HW4

Siddharth Agarwal

19/04/2021

load pakages

```
if(!require('pacman')) install.packages('pacman')
```

```
## Loading required package: pacman
```

```
pacman::p_load(ISLR, MASS, rpart, rpart.plot, caret, leaps, randomForest,
               gbm, tree, ggplot2,dplyr, tinytex)


#tinytex::install_tinytex()
```

Q1. Remove the observations with unknown salary information.

How many observations were removed in this process?

```
data('Hitters')

summary(Hitters$Salary)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    67.5   190.0   425.0   535.9   750.0  2460.0      59
```

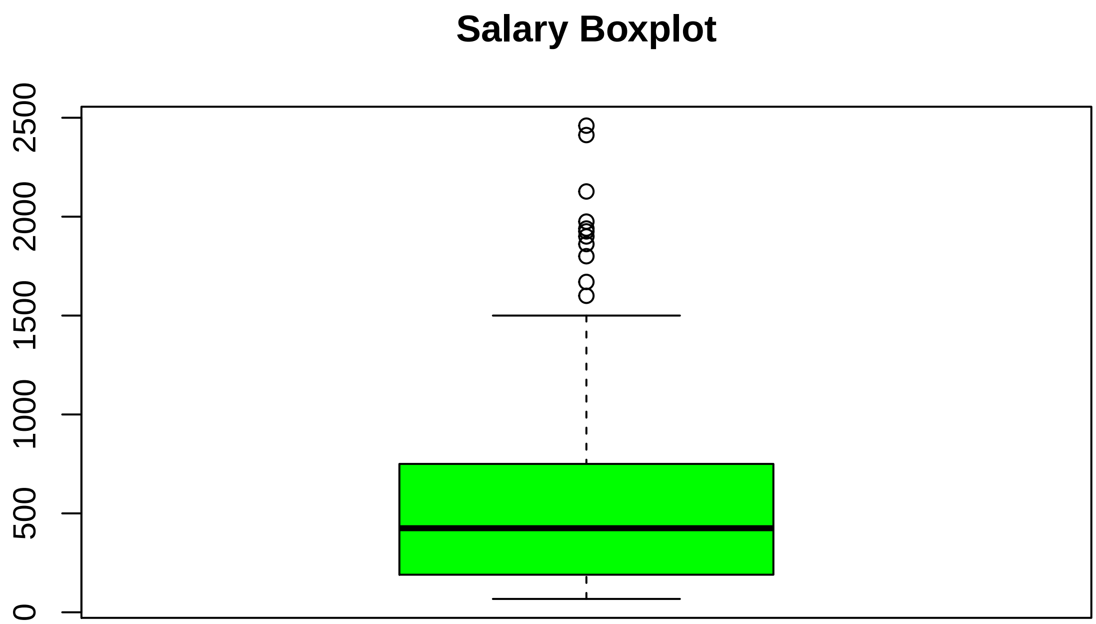
```
Hitters.modified <- subset(Hitters, !is.na(Hitters$Salary))

#summary(Hitters.modified)
```

Ans 1: 59 observations having unknown salary were removed

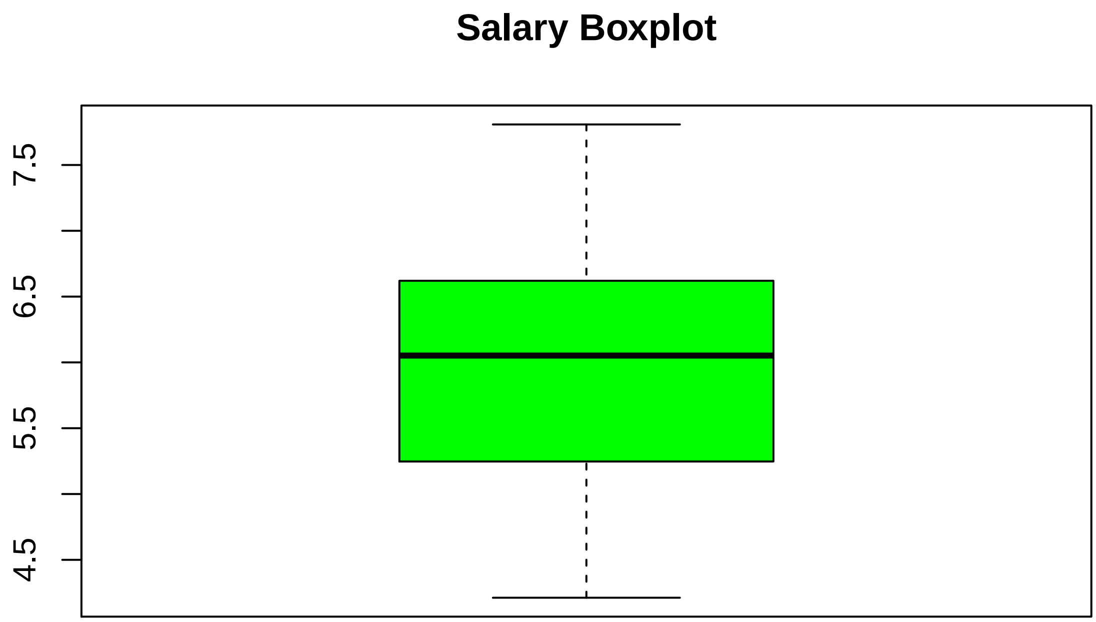
Q2. Transform the salaries using a (natural) log transformation.

Is there any justification for this transformation? Explain your answer.

```r
boxplot(Hitters.modified$Salary, col = 'green', main='Salary Boxplot')
```




```r
Hitters.modified <- transform(Hitters.modified,
                              log.salary = log(Hitters.modified$Salary))


boxplot(Hitters.modified$log.salary, col = 'green', main='Salary Boxplot')
```




```r
head(Hitters.modified)
```

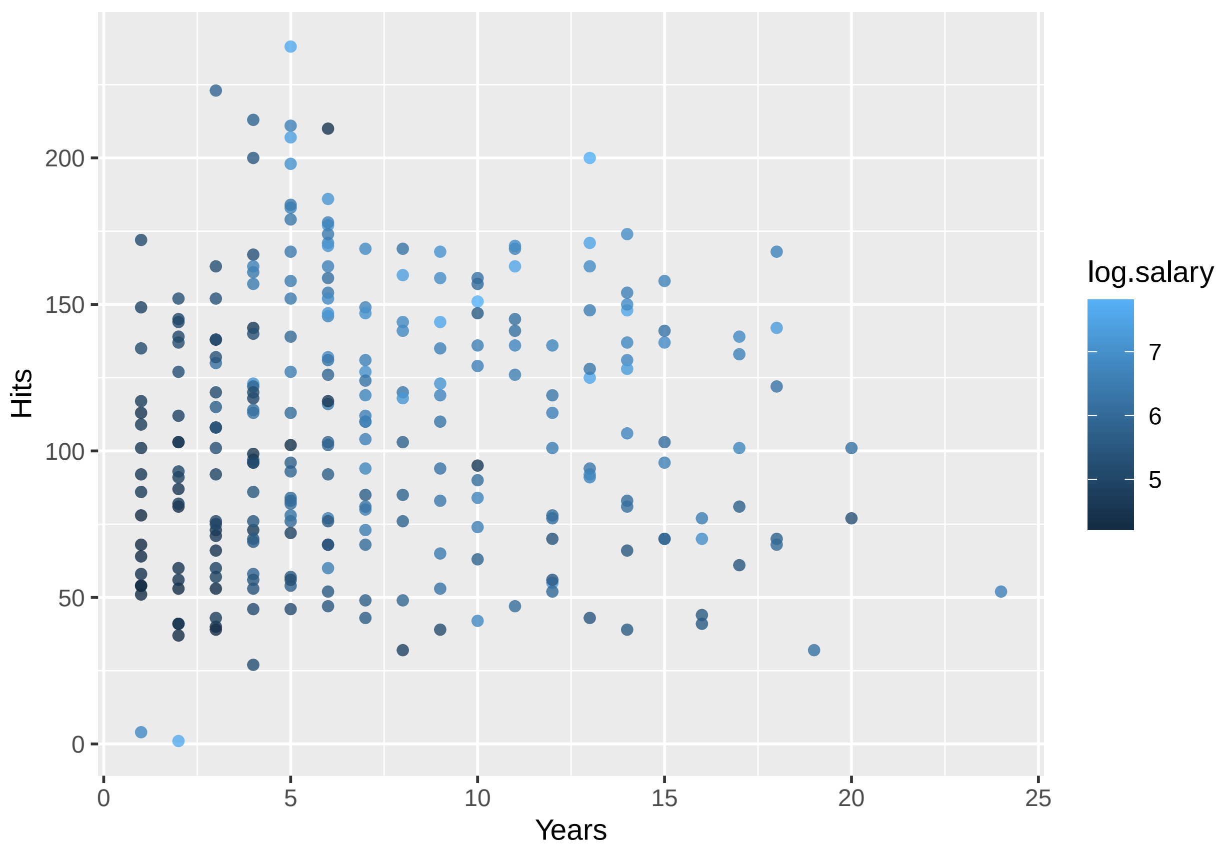
```
##                    AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun
## -Alan Ashby          315   81     7   24  38    39    14   3449   835     69
## -Alvin Davis         479  130    18   66  72    76     3   1624   457     63
## -Andre Dawson        496  141    20   65  78    37    11   5628  1575    225
## -Andres Galarraga    321   87    10   39  42    30     2    396   101     12
## -Alfredo Griffin     594  169     4   74  51    35    11   4408  1133     19
## -Al Newman           185   37     1   23   8    21     2    214    42      1
##                    CRuns CRBI CWalks League Division PutOuts Assists Errors
## -Alan Ashby          321  414    375      N        W     632      43     10
## -Alvin Davis         224  266    263      A        W     880      82     14
## -Andre Dawson        828  838    354      N        E     200      11      3
## -Andres Galarraga     48   46     33      N        E     805      40      4
## -Alfredo Griffin     501  336    194      A        W     282     421     25
## -Al Newman            30    9     24      N        E      76     127      7
##                    Salary NewLeague log.salary
## -Alan Ashby         475.0         N   6.163315
## -Alvin Davis        480.0         A   6.173786
## -Andre Dawson       500.0         N   6.214608
## -Andres Galarraga    91.5         N   4.516339
## -Alfredo Griffin    750.0         A   6.620073
## -Al Newman           70.0         A   4.248495
```

Ans 2: The data for Salary column is positively skewed as all the outlier

values have values greater than 1500. Hence it would be better to perform a log transformation here to get the data to be more normal in its distribution

Q3. Create a scatterplot with Hits on the y-axis and Years on the x-axis using all the observations. Color code the observations using the log Salary variable. What patterns do you notice on this chart, if any?

```
ggplot(Hitters.modified, aes(y=Hits, x=Years, color= log.salary)) +
  geom_point(alpha=0.8)
```



Ans 3: The graph shows that Salary tends to increase as the number of Years or Hits increase

We can see that there are few very low salaries (log.salary ~ 5 or less) for Years >5.

For Hits we do not observe any particular trend for salary values especially for Years of experiance < 5 as low salaries are distributed throughout the vertical axis.

There are few very high salaries (outliers) in the graph. One particular surprising observation is a very high salary for (Years = 2 and Hits =0)

Q4. Run a linear regression model of Log Salary on all the predictors using the entire dataset. Use regsubsets() function to perform best subset selection from the regression model.

Identify the best model using BIC.

Which predictor variables are included in this (best) model?

```r
hitter.lm <- lm(log.salary ~. -Salary, Hitters.modified)
summary(hitter.lm)
```

```
##
## Call:
## lm(formula = log.salary ~ . - Salary, data = Hitters.modified)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.22870 -0.45350  0.09424  0.40474  2.77223
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.618e+00  1.765e-01  26.171  < 2e-16 ***
## AtBat       -2.984e-03  1.232e-03  -2.421  0.01620 *
## Hits         1.308e-02  4.622e-03   2.831  0.00503 **
## HmRun        1.179e-02  1.205e-02   0.978  0.32889
## Runs        -1.419e-03  5.794e-03  -0.245  0.80670
## RBI         -1.675e-03  5.056e-03  -0.331  0.74063
## Walks        1.096e-02  3.554e-03   3.082  0.00229 **
## Years        5.696e-02  2.413e-02   2.361  0.01902 *
## CAtBat       1.283e-04  2.629e-04   0.488  0.62596
## CHits       -4.414e-04  1.311e-03  -0.337  0.73670
## CHmRun      -7.809e-05  3.144e-03  -0.025  0.98020
## CRuns        1.513e-03  1.459e-03   1.037  0.30072
## CRBI         1.312e-04  1.346e-03   0.097  0.92246
## CWalks      -1.466e-03  6.377e-04  -2.298  0.02239 *
## LeagueN      2.825e-01  1.541e-01   1.833  0.06797 .
## DivisionW   -1.656e-01  7.847e-02  -2.111  0.03580 *
## PutOuts      3.389e-04  1.505e-04   2.251  0.02526 *
## Assists      6.214e-04  4.300e-04   1.445  0.14970
## Errors      -1.197e-02  8.537e-03  -1.402  0.16225
```

```
## NewLeagueN  -1.742e-01  1.536e-01  -1.134  0.25788
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6135 on 243 degrees of freedom
## Multiple R-squared:  0.5586, Adjusted R-squared:  0.524
## F-statistic: 16.18 on 19 and 243 DF,  p-value: < 2.2e-16
```

```r
#BIC

set.seed(42)

hitter.BIC <- regsubsets(log.salary~.-Salary, data = Hitters.modified,
                         nbest = 1, nvmax = 19, method = 'seq')

sum <- summary(hitter.BIC)

sum$which
```

```
##    (Intercept) AtBat  Hits HmRun  Runs   RBI Walks Years CAtBat CHits CHmRun
## 1         TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  FALSE FALSE  FALSE
## 2         TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  FALSE  TRUE  FALSE
## 3         TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  FALSE  TRUE  FALSE
## 4         TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE   TRUE FALSE  FALSE
## 5         TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  FALSE FALSE  FALSE
## 6         TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  FALSE  TRUE  FALSE
## 7         TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  FALSE  TRUE  FALSE
## 8         TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  FALSE FALSE  FALSE
## 9         TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  FALSE FALSE  FALSE
## 10        TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  FALSE FALSE  FALSE
## 11        TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  FALSE FALSE  FALSE
## 12        TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  FALSE FALSE  FALSE
## 13        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE   TRUE
## 14        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE   TRUE
## 15        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE   TRUE
## 16        TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE   TRUE  TRUE  FALSE
## 17        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE   TRUE
## 18        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE  FALSE
## 19        TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE   TRUE  TRUE   TRUE
##    CRuns  CRBI CWalks LeagueN DivisionW PutOuts Assists Errors NewLeagueN
## 1   TRUE FALSE  FALSE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 2  FALSE FALSE  FALSE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 3  FALSE FALSE  FALSE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 4  FALSE FALSE  FALSE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 5  FALSE FALSE  FALSE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 6  FALSE FALSE  FALSE   FALSE      TRUE   FALSE   FALSE  FALSE      FALSE
## 7  FALSE FALSE  FALSE   FALSE      TRUE    TRUE   FALSE  FALSE      FALSE
## 8   TRUE FALSE   TRUE   FALSE      TRUE    TRUE   FALSE  FALSE      FALSE
## 9   TRUE FALSE   TRUE    TRUE      TRUE    TRUE   FALSE  FALSE      FALSE
## 10  TRUE FALSE   TRUE    TRUE      TRUE    TRUE   FALSE  FALSE       TRUE
## 11  TRUE FALSE   TRUE    TRUE      TRUE    TRUE   FALSE  FALSE       TRUE
## 12  TRUE FALSE   TRUE    TRUE      TRUE    TRUE    TRUE   TRUE      FALSE
## 13  TRUE  TRUE   TRUE   FALSE     FALSE   FALSE   FALSE  FALSE      FALSE
## 14  TRUE  TRUE   TRUE    TRUE     FALSE   FALSE   FALSE  FALSE      FALSE
```

```
## 15   TRUE  TRUE   TRUE    TRUE      TRUE   FALSE   FALSE  FALSE      FALSE
## 16   TRUE FALSE   TRUE    TRUE      TRUE    TRUE    TRUE   TRUE       TRUE
## 17   TRUE  TRUE   TRUE    TRUE      TRUE    TRUE    TRUE  FALSE      FALSE
## 18   TRUE  TRUE   TRUE    TRUE      TRUE    TRUE    TRUE   TRUE       TRUE
## 19   TRUE  TRUE   TRUE    TRUE      TRUE    TRUE    TRUE   TRUE       TRUE
```

```
sum$bic
```

```
##  [1] -117.03045 -156.35434 -158.14586 -159.21816  -36.91193 -157.92069
##  [7] -156.97937 -156.19540 -152.76488 -148.80615 -144.59624 -140.65413
## [13] -120.32941 -118.07255 -117.47646 -120.19950 -111.48262 -109.18595
## [19] -103.61446
```

In the best model as determined by lowest value of BIC,

the variables 'AtBat' , 'Hits', 'Walks' and 'CAtBat' should be included

Q5. Now create a training data set consisting of 80 percent of the observations, and a test data set consisting of the remaining 20 percent of the observations.

```
set.seed(42)

train <- sample(1:nrow(Hitters.modified), round(nrow(Hitters.modified)*0.8))
train.data <- Hitters.modified[train,]
test.data <- Hitters.modified[-train,]
```

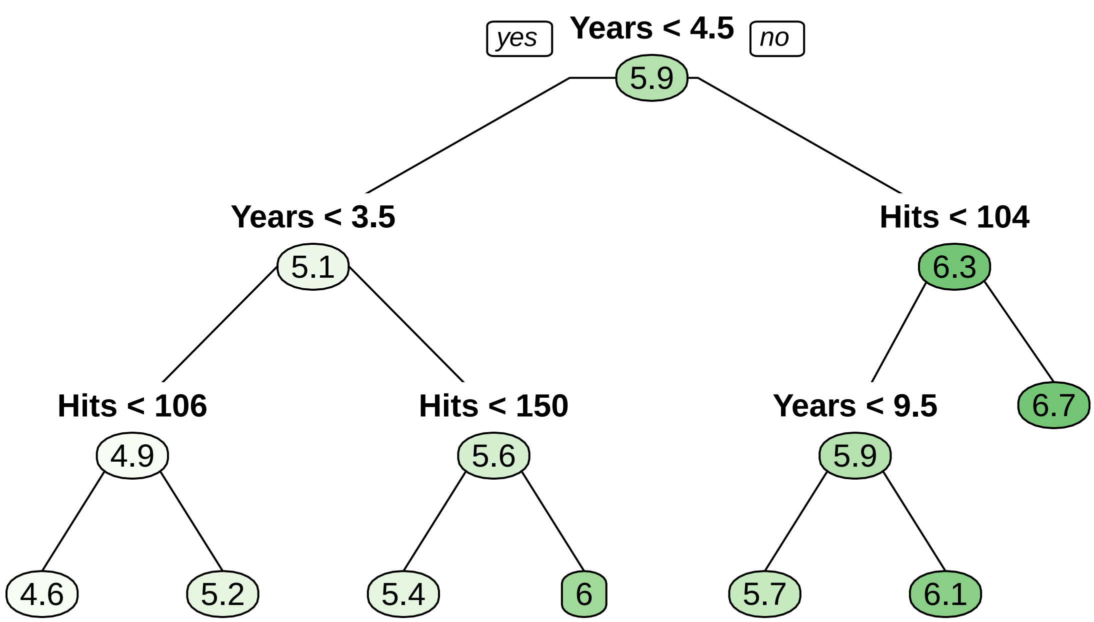
Q6. Generate a regression tree of log Salary using only Years and Hits variables from the training data set. Which players are likely to receive highest salaries according to this model? Write down the rule and elaborate on it.

```
#regression

tree.reg <- rpart(log.salary ~ Years+Hits, data = train.data, cp= 0.01,
                  minsplit=10, xval=10)

printcp(tree.reg)
```

```
##
## Regression tree:
## rpart(formula = log.salary ~ Years + Hits, data = train.data,
##     cp = 0.01, minsplit = 10, xval = 10)
##
```

```
## Variables actually used in tree construction:
## [1] Hits  Years
##
## Root node error: 159.02/210 = 0.75722
##
## n= 210
##
##          CP nsplit rel error  xerror     xstd
## 1 0.454751      0   1.00000 1.01102 0.075701
## 2 0.139001      1   0.54525 0.55093 0.055952
## 3 0.050758      2   0.40625 0.41578 0.050642
## 4 0.021216      3   0.35549 0.36725 0.049628
## 5 0.012284      4   0.33427 0.37388 0.057901
## 6 0.011025      5   0.32199 0.38741 0.058225
## 7 0.010000      6   0.31097 0.38115 0.058147
```

```r
# plotting tree

prp(tree.reg, type = 1, under= TRUE, roundint = FALSE, split.font = 2,
    varlen = -10, box.palette = 'Green')
```




```r
# rules

rpart.rules(tree.reg, cover = TRUE)
```

```
##   log.salary                                              cover
```

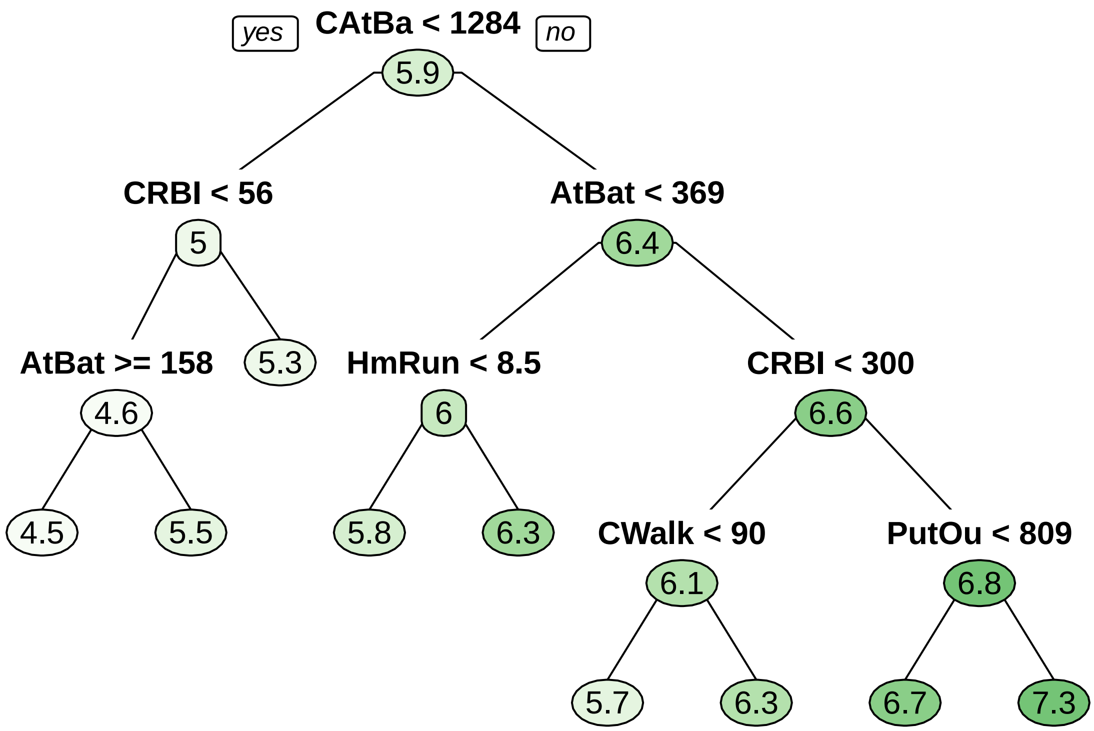
```
##          4.6 when Years <  4        & Hits <  106      14%
##          5.2 when Years <  4        & Hits >= 106       9%
##          5.4 when Years is 4 to  5 & Hits <  150       8%
##          5.7 when Years is 5 to 10 & Hits <  104      15%
##          6.0 when Years is 4 to  5 & Hits >= 150       3%
##          6.1 when Years >=      10 & Hits <  104      15%
##          6.7 when Years >=       5 & Hits >= 104      37%
```

Rule: log.salary = 6.7 for Years >= 5 AND Hits >= 104. The players who receive the highest salary are the one who have played for 5 years or more and made 104 hits or more. These players get receive a salary of an average 6.7 log salary which is almost equivalent to 812K in salary and it covers 37% of data

Q7 Now create a regression tree using all the variables in the training data set. Perform boosting on the training set with 1,000 trees for a range of values of the shrinkage parameter.

Produce a plot with different shrinkage values on the x-axis and the corresponding training set MSE on the y-axis.

```
set.seed(42)

reg.all <- rpart(log.salary~.-Salary, data = train.data, cp=0.01, minsplit=10,
                 xval=10)

prp(reg.all, type=1, under=TRUE, roundint=FALSE, split.font=2, varlen=-5,
    box.palette = 'Green')
```

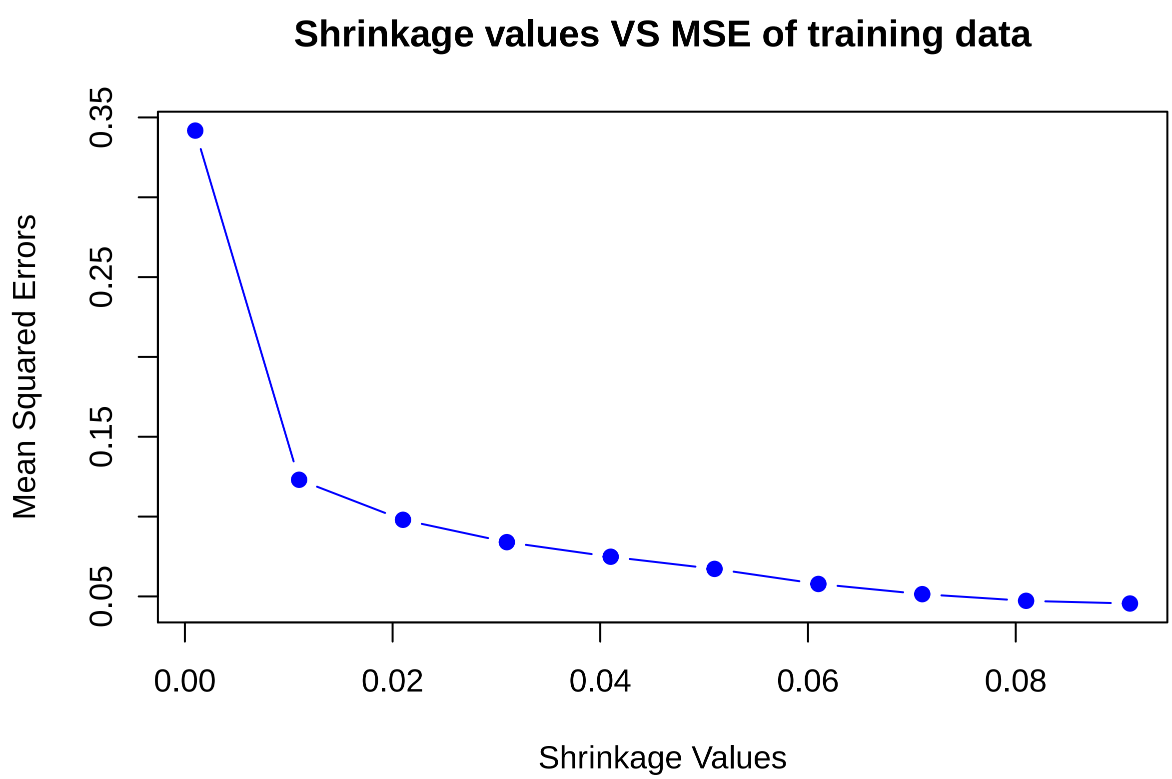



```
# Shrinkage parameter and MSE

lambda <- seq(0.001, 0.1, by=0.01)
MSE <- rep(NA, length(lambda))

#Boosting

for(i in 1: length(lambda))
{
  boost.salary <- gbm(log.salary~.-Salary, data = train.data, n.trees = 1000,
                      distribution = "gaussian", shrinkage = lambda[i])

  predictions <- predict(boost.salary, train.data, n.trees=1000,
                         shrinkage=lambda[i])

  MSE[i] <- mean((predictions - train.data$log.salary)^2)
}

y <- data.frame(MSE, lambda)

y
```

```
##           MSE lambda
## 1  0.34174773  0.001
## 2  0.12305899  0.011
## 3  0.09798219  0.021
```

```
## 4  0.08399983  0.031
## 5  0.07485064  0.041
## 6  0.06724112  0.051
## 7  0.05780616  0.061
## 8  0.05141976  0.071
## 9  0.04724223  0.081
## 10 0.04556911  0.091
```

```r
plot(lambda, MSE, pch=19, col='blue', type = 'b', xlab = 'Shrinkage Values',
     ylab='Mean Squared Errors',
     main = 'Shrinkage values VS MSE of training data')
```



We can observe that as Shrinkage value increase MSE value decreases.

Q8 Produce a plot with different shrinkage values on the x-axis and

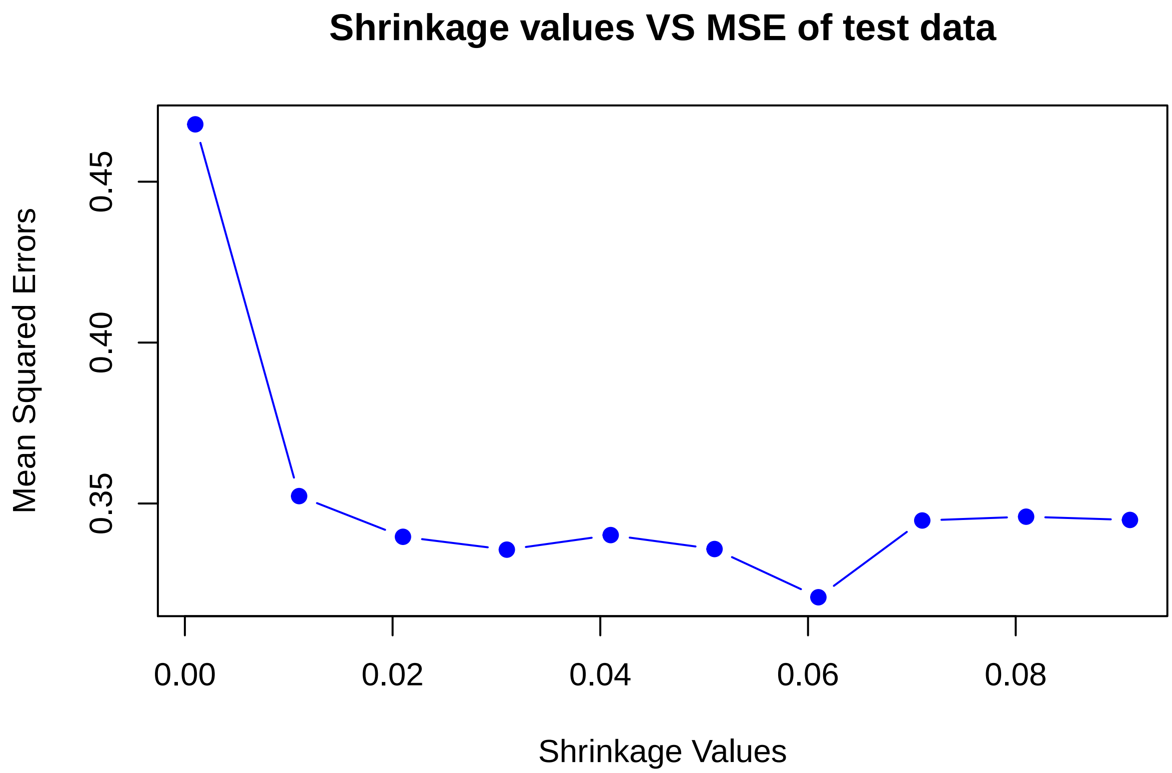
the corresponding test set MSE on the y-axis.

```r
set.seed(42)

MSE2 <- rep(NA, length(lambda))

for(i in 1: length(lambda))
{
```

```
  boost.test <- gbm(log.salary~.-Salary, data = train.data, n.trees = 1000,
                    distribution = "gaussian", shrinkage = lambda[i])

  predictions.test <- predict(boost.test, test.data, n.trees=1000)

  MSE2[i] <- mean((predictions.test - test.data$log.salary)^2)
}

y1 <- data.frame(MSE2, lambda)

y1
```

```
##         MSE2 lambda
## 1  0.4678175  0.001
## 2  0.3522949  0.011
## 3  0.3396510  0.021
## 4  0.3356590  0.031
## 5  0.3401854  0.041
## 6  0.3358513  0.051
## 7  0.3208693  0.061
## 8  0.3447181  0.071
## 9  0.3458967  0.081
## 10 0.3448906  0.091
```

```
plot(lambda, MSE2, pch=19, col='blue', type = 'b', xlab = 'Shrinkage Values',
     ylab='Mean Squared Errors', main = 'Shrinkage values VS MSE of test data')
```



We can observe that as the lambda value increases the MSE values decrease. ### lowest value of MSE is observed at lambda value 0.061

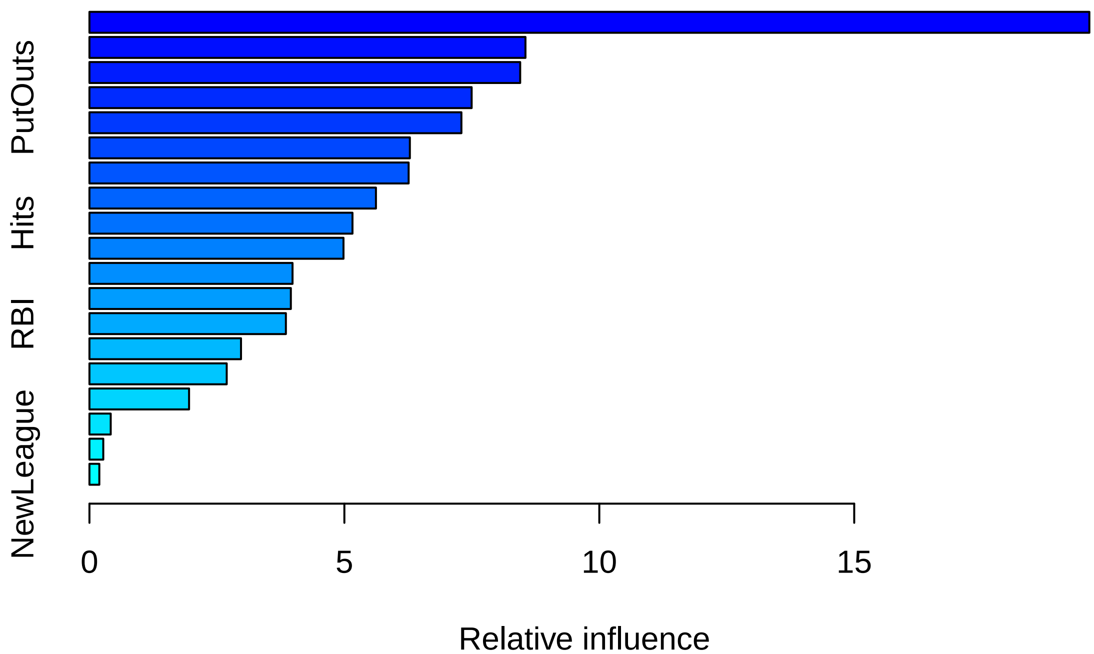
Q9 Which variables appear to be the most important predictors

in the boosted model?

```
set.seed(42)

# Boosting using best shrinkage value from last question

boost.train <- gbm(log.salary~. - Salary, data = train.data, n.trees = 1000,
                   shrinkage = 0.061)
```

```
## Distribution not specified, assuming gaussian ...
```

```
summary(boost.train)
```




```
##                 var    rel.inf
## CAtBat       CAtBat 19.6118993
## CRBI           CRBI  8.5521956
## Years         Years  8.4482322
## PutOuts     PutOuts  7.4966800
## CRuns         CRuns  7.2955661
## CHmRun       CHmRun  6.2845306
## CWalks       CWalks  6.2605352
## CHits         CHits  5.6194432
## Hits           Hits  5.1592673
## Walks         Walks  4.9827803
## HmRun         HmRun  3.9833972
## AtBat         AtBat  3.9501757
## RBI             RBI  3.8532987
## Errors       Errors  2.9726527
## Runs           Runs  2.6922767
## Assists     Assists  1.9540525
## League       League  0.4171273
## Division   Division  0.2716072
## NewLeague NewLeague  0.1942821
```

The most important predictors for the boosted model are CAtBat, CRBI and CRuns

Q10. Now apply bagging to the training set.

What is the test set MSE for this approach?

```
set.seed(42)

bag <- randomForest(log.salary~. - Salary, data = train.data, mtry=19,
                    importance=TRUE)
bag.pred <- predict(bag, test.data)

mse.bag <- mean((bag.pred- test.data$log.salary)^2)

mse.bag
```

```
## [1] 0.2463472
```

Test MSE value for bagging approach is 0.2463472