NYC Bike Share Analysis

Ed Powell

March 3, 2019

```
#install.packages("dplyr")
#install.packages("ggplot2")
#install.packages("lubridate")
#install.packages("geosphere")
#install.packages("xts")
#install.packages("randomForest")
#install.packages("caret")
#install.packages("lattice")
#install.packages("e1071")
#install.packages("splitstackshape")
#install.packages("tree")
#install.packages("devtools")
#install.packages("knitr")
#install.packages("markdown")
#install.packages("rmarkdown")


library(dplyr)
library(ggplot2)
library(lubridate)
library(geosphere)
library(xts)
library(randomForest)
library(caret)
library(e1071)
library(splitstackshape)
library(tree)
library(devtools)
library(knitr)
library(markdown)
library(rmarkdown)
nycbikedata <- read.csv("C://Users/ep927/Documents/NewYorkCity/CleanedBikeDat
a.csv", header = TRUE)


nycbikedata$Start_Year <- strptime(nycbikedata$Start_Time, "%m/%d/%Y %H:%M")

nycbikedata$Year <- year(nycbikedata$Start_Year)
nycbikedata$Month <- month(nycbikedata$Start_Year)
nycbikedata$DayOfWeek <- wday(nycbikedata$Start_Year, label = TRUE, abbr = TR
UE)
```

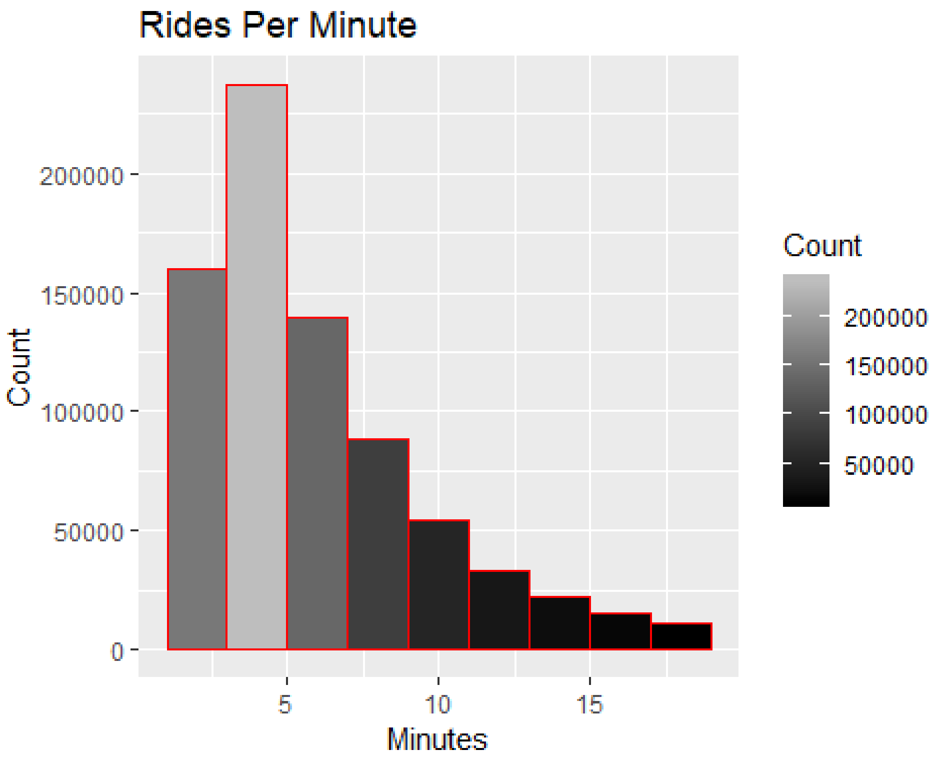
```
nycbikedata$Qtr <- quarter(nycbikedata$Start_Year)
nycbikedata$DayOfWeek <- ifelse(wday(nycbikedata$Start_Year)==1,7,wday(nycbik
edata$Start_Year)-1)
nycbikedata$Age <- (nycbikedata$Year - nycbikedata$Birth_Year)
nycbikedata$distance <-distHaversine(nycbikedata[,6:7], nycbikedata[,10:11],r
=6378137) / 1609.344
nycbikedata$RideInMin <- round(nycbikedata$Trip_Duration / 60 ,digits = 0)
```

```
str(nycbikedata)
```

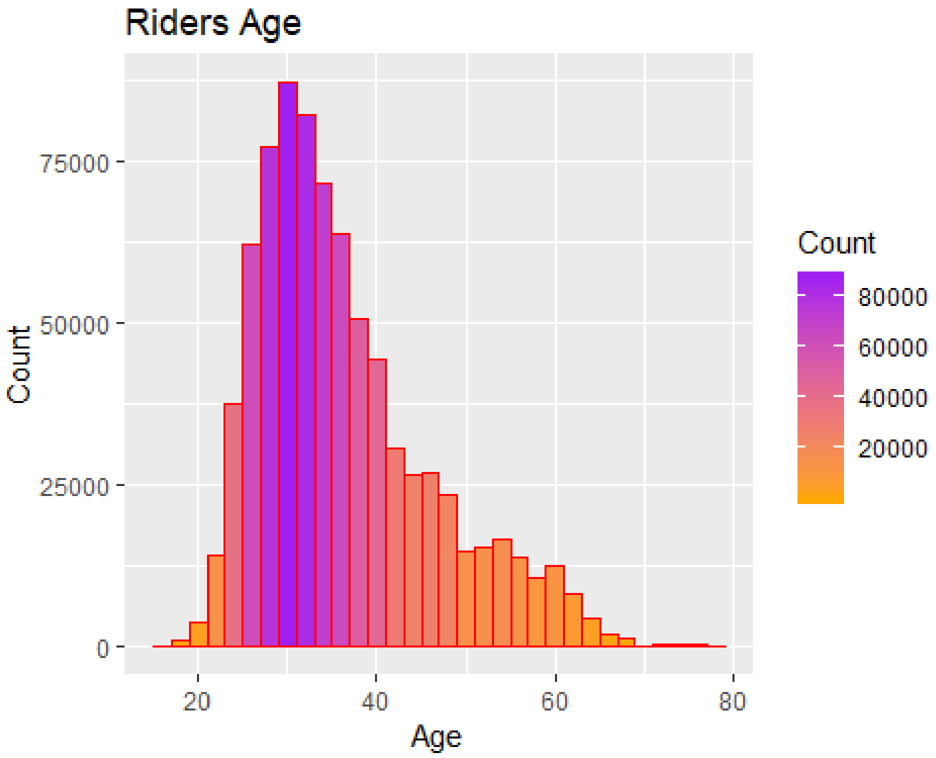
```
## 'data.frame':    802870 obs. of  23 variables:
##  $ Trip_Duration          : int  61 61 61 61 61 61 61 61 61 61 ...
##  $ Start_Time             : Factor w/ 498008 levels "1/1/2016 0:02",..: 20
729 199918 247126 308008 341766 441798 416524 425976 427339 496816 ...
##  $ Stop_Time              : Factor w/ 498759 levels "1/1/2016 0:08",..: 20
871 200195 247410 308231 342159 442373 417035 426491 427839 497548 ...
##  $ Start_Station_ID       : int  3185 3185 3185 3185 3185 3267 3270 3272 3
186 3186 ...
##  $ Start_Station_Name     : Factor w/ 61 levels "5 Corners Library",..: 11
11 11 11 11 45 33 32 24 24 ...
##  $ Start_Station_Longitude: num  -74 -74 -74 -74 -74 ...
##  $ Start_.Station_Latitude: num  40.7 40.7 40.7 40.7 40.7 ...
##  $ End_Station_ID         : int  3186 3186 3186 3186 3186 3214 3272 3270 3
185 3185 ...
##  $ End_Station_Name       : Factor w/ 183 levels "11 Ave & W 41 St",..: 76
76 76 76 76 62 86 87 38 38 ...
##  $ End_Station_Longitude  : num  -74 -74 -74 -74 -74 ...
##  $ End_Station_Latitude   : num  40.7 40.7 40.7 40.7 40.7 ...
##  $ Bike_ID                : int  24424 24491 24720 24675 24454 26176 24686
24666 26162 26253 ...
##  $ User_Type              : Factor w/ 2 levels "Customer","Subscriber": 2
2 2 2 2 2 2 2 2 2 ...
##  $ Birth_Year             : int  1983 1981 1983 1983 1981 1968 1990 1984 1
960 1992 ...
##  $ Gender                 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Start_Year             : POSIXlt, format: "2016-01-05 20:17:00" "2016-0
3-30 09:08:00" ...
##  $ Year                   : num  2016 2016 2016 2016 2016 ...
##  $ Month                  : num  1 3 5 6 7 8 8 8 8 9 ...
##  $ DayOfWeek              : num  2 3 3 1 1 1 2 7 1 5 ...
##  $ Qtr                    : int  1 1 2 2 3 3 3 3 3 3 ...
##  $ Age                    : num  33 35 33 33 35 48 26 32 56 24 ...
##  $ distance               : num  0.134 0.134 0.134 0.134 0.134 ...
##  $ RideInMin              : num  1 1 1 1 1 1 1 1 1 1 ...
```

```
summary(nycbikedata)
```

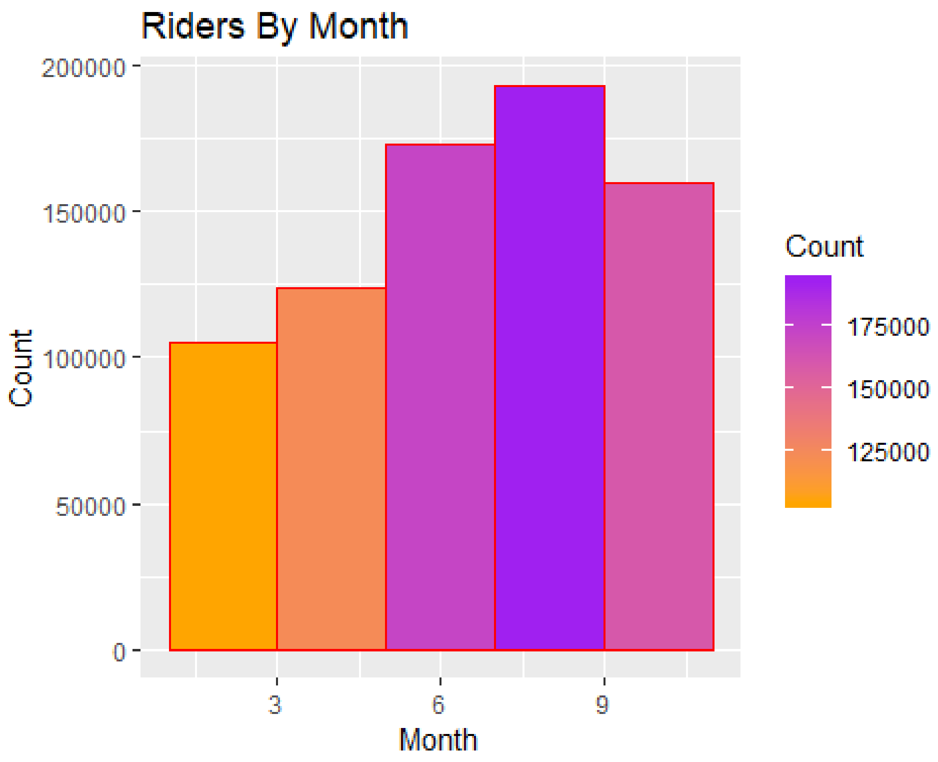
```
##  Trip_Duration               Start_Time               Stop_Time
##  Min.   :   61.0   11/2/2016 8:17 :    14   10/4/2018 8:28 :    13
##  1st Qu.:  229.0   10/4/2018 8:22 :    13   10/10/2018 8:57:    12
##  Median :  333.0   10/10/2018 8:41:    12   10/9/2018 8:46 :    12
##  Mean   :  476.7   10/12/2018 8:28:    12   6/29/2018 8:25 :    12
##  3rd Qu.:  534.0   6/8/2018 8:02  :    12   6/5/2018 8:45  :    12
##  Max.   :40968.0   8/10/2018 8:28 :    12   9/28/2016 8:37 :    12
##                    (Other)        :802795   (Other)        :802797
##  Start_Station_ID      Start_Station_Name Start_Station_Longitude
##  Min.   :3183     Grove St PATH : 99088   Min.   :-74.10
##  1st Qu.:3187     Hamilton Park : 50761   1st Qu.:-74.05
##  Median :3203     Exchange Place: 50260   Median :-74.04
##  Mean   :3235     Sip Ave       : 48254   Mean   :-74.05
##  3rd Qu.:3267     Newport PATH  : 36234   3rd Qu.:-74.04
##  Max.   :3694     Newark Ave    : 25244   Max.   :-74.03
##                   (Other)       :493029
##  Start_.Station_Latitude End_Station_ID       End_Station_Name
##  Min.   :40.69           Min.   :  79   Grove St PATH :127115
##  1st Qu.:40.72           1st Qu.:3186   Exchange Place: 62630
##  Median :40.72           Median :3202   Hamilton Park : 47848
##  Mean   :40.72           Mean   :3230   Sip Ave       : 43900
##  3rd Qu.:40.73           3rd Qu.:3220   Newport PATH  : 38593
##  Max.   :40.75           Max.   :3694   Newark Ave    : 23980
##                                         (Other)       :458804
##  End_Station_Longitude End_Station_Latitude    Bike_ID
##  Min.   :-74.10        Min.   :40.68        Min.   :14529
##  1st Qu.:-74.05        1st Qu.:40.72        1st Qu.:26157
##  Median :-74.04        Median :40.72        Median :26314
##  Mean   :-74.05        Mean   :40.72        Mean   :27852
##  3rd Qu.:-74.04        3rd Qu.:40.73        3rd Qu.:29587
##  Max.   :-73.95        Max.   :40.81        Max.   :35009
##
##       User_Type        Birth_Year       Gender
##  Customer  :  9959   Min.   :1940   Min.   :1.000
##  Subscriber:792911   1st Qu.:1975   1st Qu.:1.000
##                      Median :1983   Median :1.000
##                      Mean   :1980   Mean   :1.224
##                      3rd Qu.:1988   3rd Qu.:1.000
##                      Max.   :2002   Max.   :2.000
##
##    Start_Year                       Year          Month
##  Min.   :2016-01-01 00:02:00   Min.   :2016   Min.   : 1.000
##  1st Qu.:2016-11-23 11:37:00   1st Qu.:2016   1st Qu.: 5.000
##  Median :2017-09-19 12:18:30   Median :2017   Median : 7.000
##  Mean   :2017-09-08 10:29:15   Mean   :2017   Mean   : 7.209
##  3rd Qu.:2018-06-22 20:08:30   3rd Qu.:2018   3rd Qu.:10.000
##  Max.   :2018-12-31 23:25:00   Max.   :2018   Max.   :12.000
##
##    DayOfWeek          Qtr             Age           distance
##  Min.   :1.000   Min.   :1.000   Min.   :16.00   Min.   :0.08678
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:30.00   1st Qu.:0.38596
##  Median :4.000   Median :3.000   Median :35.00   Median :0.55040
##  Mean   :3.685   Mean   :2.737   Mean   :36.86   Mean   :0.65634
##  3rd Qu.:5.000   3rd Qu.:4.000   3rd Qu.:42.00   3rd Qu.:0.84295
##  Max.   :7.000   Max.   :4.000   Max.   :78.00   Max.   :7.54271
##
##    RideInMin
##  Min.   :  1.000
##  1st Qu.:  4.000
##  Median :  6.000
##  Mean   :  7.945
##  3rd Qu.:  9.000
##  Max.   :683.000
##

# Graphs for some of the datat

ggplot(data=nycbikedata, aes(nycbikedata$RideInMin)) +
  labs(title = "Rides Per Minute") +
  labs(x="Minutes", y="Count") +
  geom_histogram(breaks=seq(1,20, by =2),
                 col="red",
                 aes(fill=..count..)) +
                 scale_fill_gradient("Count", low = "black", high = "gray")
```



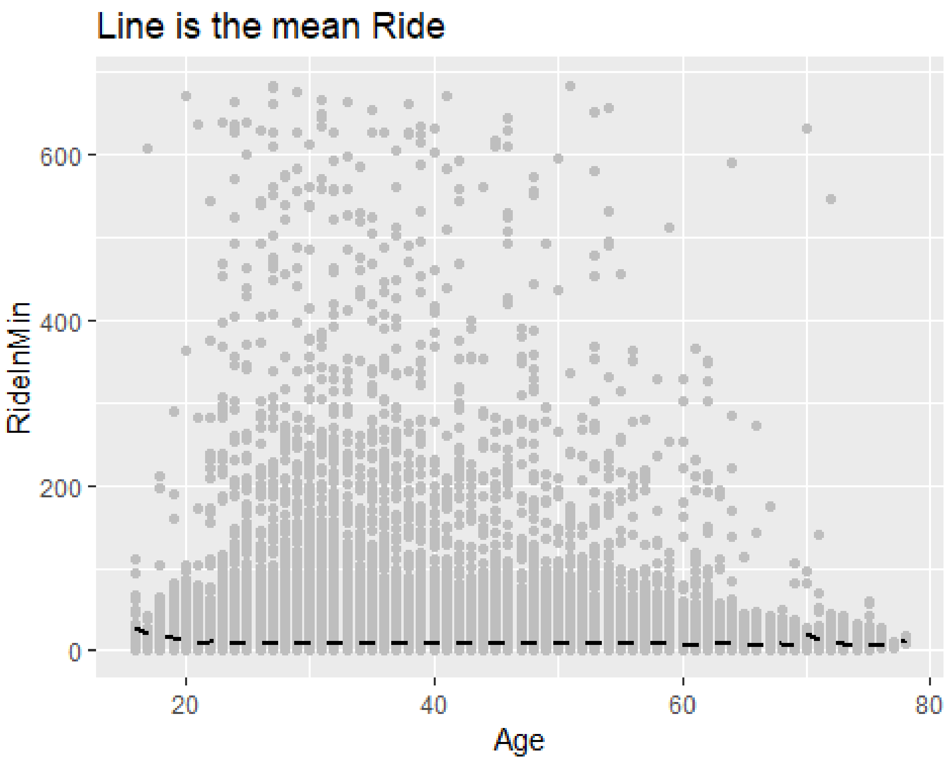
```
ggplot(data=nycbikedata, aes(nycbikedata$Age)) +
  labs(title = "Riders Age") +
  labs(x="Age", y="Count") +
  geom_histogram(breaks=seq(15,80, by =2),
                 col="red",
                 aes(fill=..count..)) +
                 scale_fill_gradient("Count", low = "orange", high = "purple"
)
```



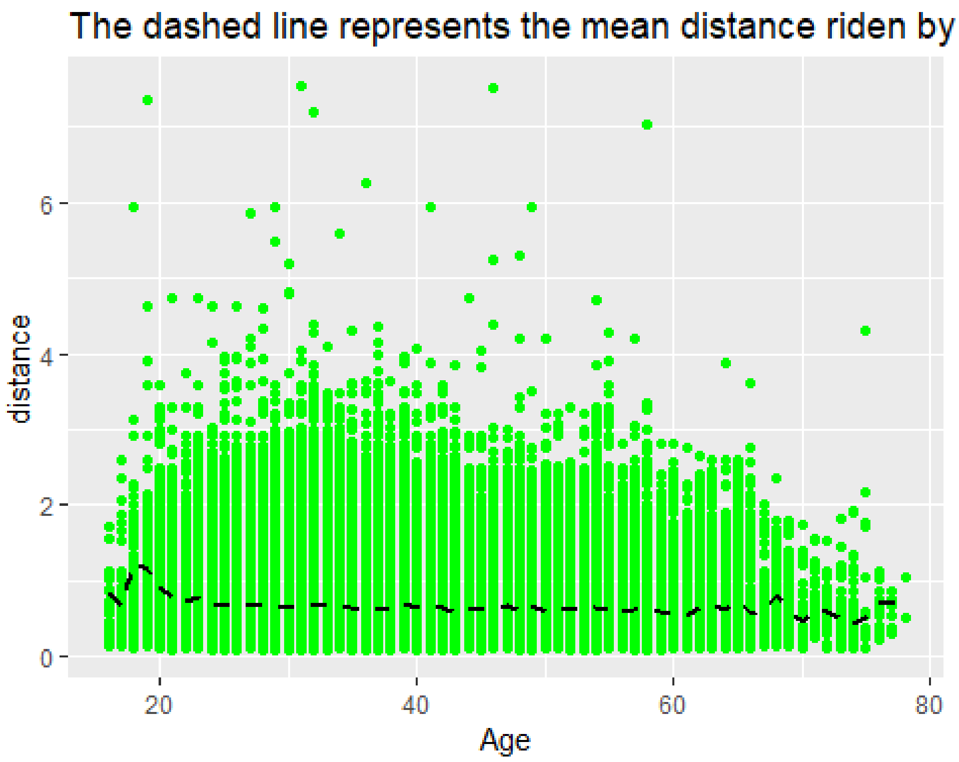
```
ggplot(data=nycbikedata, aes(nycbikedata$Month)) +
  labs(title = "Riders By Month") +
  labs(x="Month", y="Count") +
  geom_histogram(breaks=seq(1,12,by = 2),
                 col="red",
                 aes(fill=..count..)) +
                 scale_fill_gradient("Count", low = "orange", high = "purple"
)
```



```
ggplot(nycbikedata, aes(Age, RideInMin)) +
  labs(title = "Line is the mean Ride") +
  geom_point(color = "grey") +
  stat_summary(fun.y = "mean", geom = "line", size = 1, linetype = "dashed")
```



```r
ggplot(nycbikedata, aes(Age, distance)) +
  geom_point(color = "green") +
  labs(title = "The dashed line represents the mean distance riden by Age") +
  stat_summary(fun.y = "mean", geom = "line", size = 1, linetype = "dashed")
```



```
#Selecting the fields to be used in predicting Gender using RandomForset.  Si
nce Gender is a int I need to convert it to a Factor

df <- select(nycbikedata, Age, Gender, Year, Month,DayOfWeek,distance,RideInM
in, Start_Station_ID, End_Station_ID)

small.sample <- stratified(df, "Gender", size=10000)

small.sample$Gender <- as.factor(small.sample$Gender)

table(small.sample$Gender)

## 
##     1     2 
## 10000 10000

str(small.sample)

## Classes 'data.table' and 'data.frame':   20000 obs. of  9 variables:
##  $ Age             : num  46 34 32 29 31 42 29 37 47 32 ...
##  $ Gender          : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Year            : num  2017 2016 2018 2018 2017 ...
##  $ Month           : num  4 9 11 7 2 8 11 1 7 10 ...
##  $ DayOfWeek       : num  3 2 2 2 1 7 3 4 3 2 ...
##  $ distance        : num  0.424 0.386 1.01 1.898 0.583 ...
##  $ RideInMin       : num  4 2 8 11 6 44 4 7 6 5 ...
##  $ Start_Station_ID: int  3195 3279 3214 3194 3187 3211 3194 3183 3194 327
```

```
0 ...
##  $ End_Station_ID  : int  3194 3186 3202 3183 3214 3213 3195 3276 3195 318
5 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```r
#Data Partition
#ind = Independent Variable. The data will be split 70/30
#rf = Random Forest

set.seed(123)
ind <- sample(2, nrow(small.sample), replace=TRUE, prob = c(0.7, 0.3))
train <- small.sample[ind==1,]
test <- small.sample[ind==2,]

#Random Forset - Prior to Tuning the model

set.seed(111)
rf <-randomForest(Gender~., data = train)
print(rf)
```

```
##
## Call:
##  randomForest(formula = Gender ~ ., data = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 31.8%
## Confusion matrix:
##      1    2 class.error
## 1 4730 2318   0.3288876
## 2 2159 4873   0.3070250
```

```r
attributes(rf)
```

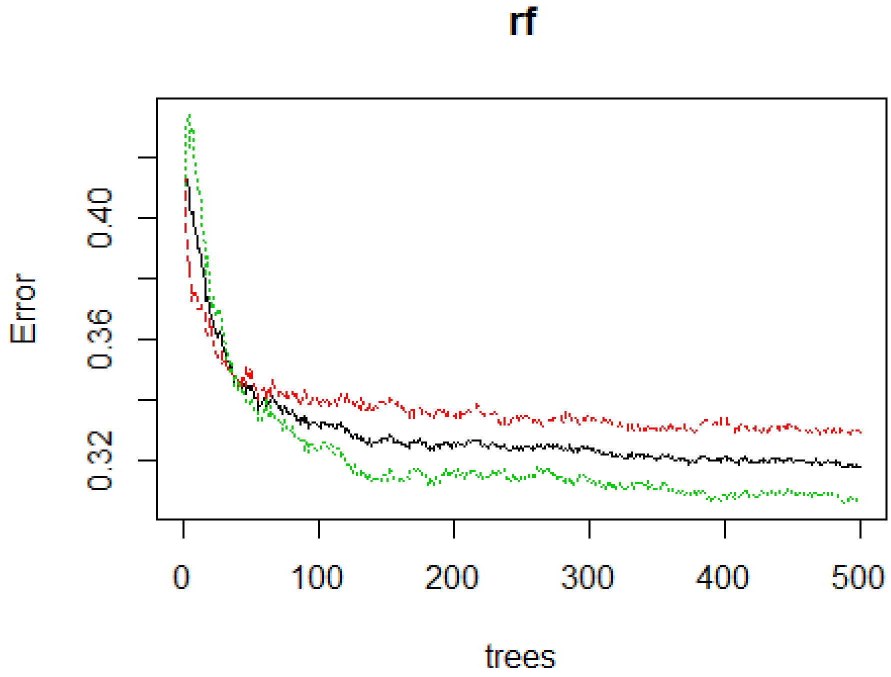
```
## $names
##  [1] "call"            "type"            "predicted"
##  [4] "err.rate"        "confusion"       "votes"
##  [7] "oob.times"       "classes"         "importance"
## [10] "importanceSD"    "localImportance" "proximity"
## [13] "ntree"           "mtry"            "forest"
## [16] "y"               "test"            "inbag"
## [19] "terms"
##
## $class
## [1] "randomForest.formula" "randomForest"
```

```r
#Prediction and Confusion Matrix - Train Data
#Pred1 = Predication 1
pred1 <- predict(rf, train)
confusionMatrix(pred1,train$Gender)
```

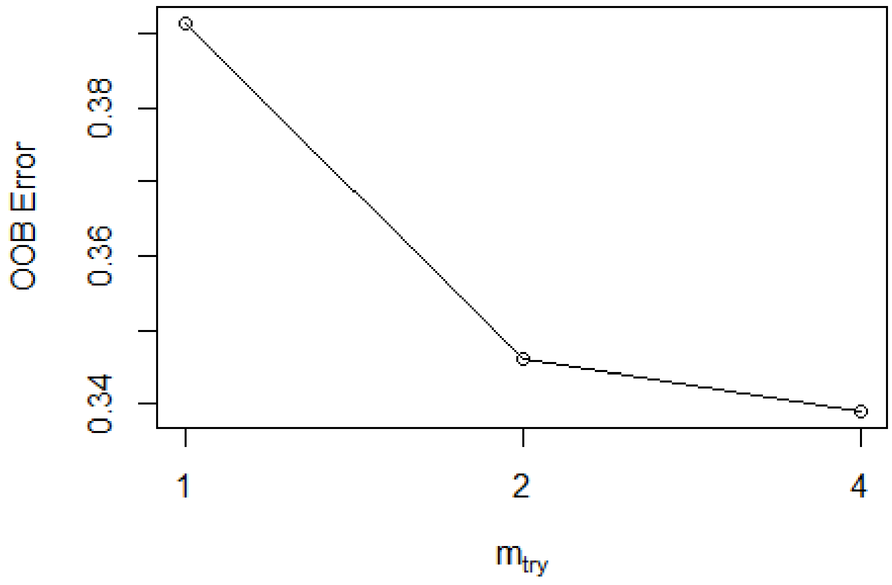
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    1    2
##          1 7041   36
##          2    7 6996
##                                           
##                Accuracy : 0.9969          
##                  95% CI : (0.9959, 0.9978)
##     No Information Rate : 0.5006          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9939          
##  Mcnemar's Test P-Value : 1.955e-05       
##                                           
##             Sensitivity : 0.9990          
##             Specificity : 0.9949          
##          Pos Pred Value : 0.9949          
##          Neg Pred Value : 0.9990          
##              Prevalence : 0.5006          
##          Detection Rate : 0.5001          
##    Detection Prevalence : 0.5026          
##       Balanced Accuracy : 0.9969          
##                                           
##        'Positive' Class : 1               
## 
```

```r
#Prediction and Confusion with Matrix - Test Data
#Pred2 = Prediction 2

pred2 <- predict(rf, test)
confusionMatrix(pred2, test$Gender)
```

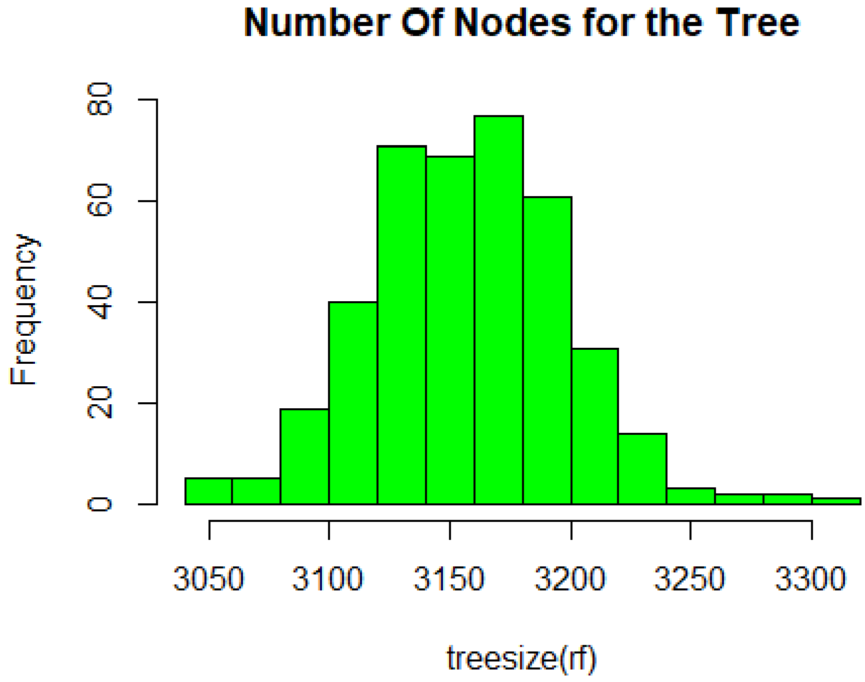
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    1    2
##          1 1989  932
##          2  963 2036
##                                           
##                Accuracy : 0.6799          
##                  95% CI : (0.6678, 0.6918)
##     No Information Rate : 0.5014          
##     P-Value [Acc > NIR] : <2e-16          
##                                           
##                   Kappa : 0.3598          
##  Mcnemar's Test P-Value : 0.4907          
##                                           
##             Sensitivity : 0.6738          
##             Specificity : 0.6860          
##          Pos Pred Value : 0.6809
##          Neg Pred Value : 0.6789
##              Prevalence : 0.4986
##          Detection Rate : 0.3360
##    Detection Prevalence : 0.4934
##       Balanced Accuracy : 0.6799
##
##        'Positive' Class : 1
##

#Error Rate

plot(rf)
```



```
#Tune my try
#train <- as.data.frame(train) will remove Error in randomForest.default(x, y
, mtry = mtryStart, ntree = ntreeTry, : length of response must be the same a
s predictors

train <- as.data.frame(train)
t <- tuneRF(train[,-2], train[,2],
       stepFactor = 0.5,
       plot=TRUE,
       improve = 0.05)

## mtry = 2  OOB error = 34.62%
## Searching left ...
## mtry = 4     OOB error = 33.9%
## 0.0207222 0.05
## Searching right ...
## mtry = 1     OOB error = 39.14%
## -0.1306935 0.05
```



```
#Tune my model
# using the information from mtry I used that information to tune my model.

rf <-randomForest(Gender~., data = train,
                  ntree = 400,
                  mtry = 4,
                  importance = TRUE,
                   proximity = TRUE)


print(rf)

##
## Call:
##  randomForest(formula = Gender ~ ., data = train, ntree = 400,      mtry =
4, importance = TRUE, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 400
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 31.78%
## Confusion matrix:
##       1    2 class.error
## 1 4659 2389   0.3389614
## 2 2085 4947   0.2965017

rf$confusion

##       1    2 class.error
## 1 4659 2389   0.3389614
## 2 2085 4947   0.2965017

hist(treesize(rf),
     main = "Number Of Nodes for the Tree",
     col = "green")
```

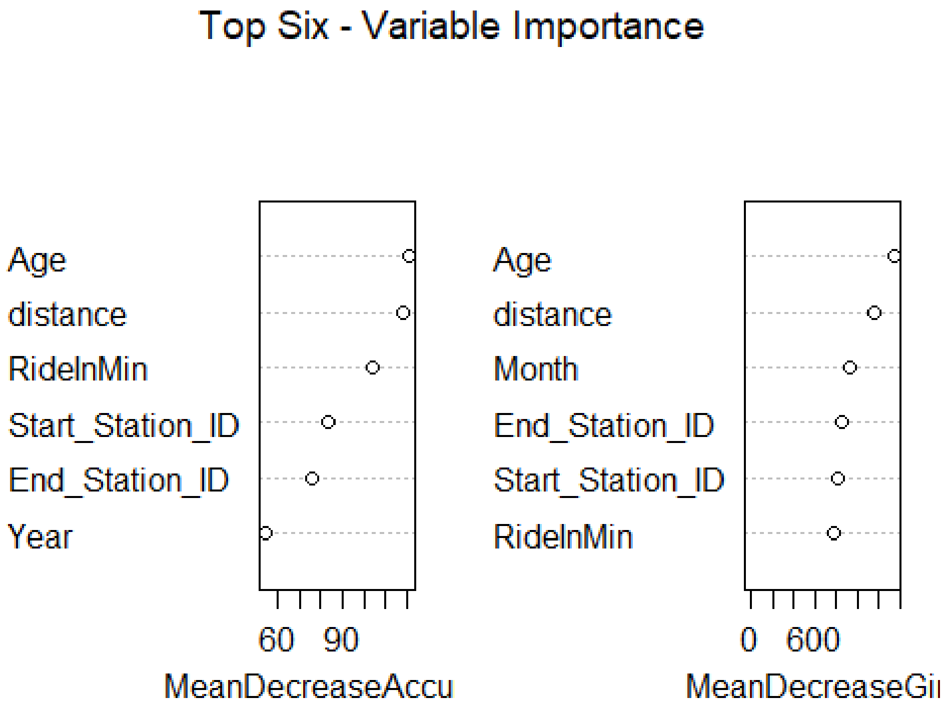


```
#After modeled tuned

pred1 <- predict(rf, train)
confusionMatrix(pred1,train$Gender)

## Confusion Matrix and Statistics
##
##            Reference
## Prediction    1    2
##          1 7048   12
##          2    0 7020
##
##                 Accuracy : 0.9991
##                   95% CI : (0.9985, 0.9996)
##      No Information Rate : 0.5006
##      P-Value [Acc > NIR] : < 2.2e-16
## 
##                    Kappa : 0.9983
##   Mcnemar's Test P-Value : 0.001496
## 
##              Sensitivity : 1.0000
##              Specificity : 0.9983
##           Pos Pred Value : 0.9983
##           Neg Pred Value : 1.0000
##               Prevalence : 0.5006
##           Detection Rate : 0.5006
##     Detection Prevalence : 0.5014
##        Balanced Accuracy : 0.9991
## 
##         'Positive' Class : 1
## 
```

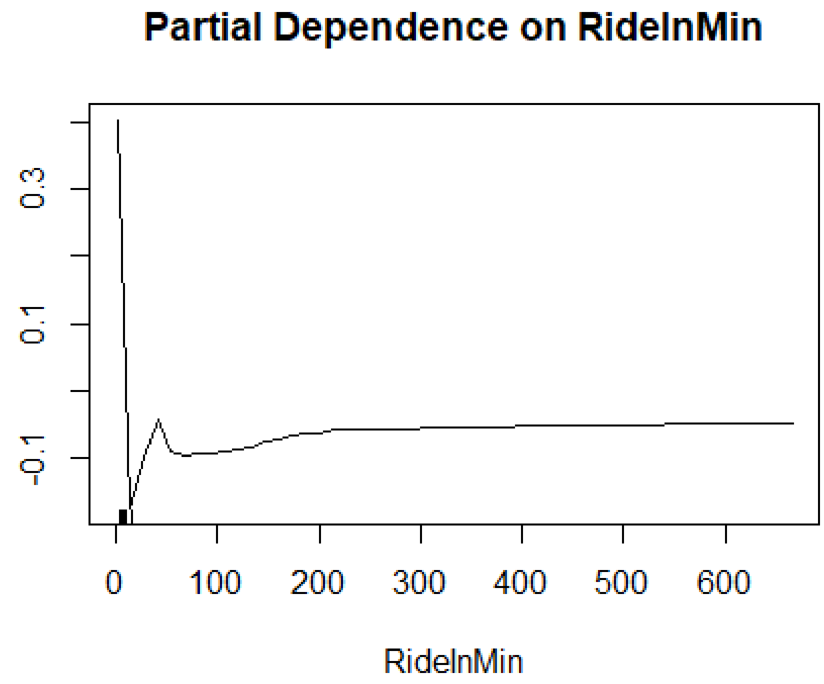
```
#After Modeled tuned

pred2 <- predict(rf, test)
confusionMatrix(pred2, test$Gender)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    1    2
##          1 1981  848
##          2  971 2120
##                                           
##                 Accuracy : 0.6927
##                   95% CI : (0.6808, 0.7045)
##      No Information Rate : 0.5014
##      P-Value [Acc > NIR] : < 2e-16
## 
##                    Kappa : 0.3854
##   Mcnemar's Test P-Value : 0.00423
## 
##              Sensitivity : 0.6711
##              Specificity : 0.7143
##           Pos Pred Value : 0.7002
##           Neg Pred Value : 0.6859
##               Prevalence : 0.4986
##           Detection Rate : 0.3346
##     Detection Prevalence : 0.4779
##        Balanced Accuracy : 0.6927
## 
##         'Positive' Class : 1
## 
```

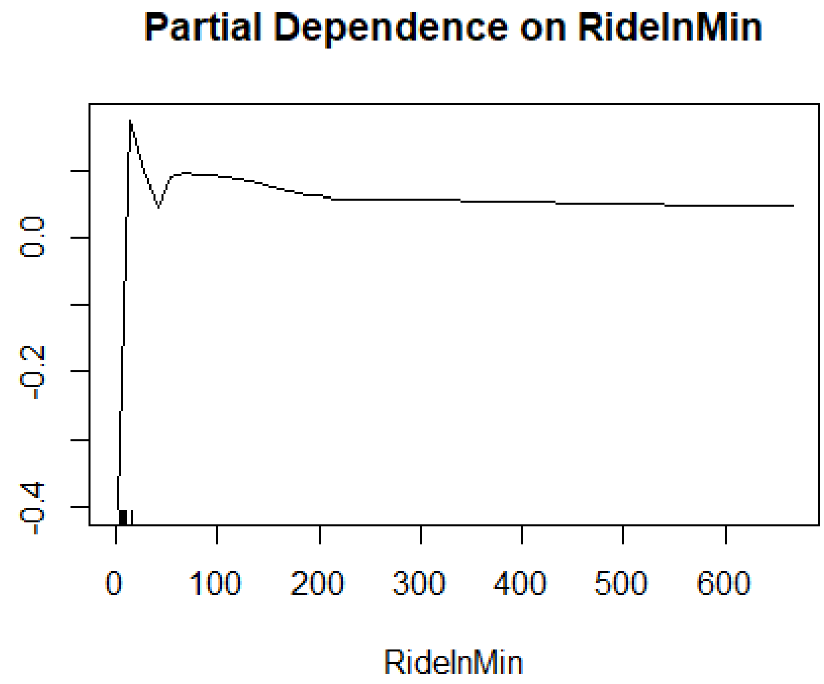
```
#Importance of Variables

varImpPlot(rf,
           sort = TRUE,
           n.var = 6,
           main = "Top Six - Variable Importance")
```



```
varUsed(rf)

## [1] 231706  84733 196827 159184 179307 128370 138755 143979

partialPlot(rf, train, RideInMin, "1")
```



```
partialPlot(rf,train,RideInMin,"2")
```


Partial Dependence on RideInMin
RideInMin