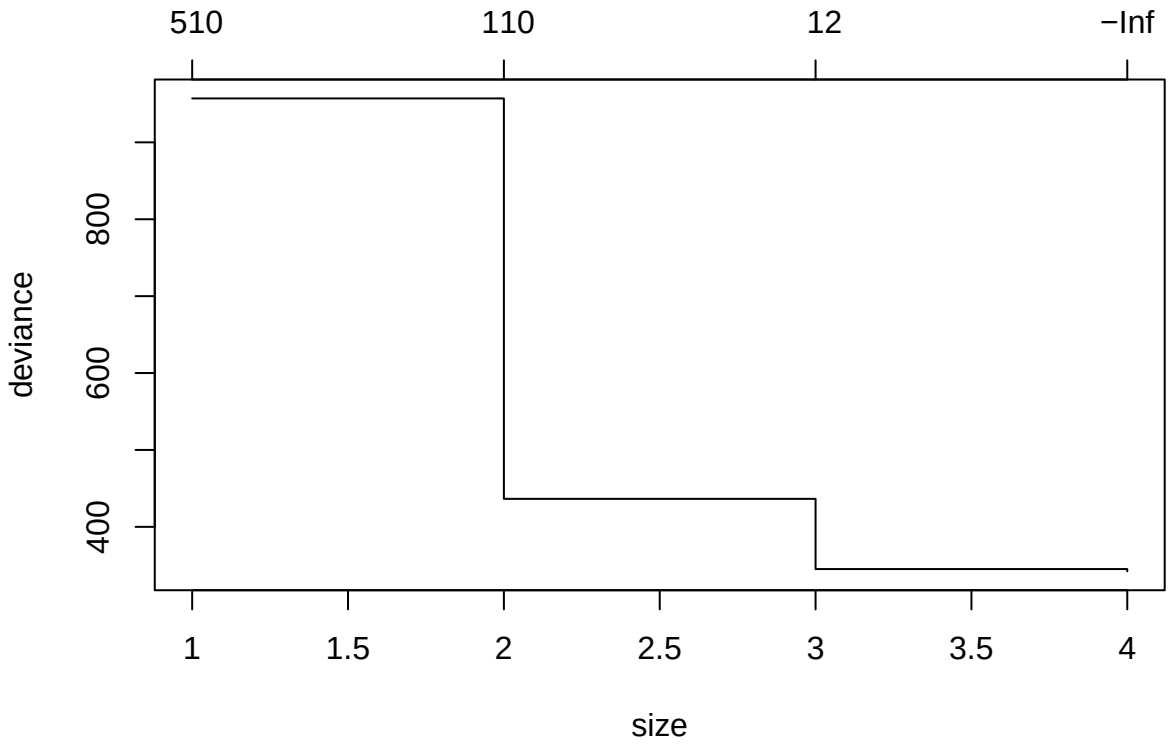
Performing a regression of the mpg with respect to the other variables using a regression tree (providing no constraints on the number of terminal nodes or partitions), we end up with the following tree:

```
mpg.tree <- tree(mpg ~ ., data=mpg.train) # Train regression tree

# Plots regression Tree
plot(mpg.tree)
text(mpg.tree)
```



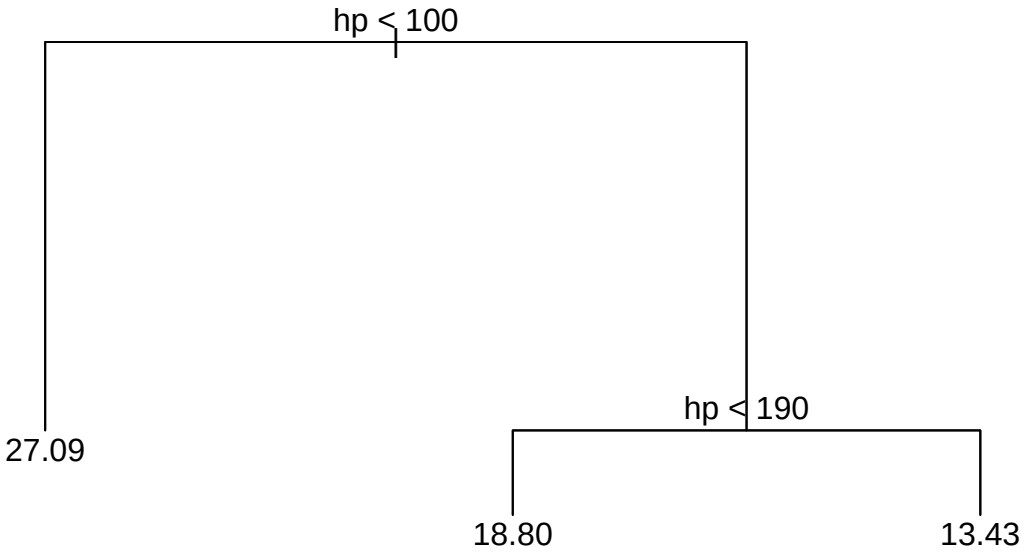
One of the drawbacks of regression trees is their ability to overfit the training data. As a check, we perform cross-validation of our tree, pruning back the least important partition all the way to the stem and plot the deviance of the regression tree from the training data.

```r
mpg.cv.tree <- cv.tree(mpg.tree) # Cross-Validation of pruning back regression tree

# Plot deviance vs. size
plot(mpg.cv.tree)
```



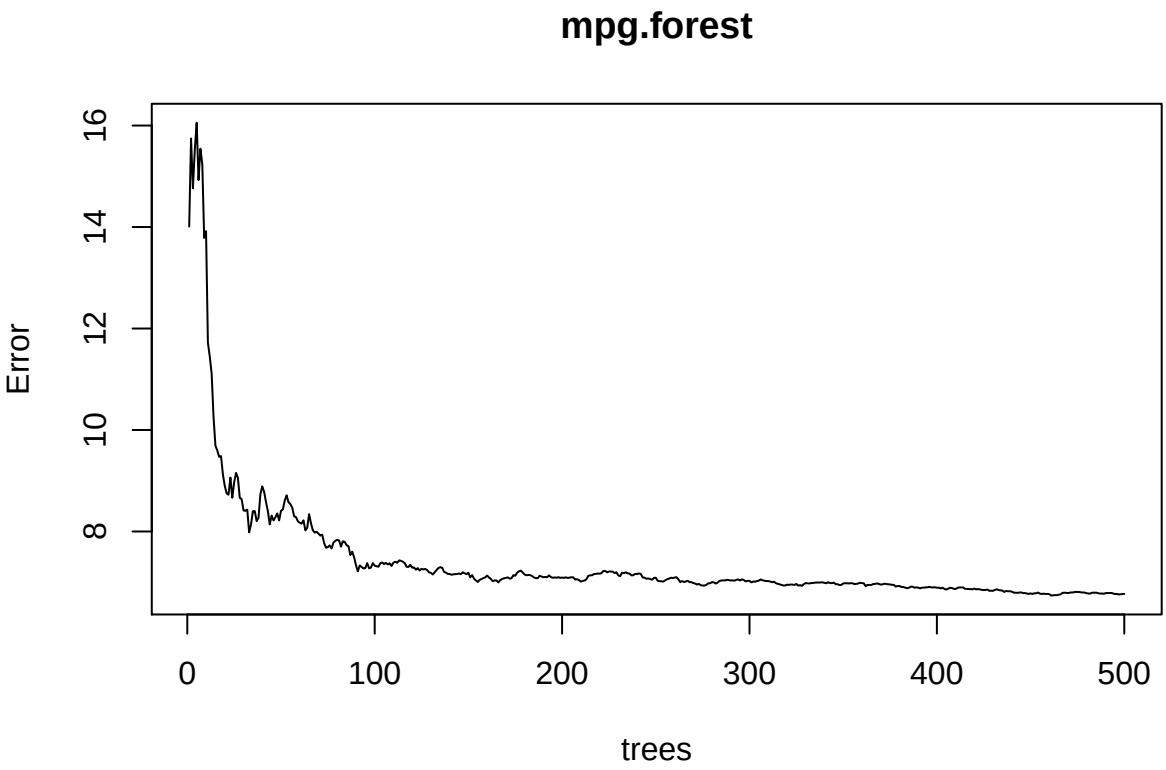
The decision of how far back to prune the model is a slightly subjective one. We want to reduce the deviance as much as possible with as few nodes as possible. There appears to be a consistent drop off in improvement of the deviation above 3 terminal nodes. using the prune.tree() function, we prune back our original regression tree model back to include only the 3 most important nodes.

```r
size <- 3 # "Optimum" size of regression tree

mpg.prune.tree <- prune.tree(mpg.tree, best=size) # Prunes original tree back

# Plots pruned regression tree
plot(mpg.prune.tree)
text(mpg.prune.tree)
```



But we can do more than just a regression tree—we can do 500 regression trees. Using the function randomForests, we can perform the same regression on the data set using 500 regression trees with varying partitions. These 500 models then vote on the best regression. In the plot below, we can see how the deviance of the model becomes better as we use more trees. Clearly we could have gotten away with fewer trees in the future.

```
mpg.forest <- randomForest(mpg ~ ., data=mpg.train) # Train random forest

# Plots error vs. number of trees
plot(mpg.forest)
```



Comparing each model's predictions of the mpg of cars in the test data set with their actual values:

```r
# Standard Regression Tree
mpg.pred <- predict(mpg.tree, newdata = mpg.test)
# Pruned Regression Tree
mpg.prune.pred <- predict(mpg.prune.tree, newdata = mpg.test)
# Random Forest
mpg.forest.pred <- predict(mpg.forest, newdata = mpg.test)


matrix(c(mpg.test$mpg, mpg.pred, mpg.prune.pred, mpg.forest.pred),nrow=4,byrow=TRUE)
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 16.40000 17.30000 15.20000 30.40000 21.50000 13.30000 30.40000
## [2,] 17.66000 17.66000 17.66000 27.08571 27.08571 13.43333 19.75000
## [3,] 18.80000 18.80000 18.80000 27.08571 27.08571 13.43333 18.80000
## [4,] 17.09933 17.01405 16.68655 29.21516 24.12963 15.31308 24.87142
##          [,8]
## [1,] 21.40000
## [2,] 19.75000
## [3,] 18.80000
## [4,] 23.73954
```

```r
# Row 1: True mpg of test data
# Row 2: Standard Regression Tree
# Row 3: Pruned Regression Tree
# Row 4: Random Forest
```

To evaluate each of the models, we calculated the MSE of the cars' predicted mpg to their true mpg. It is hard to speak quantitatively about the results here because every time the markdown is compiled the model is retrained on a slightly different training set. However, over the number of trials the random forest performed significantly better than the regression trees, while the pruned regression tree oftemn actually performed worse than the default regression tree. This is probably because the original tree was not overfitting and so reducing the size of the tree simply removed an important node of the tree. Occasionally, we do see an improvement in the pruned tree, indicating that the model did in fact overfit the trianing data.

```r
#Standard Regression Tree
mse(mpg.pred,mpg.test$mpg)
```

```
## [1] 20.76453
```

```r
# Pruned Regression Tree
mse(mpg.prune.pred,mpg.test$mpg)
```

```
## [1] 25.56156
```

```r
# Random Forest
mse(mpg.forest.pred,mpg.test$mpg)
```

```
## [1] 6.398821
```

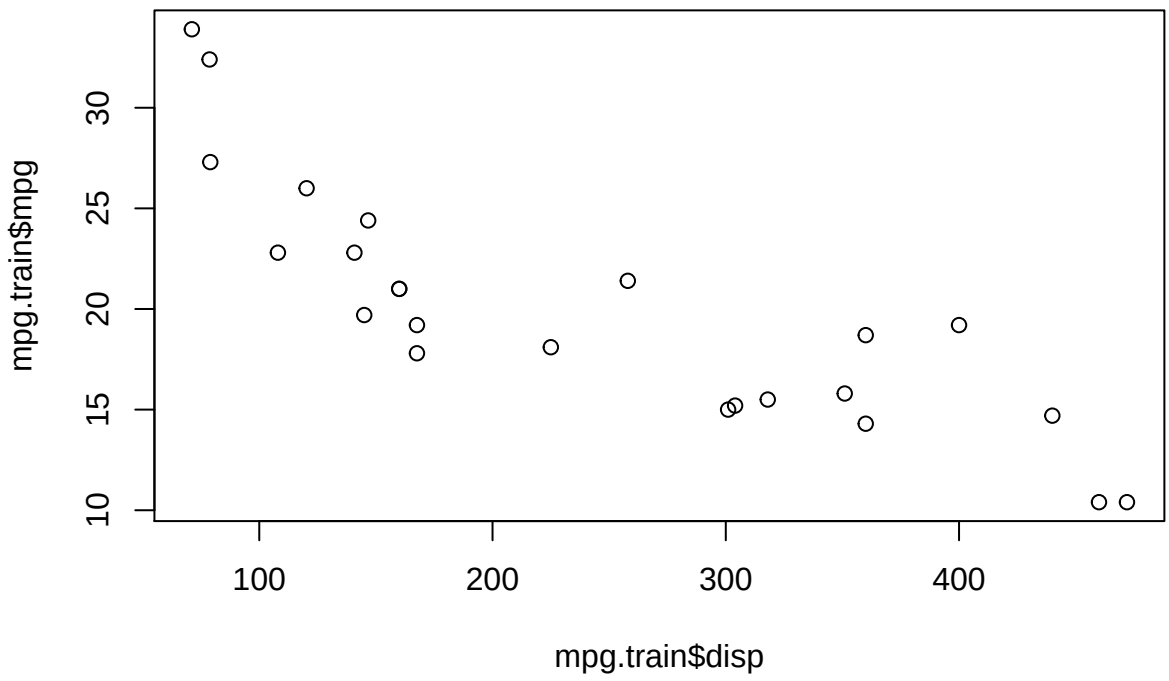
Additionally, we can take a look at what the important partitions of the trees are and how they compare with the relation between those values and the true mpg of the cars in the data set. These important partitions can be seen in the plots above for the regression trees and here for the random forest:

```
mpg.forest$importance
```

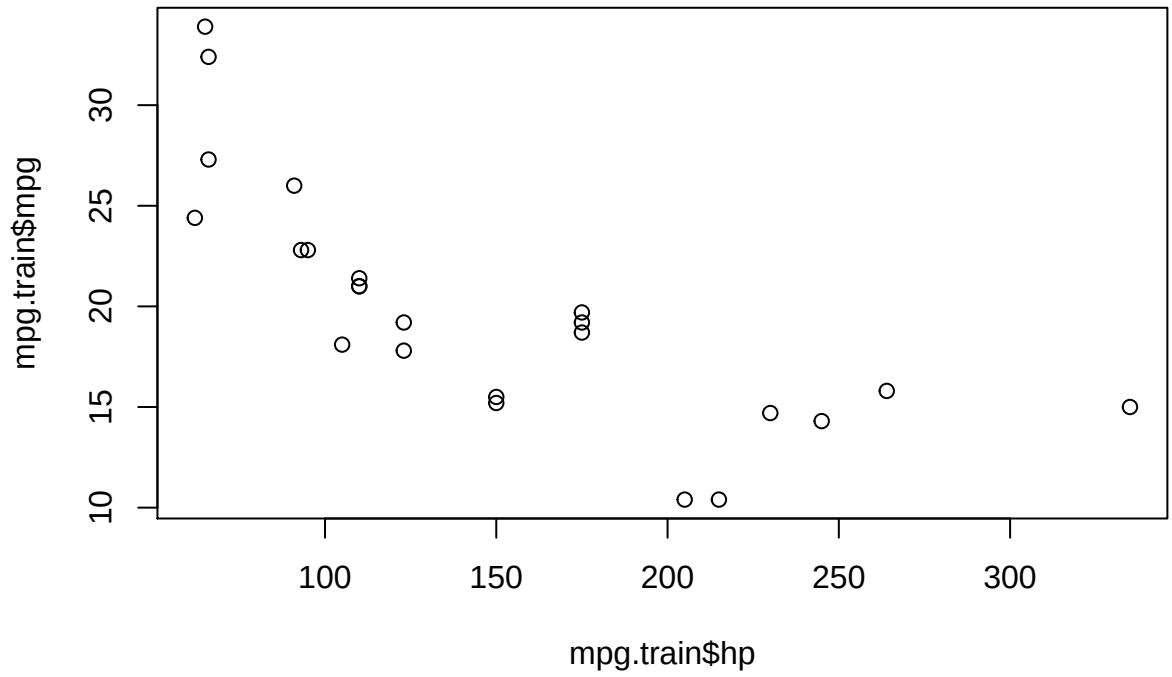
```
##      IncNodePurity
## cyl     127.823864
## disp    152.656123
## hp      172.774935
## drat     58.868144
## wt      139.216889
## qsec     40.866901
## vs        9.683635
## am        6.235157
## gear      7.860286
## carb     22.531370
```

Commonly the most important partitions are those in Displacement, Horse Power, the Number of Cylinders, and weight. As we would expect from the important partitions, all of these variables have a clear (if not linear) relationship with the mpg of the cars.

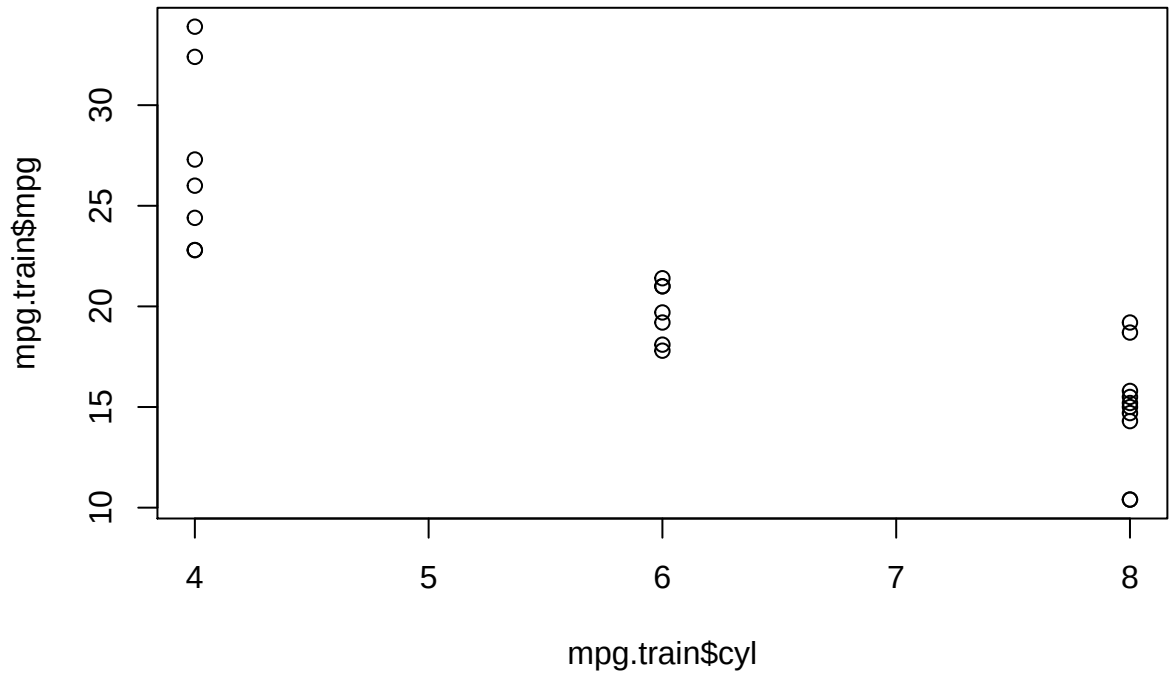
```
plot(mpg.train$disp,mpg.train$mpg)
```



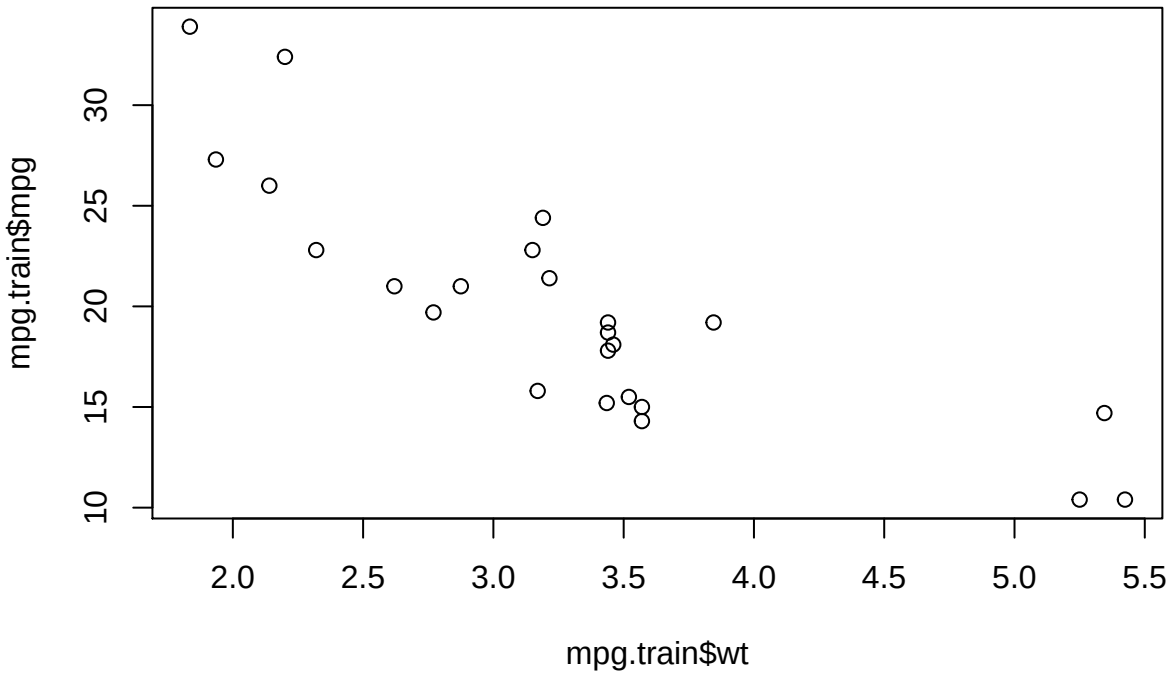

```
plot(mpg.train$hp,mpg.train$mpg)
```




```
plot(mpg.train$cyl,mpg.train$mpg)
```




```
plot(mpg.train$wt,mpg.train$mpg)
```


mpg.train$mpg
10
15
20
25
30
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
mpg.train$wt