BUAN6356_Homework4_RTolawat_GShinde_WArey_SBhaygude_RBhatt

Rohit Tolawat, Gauri Shinde, William Arey, Shivraj Bhaygude, Rahul Bhatt

4/4/2020

Loading all the relevant packages

```
pacman::p_load(ISLR, gplots, ggplot2, leaps, caret, tree, rpart, rpart.plot, caret, gbm,
               randomForest, ca, data.table)
theme_set(theme_classic())
```

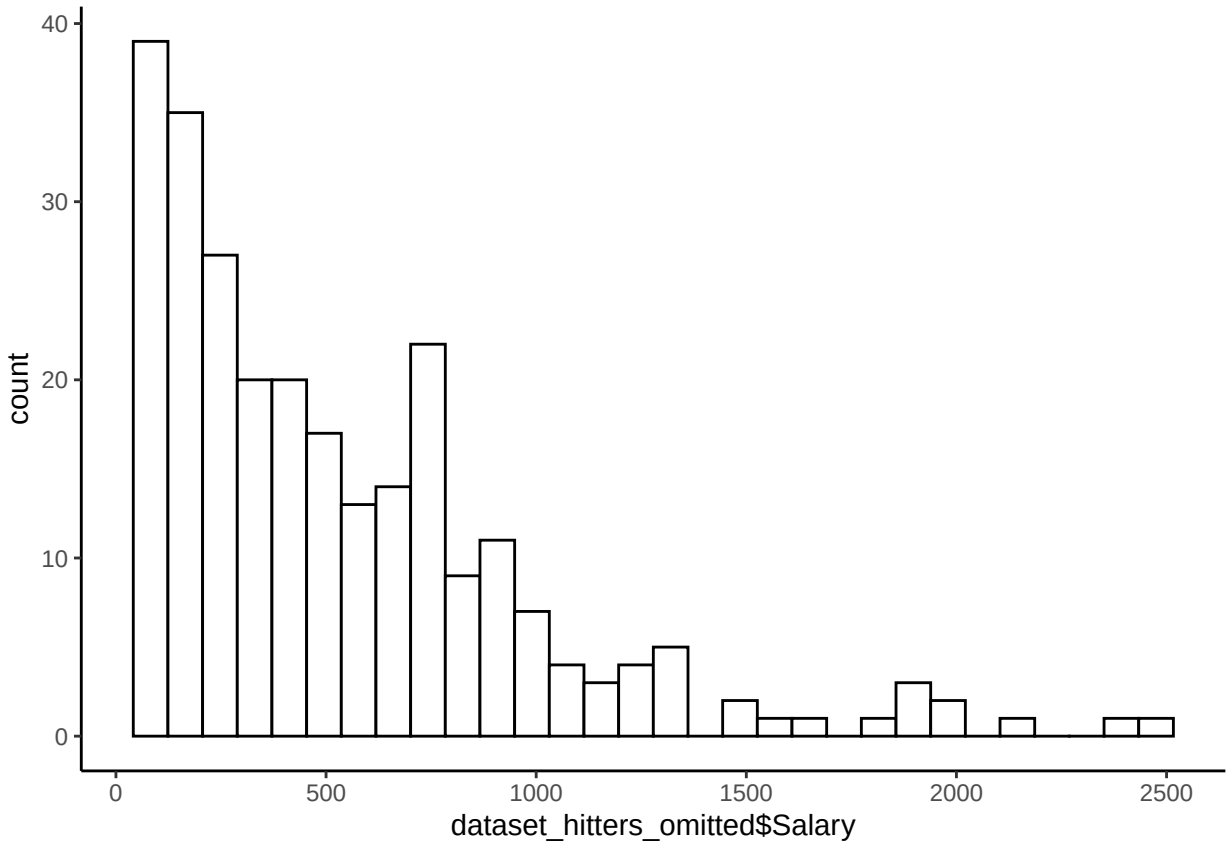
```
#Question 1
#Reading the dataset from the ISLR package
dataset_hitters <-  Hitters

#Omitting the NA records from the dataset
dataset_hitters_omitted <- setDT(na.omit(dataset_hitters))
nrow(dataset_hitters) - nrow(dataset_hitters_omitted)
```

```
## [1] 59
```

59 observations/records do not have salary information. They are removed from the dataset.

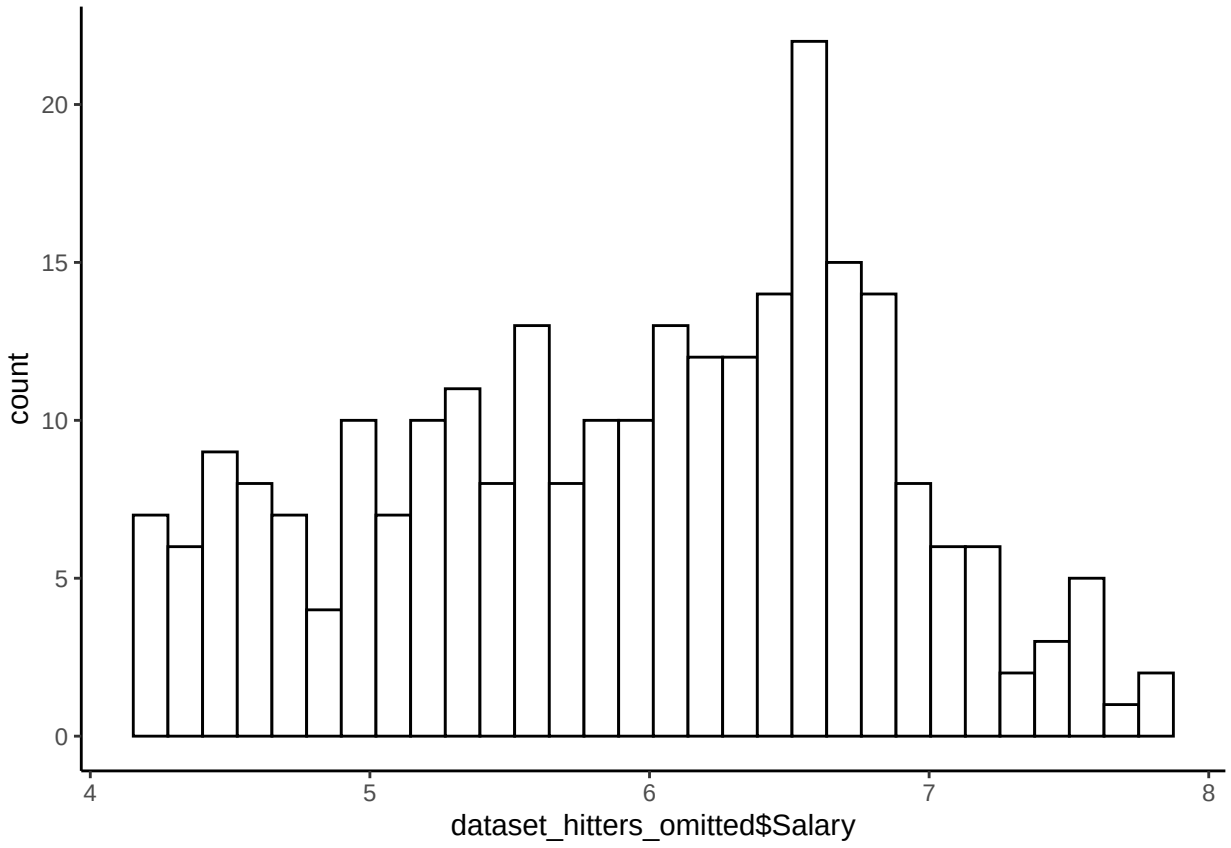
```
#Question 2
ggplot(dataset_hitters_omitted, aes(x=dataset_hitters_omitted$Salary)) +
  geom_histogram(color = "black", fill = "white")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Scaling salary to logarithmic scale to make it normally distributed
dataset_hitters_omitted$Salary <-  log(dataset_hitters_omitted$Salary)
ggplot(dataset_hitters_omitted, aes(x=dataset_hitters_omitted$Salary)) +
  geom_histogram( color = "black", fill = "white")
```

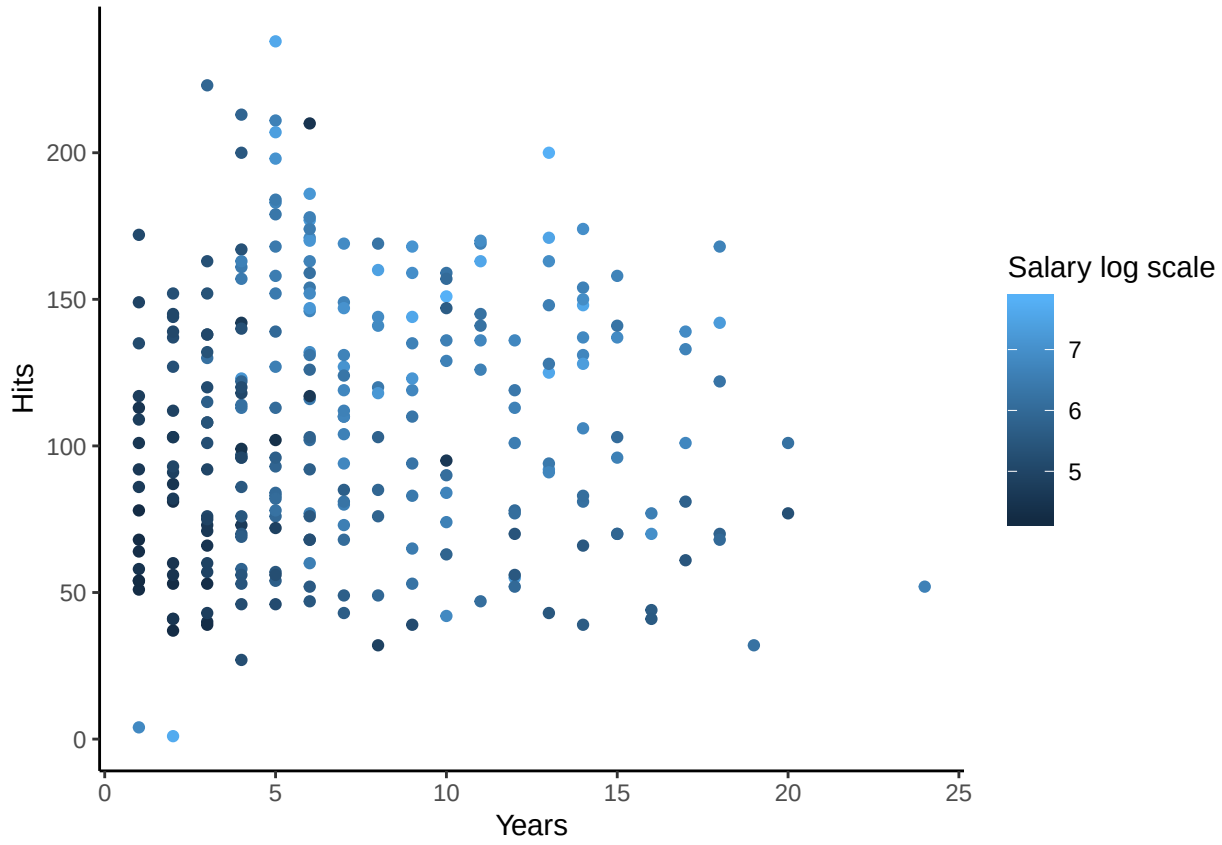
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The plot of the salary seems to be skewed to the right with very few hitters in the upper end of the salary spectrum. Transforming salary into a logarithmic scale removes the skewness and reduces the impact of outliers as one can see from the two histograms above

One of the reason of log transformation is also to check if the transformation helps in producing normally distributed results. In this case, that does not seem to happen.

```
#Question 3
ggplot(dataset_hitters_omitted, aes(x=dataset_hitters_omitted$Years,
                                    y=dataset_hitters_omitted$Hits)) +
  geom_point(aes(col = dataset_hitters_omitted$Salary)) +
  guides(colour = guide_colourbar(order = 1),
         alpha = guide_legend(order = 2),
         size = guide_legend(order = 3),
         fill = guide_legend(reverse = TRUE)) + labs(x="Years", y="Hits",
                                                     color = "Salary log scale")
```



On plotting the Years v/s Hits in the scatter plot and coloring it based on the salary log scale, it seems that Hitters with more number of years are paid more.

```
#Question 4 - Linear Regression and regsubsets

hitters_lm <- lm(Salary~., data = dataset_hitters_omitted)

hitters_exhaustive <- regsubsets(dataset_hitters_omitted$Salary ~ .,
                                 data = dataset_hitters_omitted,
                                 nbest = 1,
                                 nvmax = dim(dataset_hitters_omitted)[2],
                                 method = "exhaustive")

summary_hitters_exhaustive <- summary(hitters_exhaustive)
summary_hitters_exhaustive
```

```
## Subset selection object
## Call: regsubsets.formula(dataset_hitters_omitted$Salary ~ ., data = dataset_hitters_omitted,
##     nbest = 1, nvmax = dim(dataset_hitters_omitted)[2], method = "exhaustive")
## 19 Variables  (and intercept)
##           Forced in Forced out
```

```
## AtBat          FALSE      FALSE
## Hits           FALSE      FALSE
## HmRun          FALSE      FALSE
## Runs           FALSE      FALSE
## RBI            FALSE      FALSE
## Walks          FALSE      FALSE
## Years          FALSE      FALSE
## CAtBat         FALSE      FALSE
## CHits          FALSE      FALSE
## CHmRun         FALSE      FALSE
## CRuns          FALSE      FALSE
## CRBI           FALSE      FALSE
## CWalks         FALSE      FALSE
## LeagueN        FALSE      FALSE
## DivisionW      FALSE      FALSE
## PutOuts        FALSE      FALSE
## Assists        FALSE      FALSE
## Errors         FALSE      FALSE
## NewLeagueN     FALSE      FALSE
## 1 subsets of each size up to 19
## Selection Algorithm: exhaustive
##           AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun CRuns CRBI
## 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 ) "*"   "*"  "*"   "*"  "*" "*"   "*"   "*"    "*"   "*"    "*"   "*"
##           CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 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 ) "*"    "*"     "*"       "*"     "*"     "*"    "*"
```

```
summary_hitters_exhaustive$bic
```

```
##  [1] -117.0304 -156.4291 -159.2777 -159.2182 -159.0885 -157.9207 -157.1229
##  [8] -156.1954 -152.7649 -148.8061 -144.5962 -140.6541 -136.5480 -131.0939
## [15] -125.7112 -120.1995 -114.7125 -109.1859 -103.6145
```

```
summary_hitters_exhaustive$which[which.min(summary_hitters_exhaustive$bic),]
```

```
## (Intercept)       AtBat        Hits       HmRun        Runs         RBI
##        TRUE       FALSE        TRUE       FALSE       FALSE       FALSE
##       Walks       Years      CAtBat       CHits      CHmRun       CRuns
##        TRUE        TRUE       FALSE       FALSE       FALSE       FALSE
##        CRBI      CWalks     LeagueN   DivisionW     PutOuts     Assists
##       FALSE       FALSE       FALSE       FALSE       FALSE       FALSE
##      Errors  NewLeagueN
##       FALSE       FALSE
```

When using BIC to evaluate and compare subsets of predictors, the model with the lowest BIC value is the best model.

The predictor variables included in the best model (the model with the smallest BIC value) are: Hits, Walks, and Years.

```
#Question 5 - Training and test datasets
set.seed(42)
train.index <- sample(1:nrow(dataset_hitters_omitted), 0.8*(nrow(dataset_hitters_omitted)))
hitters_train <- dataset_hitters_omitted[train.index, ]
hitters_valid<- dataset_hitters_omitted[-train.index, ]
nrow(hitters_train)
```

```
## [1] 210
```

```
nrow(hitters_valid)
```

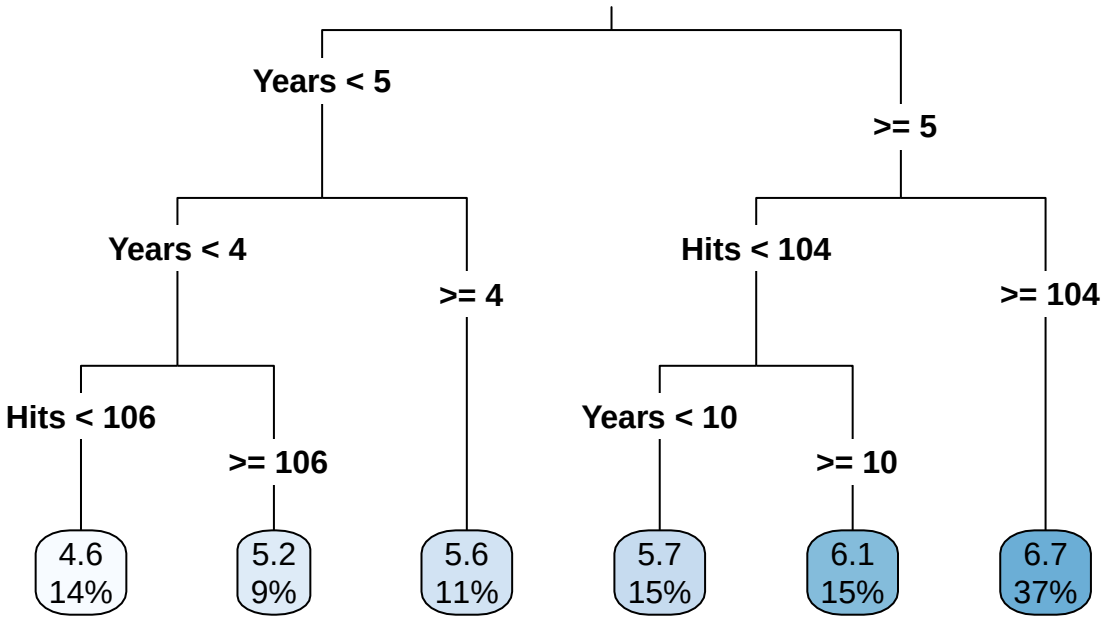
```
## [1] 53
```

```
# Question 6- regression tree model
rpart.hitters <-  rpart(Salary ~ Years + Hits, hitters_train, method = "anova")
summary(rpart.hitters)
```

```
## Call:
## rpart(formula = Salary ~ Years + Hits, data = hitters_train,
##     method = "anova")
##   n= 210
##
##           CP nsplit rel error    xerror       xstd
## 1 0.45475137      0 1.0000000 1.0110240 0.07570057
## 2 0.13900068      1 0.5452486 0.5509269 0.05595155
## 3 0.05075809      2 0.4062479 0.4157786 0.05064154
## 4 0.02121570      3 0.3554899 0.3672499 0.04962768
## 5 0.01228426      4 0.3342742 0.3738799 0.05790050
## 6 0.01000000      5 0.3219899 0.3830406 0.05817839
##
## Variable importance
## Years  Hits
##    74    26
##
## Node number 1: 210 observations,    complexity param=0.4547514
##   mean=5.920716, MSE=0.7572154
##   left son=2 (70 obs) right son=3 (140 obs)
##   Primary splits:
##       Years < 4.5   to the left,  improve=0.4547514, (0 missing)
##       Hits  < 109   to the left,  improve=0.2450835, (0 missing)
##   Surrogate splits:
##       Hits < 29.5  to the left,  agree=0.676, adj=0.029, (0 split)
##
## Node number 2: 70 observations,    complexity param=0.05075809
##   mean=5.090843, MSE=0.3939711
##   left son=4 (47 obs) right son=5 (23 obs)
##   Primary splits:
##       Years < 3.5   to the left,  improve=0.2926723, (0 missing)
##       Hits  < 112.5 to the left,  improve=0.2431599, (0 missing)
##   Surrogate splits:
##       Hits < 154.5 to the left,  agree=0.729, adj=0.174, (0 split)
##
## Node number 3: 140 observations,    complexity param=0.1390007
##   mean=6.335653, MSE=0.4223205
##   left son=6 (63 obs) right son=7 (77 obs)
##   Primary splits:
##       Hits  < 103.5 to the left,  improve=0.37383980, (0 missing)
##       Years < 6.5   to the left,  improve=0.04037696, (0 missing)
##   Surrogate splits:
##       Years < 14.5  to the right, agree=0.593, adj=0.095, (0 split)
##
## Node number 4: 47 observations,    complexity param=0.0212157
##   mean=4.853302, MSE=0.2676461
##   left son=8 (29 obs) right son=9 (18 obs)
##   Primary splits:
##       Hits  < 105.5 to the left,  improve=0.26818680, (0 missing)
##       Years < 2.5   to the left,  improve=0.09899798, (0 missing)
##   Surrogate splits:
##       Years < 2.5   to the left,  agree=0.638, adj=0.056, (0 split)
##
## Node number 5: 23 observations
```

```
##   mean=5.576252, MSE=0.3011869
##
## Node number 6: 63 observations,    complexity param=0.01228426
##   mean=5.896375, MSE=0.290369
##   left son=12 (32 obs) right son=13 (31 obs)
##   Primary splits:
##       Years < 9.5   to the left,  improve=0.10678170, (0 missing)
##       Hits  < 69    to the left,  improve=0.09000896, (0 missing)
##   Surrogate splits:
##       Hits < 60.5  to the left,  agree=0.556, adj=0.097, (0 split)
##
## Node number 7: 77 observations
##   mean=6.695062, MSE=0.2432259
##
## Node number 8: 29 observations
##   mean=4.642228, MSE=0.2481879
##
## Node number 9: 18 observations
##   mean=5.193367, MSE=0.1115722
##
## Node number 12: 32 observations
##   mean=5.723063, MSE=0.1975065
##
## Node number 13: 31 observations
##   mean=6.075278, MSE=0.3232145
```

```r
#plotting the visual tree
rpart.plot(rpart.hitters, type = 3)
```



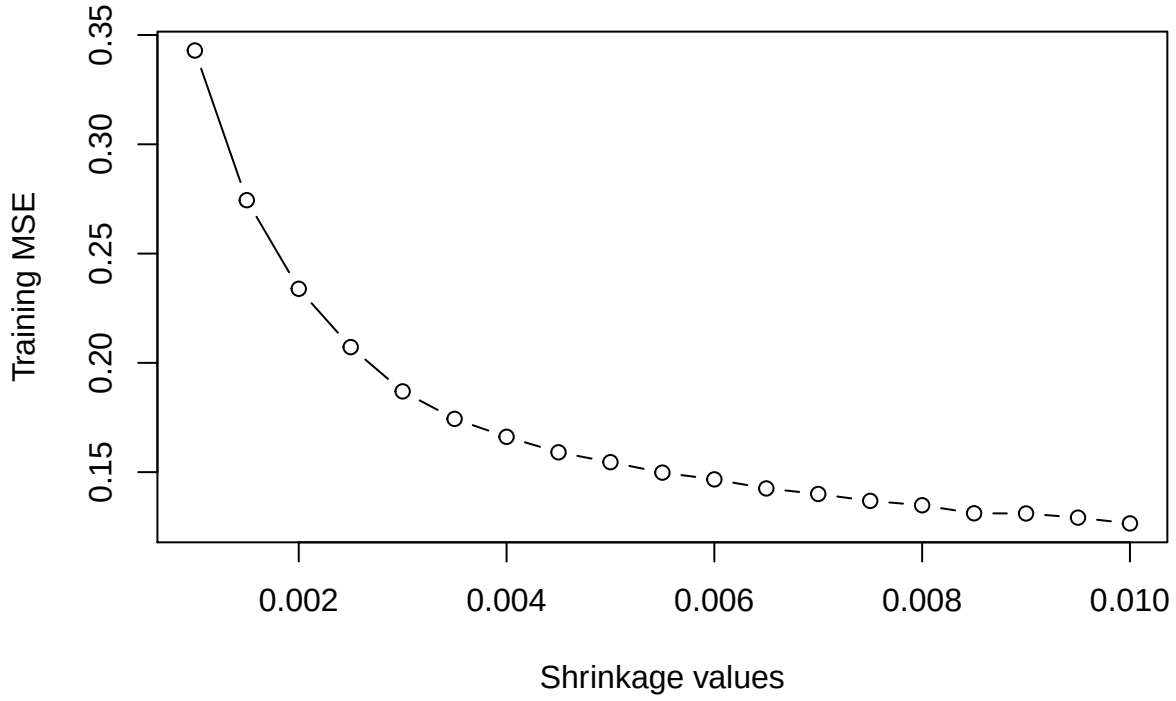

```
# Outputting the regression rules
rpart.rules(rpart.hitters)
```

```
##  Salary
##     4.6 when Years <  4       & Hits <  106
##     5.2 when Years <  4       & Hits >= 106
##     5.6 when Years is 4 to  5
##     5.7 when Years is 5 to 10 & Hits <  104
##     6.1 when Years >=      10 & Hits <  104
##     6.7 when Years >=       5 & Hits >= 104
```

As we can see, salary is highest for players who have experience years >= 5 and hits >= 104.

```
#Question 7-Regression trees using all the data

#choosing lambda values ranging from .001 to .01 incrementing by .0005
lambdaVals = seq(.001, 0.01 , by = .0005)
train.err = rep(NA, length(lambdaVals))
#Performing boosting on training data set using range of value of lambda
for (i in 1:length(lambdaVals)) {
    boost.hitters = gbm(Salary ~ ., data = hitters_train, distribution = "gaussian",
```

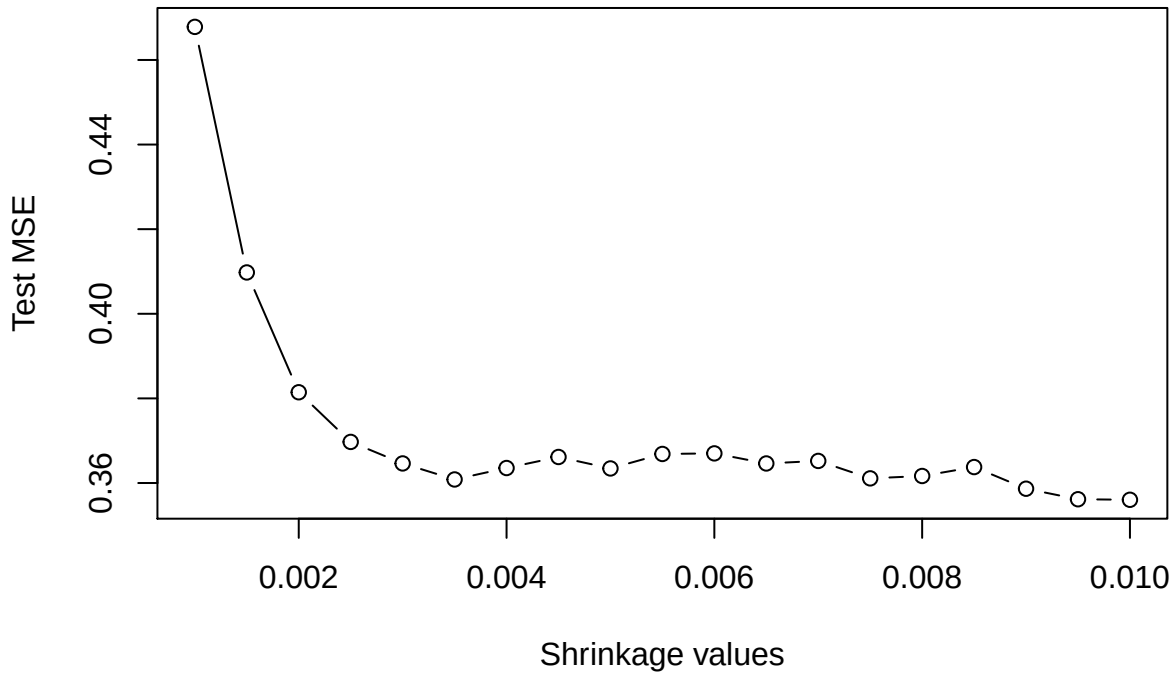
```
                        n.trees = 1000, shrinkage = lambdaVals[i])
    pred.train = predict(boost.hitters, hitters_train, n.trees = 1000)
    train.err[i] = mean((pred.train - hitters_train$Salary)^2)
}
#Plotting the MSE VS Shrinkage values
plot(lambdaVals, train.err, type = "b", xlab = "Shrinkage values", ylab = "Training MSE")
```



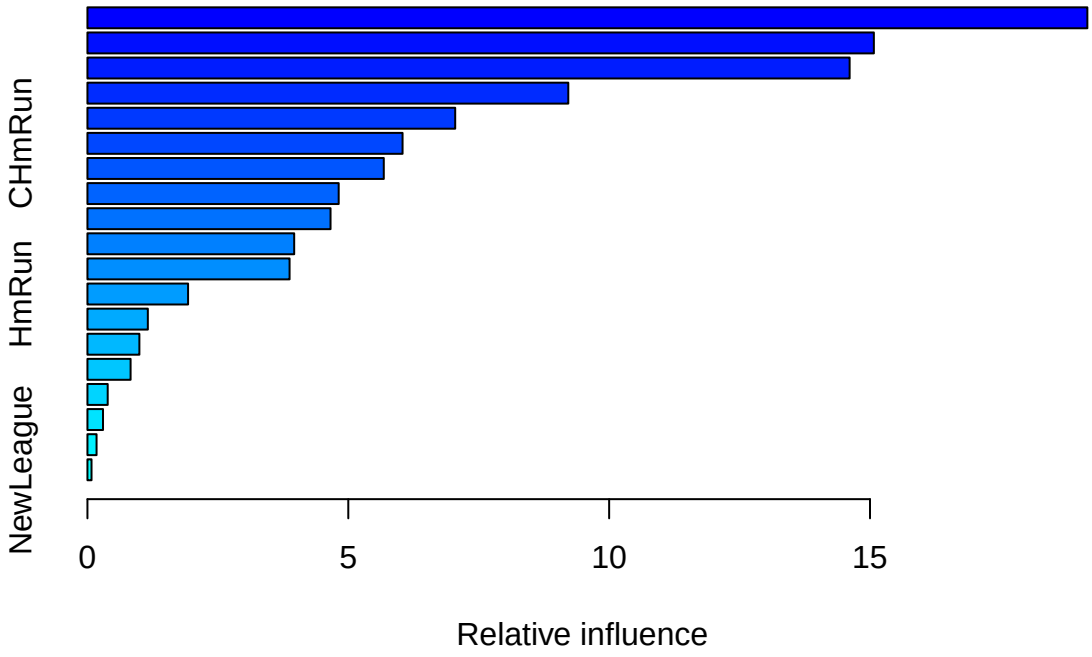
As per the graph, MSE keeps going down as the lambda going up. Among 0.002 to 0-01, the best lambda is 1, which yields the minimum MSE (0.000897402).

```
#Question 8 Produce a plot with different shrinkage values on the x-axis and the
#corresponding test set MSE on the y-axis.
set.seed(42)
test.err <- rep(NA, length(lambdaVals))
#Calculating MSE for test data for different values of lambda
for (i in 1:length(lambdaVals)) {
    boost.hitters = gbm(Salary ~ ., data = hitters_train, distribution = "gaussian",
                        n.trees = 1000, shrinkage = lambdaVals[i])
    pred.test = predict(boost.hitters, hitters_valid, n.trees = 1000)
    test.err[i] = mean((pred.test - hitters_valid$Salary)^2)
```

```
}
#Plotting Shrinkage values corresponding to MSE for Test Data set
plot(lambdaVals, test.err, type = "b", xlab = "Shrinkage values", ylab = "Test MSE")
```




```
#Question 9  Which variables appear to be the most important predictors in the boosted model?

summary(boost.hitters)
```




```
##                 var     rel.inf
## CAtBat       CAtBat 19.16654985
## CRBI           CRBI 15.07423605
## CRuns         CRuns 14.60964512
## Years         Years  9.21701673
## CHits         CHits  7.04917557
## CHmRun       CHmRun  6.04037476
## CWalks       CWalks  5.68025319
## Hits           Hits  4.81598625
## PutOuts     PutOuts  4.65987525
## RBI             RBI  3.96296823
## Walks         Walks  3.87450303
## HmRun         HmRun  1.92911791
## AtBat         AtBat  1.15798676
## Errors       Errors  0.99560404
## Runs           Runs  0.82678396
## Assists     Assists  0.38856504
## Division   Division  0.29861861
## League       League  0.17468537
## NewLeague NewLeague  0.07805429
```

Answer: CAtBat and CRBI are the most important predictors in the boosted model.

```
#QUESTION 10: Now apply bagging to the training set.
#What is the test set MSE for this approach?
bag.model <- randomForest(Salary~., data=hitters_train,
                          mtry = 13, importance = TRUE)
test_pred <- predict(bag.model, newdata=hitters_valid)
mean((test_pred-hitters_valid$Salary)^2)
```

```
## [1] 0.2447208
```

Answer: 0.244 is the test set MSE.