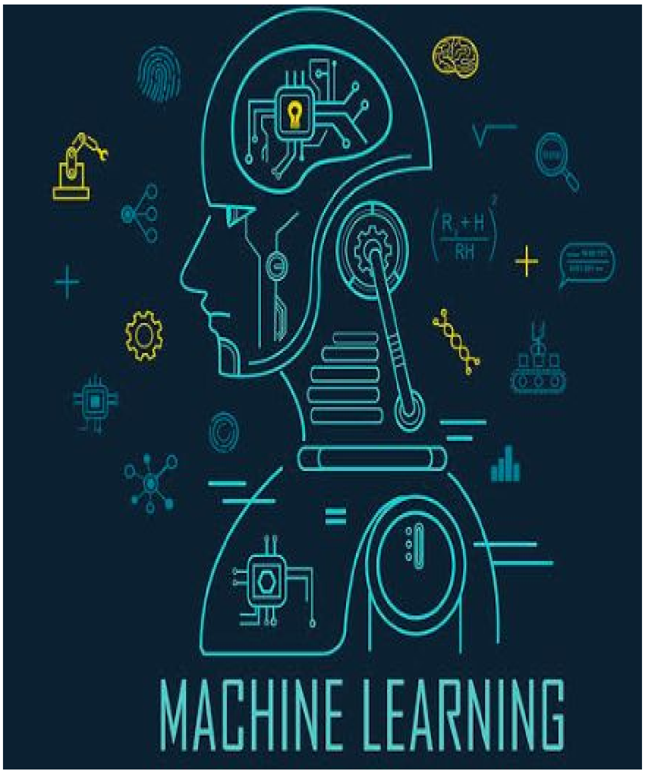


Baijayanti Chakraborty
Project 5 : Machine Learning
PGP-BABI - Online

1. Project Objective

This project requires you to understand what mode of transport employees prefers to commute to their office. The Cars.csv provided includes employee information about their mode of transport as well as their personal and professional details like age, salary, work exp. We need to predict whether or not an employee will use Car as a mode of transport. Also, which variables are a significant predictor behind this decision.

The different steps taken to reach a conclusion :

EDA

- Perform an EDA on the data
- Illustrate the insights based on EDA
- Check for Multicollinearity - Plot the graph based on Multicollinearity & treat it.

Data Preparation

Prepare the data for analysis (SMOTE)

Modeling

- Create multiple models and explore how each model perform using appropriate model performance metrics
  - KNN
  - Naive Bayes (is it applicable here? comment and if it is not applicable, how can you build an NB model in this case?)
  - Logistic Regression
- Apply both bagging and boosting modeling procedures to create 2 models and compare its accuracy with the best model of the above step.

Actionable Insights & Recommendations

- Summarize your findings from the exercise in a concise yet actionable note

2.Exploratory Data Analysis(EDA)

2.1 Include the needed libraries for the analysis

```
#include the needed libraries
library(DataExplorer)
library(psych)
library(ggplot2)

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha

library(dplyr)

##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(caret)

## Loading required package: lattice

library(car)

## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:psych':
##
##     logit

library(DMwR)

## Loading required package: grid

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

library(corrplot)

## corrplot 0.84 loaded

#library(MVN)
library(plyr)

##
------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first,
then dplyr:
## library(plyr); library(dplyr)
```

```
##
------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following object is masked from 'package:DMwR':
##
##     join

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```r
library(e1071)
library(mlogit)
```

```
## Loading required package: Formula

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: lmtest
```

```r
library(ggcorrplot)
library(RColorBrewer)
library(VIM)
```

```
## Loading required package: colorspace

## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## VIM is ready to use.
##  Since version 4.0.0 the GUI is in its own package VIMGUI.
##
##            Please use the package to use the new (and old) GUI.

## Suggestions and bug-reports can be submitted at:
https://github.com/alexkowa/VIM/issues
```

```
##
## Attaching package: 'VIM'

## The following object is masked from 'package:DMwR':
##
##     kNN

## The following object is masked from 'package:datasets':
##
##     sleep

library(class)
library(descr)
library(ipred)
library(rpart)
library(gbm)

## Loaded gbm 2.1.5

library(xgboost)

##
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':
##
##     slice

library(caret)
library(ipred)
library(plyr)
library(rpart)
library(knitr)
```

2.2 Environment setup

```
#clear the environment
rm(list = ls())

#read the dataset
cars = read.csv("Cars.csv")
```

2.3 Data Understanding and Insights gained

2.3.1 Data Understanding

```
#basic statistics of the dataset
summary(cars)
```

```
##       Age           Gender       Engineer           MBA        
##  Min.   :18.00   Female:121   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:25.00   Male  :297   1st Qu.:0.2500   1st Qu.:0.0000  
##  Median :27.00                Median :1.0000   Median :0.0000  
##  Mean   :27.33                Mean   :0.7488   Mean   :0.2614  
##  3rd Qu.:29.00                3rd Qu.:1.0000   3rd Qu.:1.0000  
##  Max.   :43.00                Max.   :1.0000   Max.   :1.0000  
##                                                NA's   :1       
##     Work.Exp          Salary          Distance        license      
##  Min.   : 0.000   Min.   : 6.500   Min.   : 3.20   Min.   :0.0000  
##  1st Qu.: 3.000   1st Qu.: 9.625   1st Qu.: 8.60   1st Qu.:0.0000  
##  Median : 5.000   Median :13.000   Median :10.90   Median :0.0000  
##  Mean   : 5.873   Mean   :15.418   Mean   :11.29   Mean   :0.2033  
##  3rd Qu.: 8.000   3rd Qu.:14.900   3rd Qu.:13.57   3rd Qu.:0.0000  
##  Max.   :24.000   Max.   :57.000   Max.   :23.40   Max.   :1.0000  
##                                                                    
##             Transport  
##  2Wheeler        : 83  
##  Car             : 35  
##  Public Transport:300  
##                        
##                        
##                        
##                        
```

```
str(cars)
```

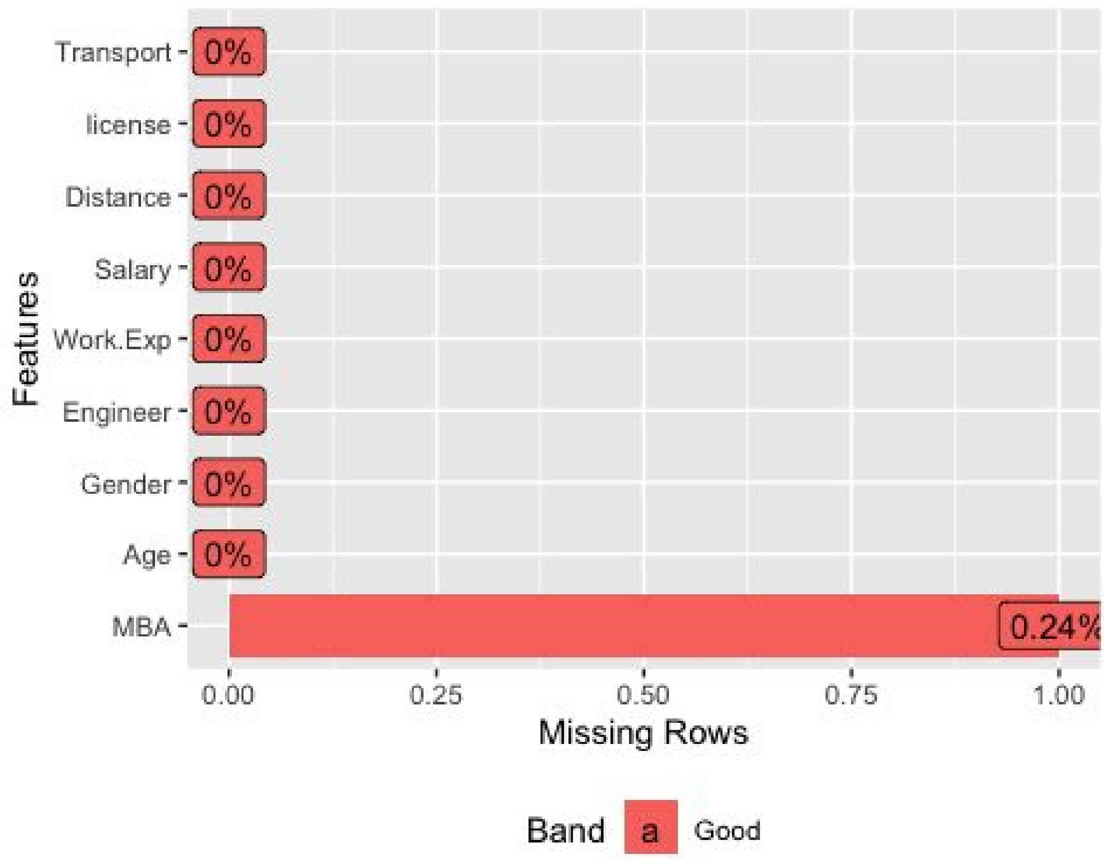
```
## 'data.frame':    418 obs. of  9 variables:
##  $ Age      : int  28 24 27 25 25 21 23 23 24 28 ...
##  $ Gender   : Factor w/ 2 levels "Female","Male": 2 2 1 2 1 2 2 2 2 2 
...
##  $ Engineer : int  1 1 1 0 0 0 1 0 1 1 ...
##  $ MBA      : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ Work.Exp : int  5 6 9 1 3 3 3 0 4 6 ...
##  $ Salary   : num  14.4 10.6 15.5 7.6 9.6 9.5 11.7 6.5 8.5 13.7 ...
##  $ Distance : num  5.1 6.1 6.1 6.3 6.7 7.1 7.2 7.3 7.5 7.5 ...
##  $ license  : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Transport: Factor w/ 3 levels "2Wheeler","Car",..: 1 1 1 1 1 1 1 1 1 
1 ...
```

```
describe(cars)
```

```
##             vars   n  mean   sd median trimmed  mad  min  max range 
skew
```

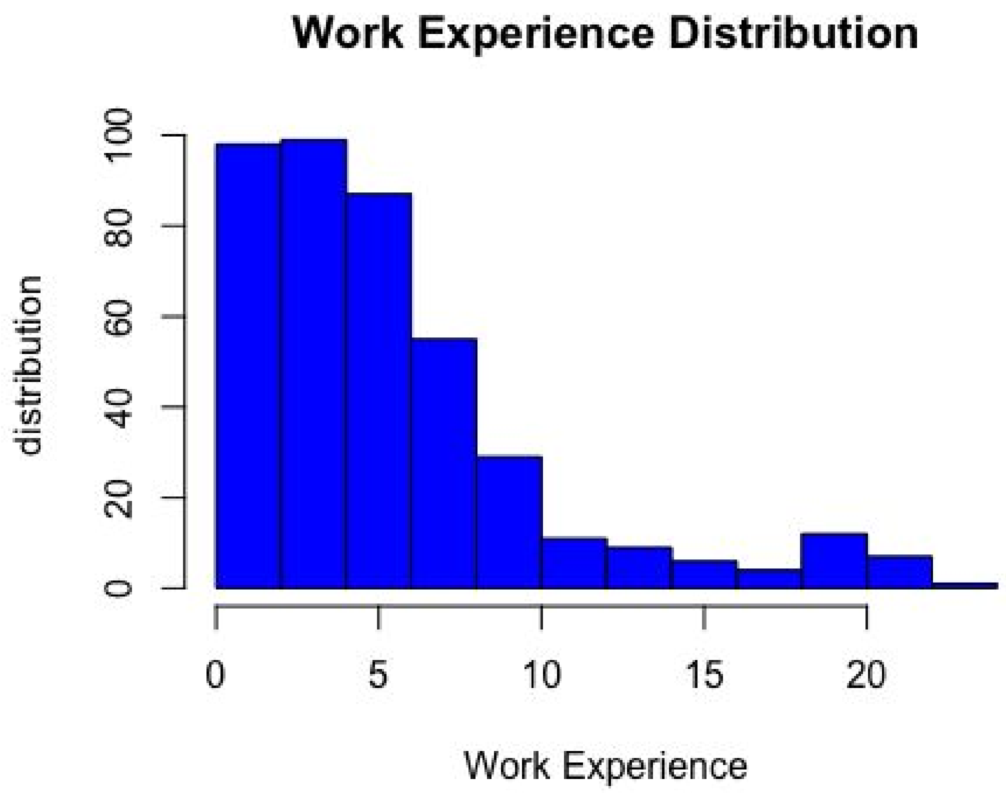
```
## Age            1 418 27.33 4.15   27.0   26.89 2.97 18.0 43.0   25.0
1.09
## Gender*        2 418  1.71 0.45    2.0    1.76 0.00  1.0  2.0    1.0
-0.93
## Engineer       3 418  0.75 0.43    1.0    0.81 0.00  0.0  1.0    1.0
-1.14
## MBA            4 417  0.26 0.44    0.0    0.20 0.00  0.0  1.0    1.0
1.08
## Work.Exp       5 418  5.87 4.82    5.0    5.12 2.97  0.0 24.0   24.0
1.52
## Salary         6 418 15.42 9.66   13.0   13.22 4.15  6.5 57.0   50.5
2.28
## Distance       7 418 11.29 3.70   10.9   11.08 3.56  3.2 23.4   20.2
0.55
## license        8 418  0.20 0.40    0.0    0.13 0.00  0.0  1.0    1.0
1.47
## Transport*     9 418  2.52 0.81    3.0    2.65 0.00  1.0  3.0    2.0
-1.20
##            kurtosis   se
## Age            1.67 0.20
## Gender*       -1.15 0.02
## Engineer      -0.69 0.02
## MBA           -0.83 0.02
## Work.Exp       2.29 0.24
## Salary         4.82 0.47
## Distance       0.05 0.18
## license        0.16 0.02
## Transport*    -0.38 0.04
```

```
#checking for the null values
plot_missing(cars)
```

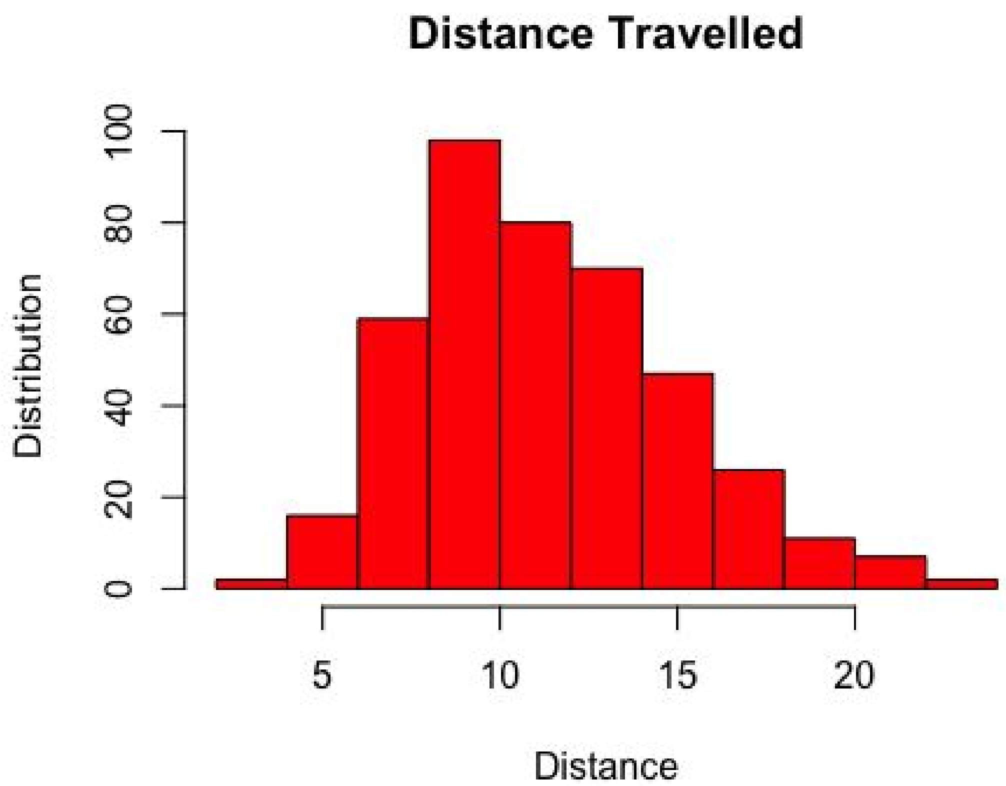



```r
#creating sample visuals for the dataset
hist(cars$Salary,main = "Salary distribution" , xlab = "Salary" , ylab =
"Distribution" , col = "deeppink")
```

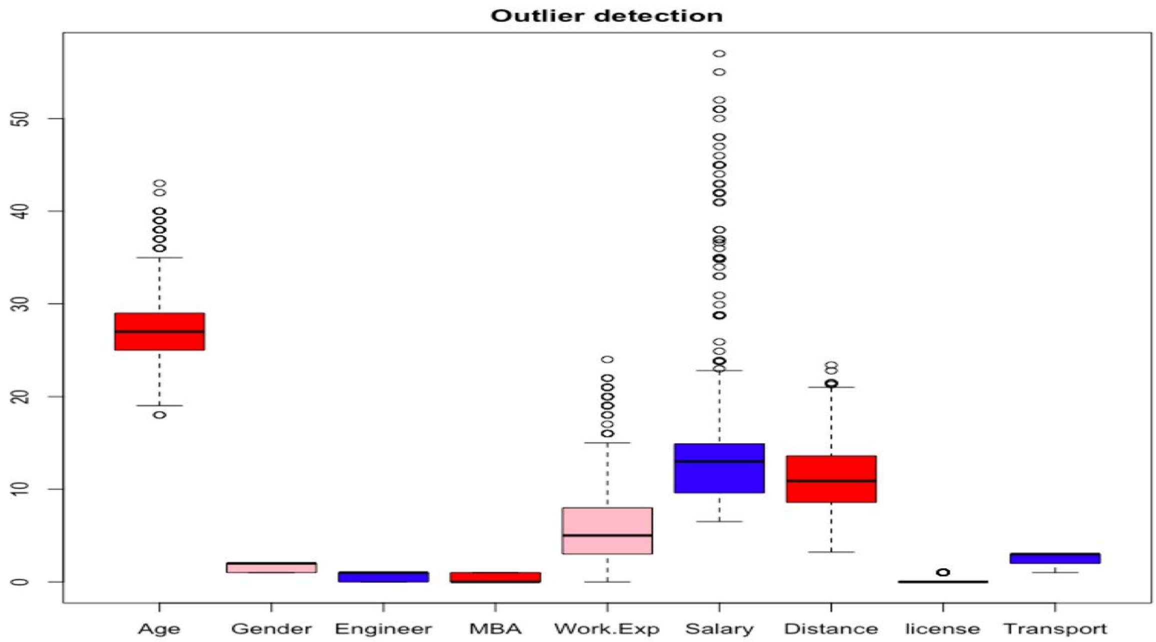
```
hist(cars$Work.Exp,main = "Work Experience Distribution" , xlab = "Work
Experience" , ylab = "distribution" , col = "blue")
```



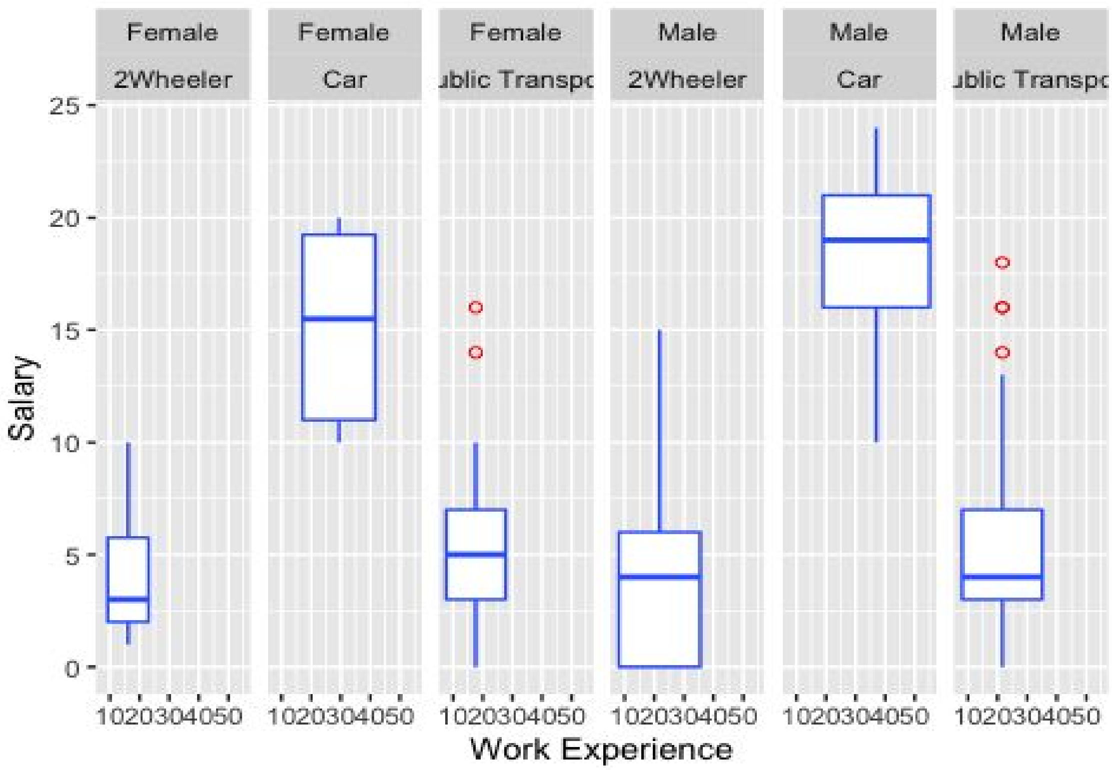

```
hist(cars$Distance , main = "Distance Travelled" , xlab = "Distance" ,
ylab = "Distribution" , col = "red")
```



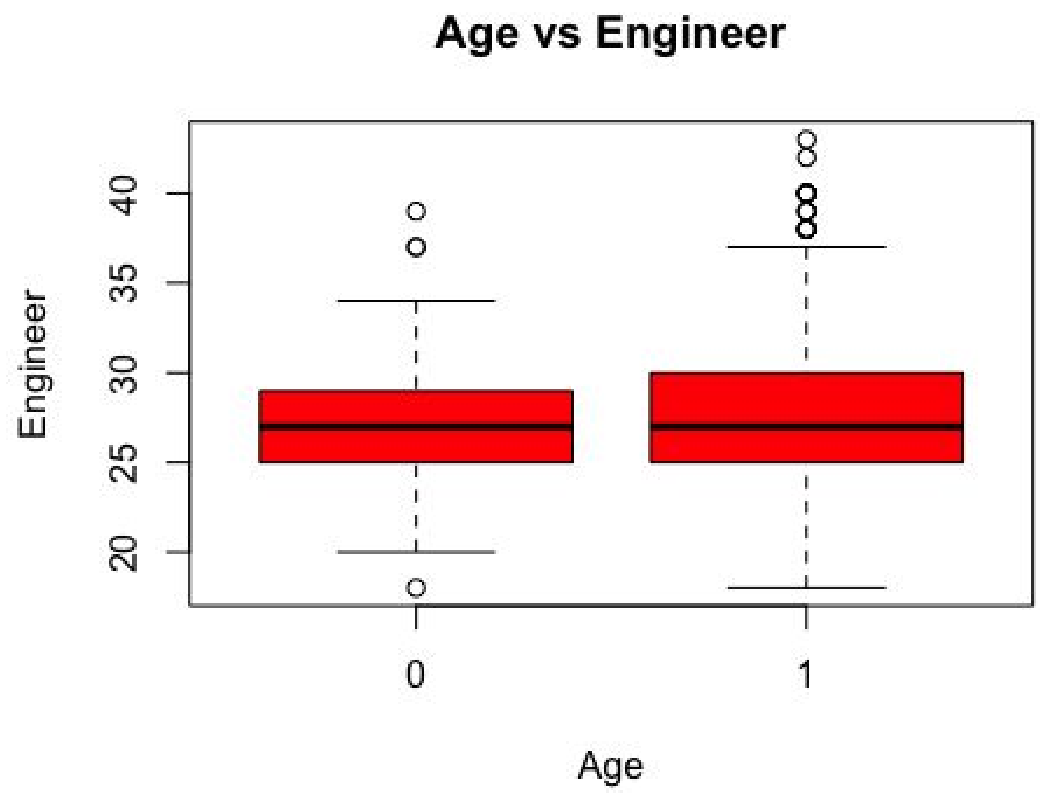
```r
boxplot(cars , col = c("red","pink","blue"),main = "Outlier detection")
```



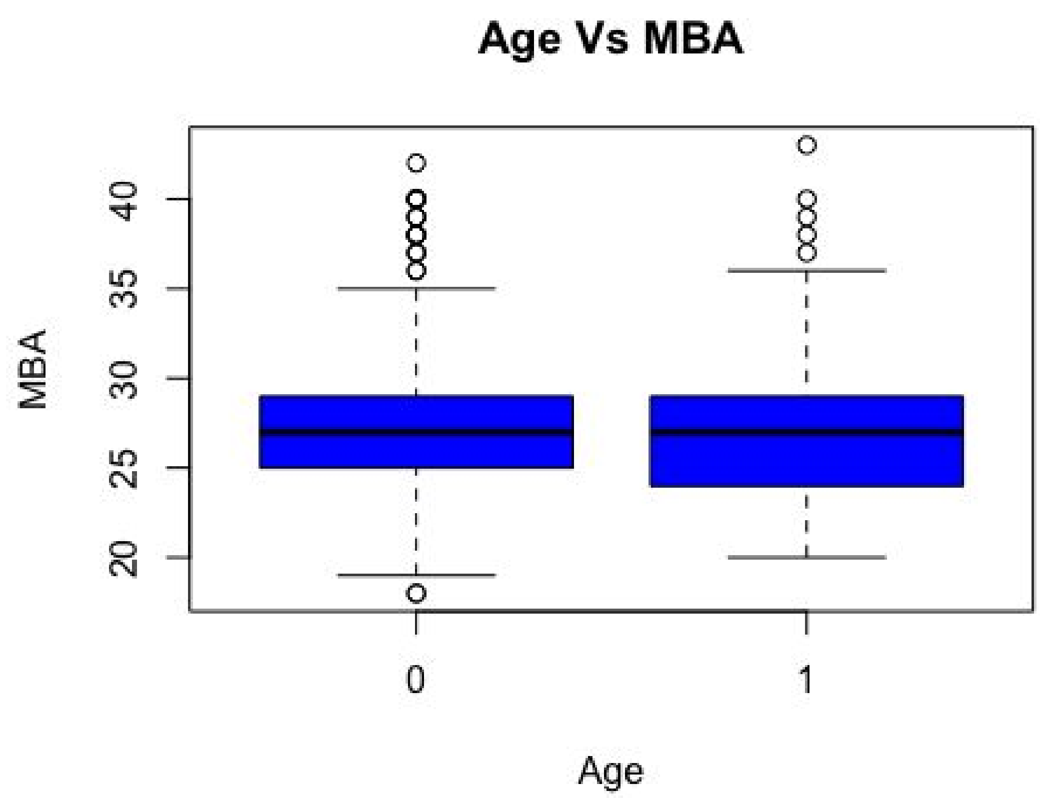

```r
work_salary_dist = ggplot(cars, aes(x = cars$Salary, y = cars$Work.Exp)) +
 facet_grid(~ cars$Gender + cars$Transport)+
 geom_boxplot(na.rm = TRUE, colour = "#3366FF",outlier.colour = "red",
              outlier.shape = 1) +
 labs(x = "Work Experience", y = "Salary") +
 scale_x_continuous() +
 scale_y_continuous() +
 theme(legend.position="bottom", legend.direction="horizontal")

work_salary_dist

## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```



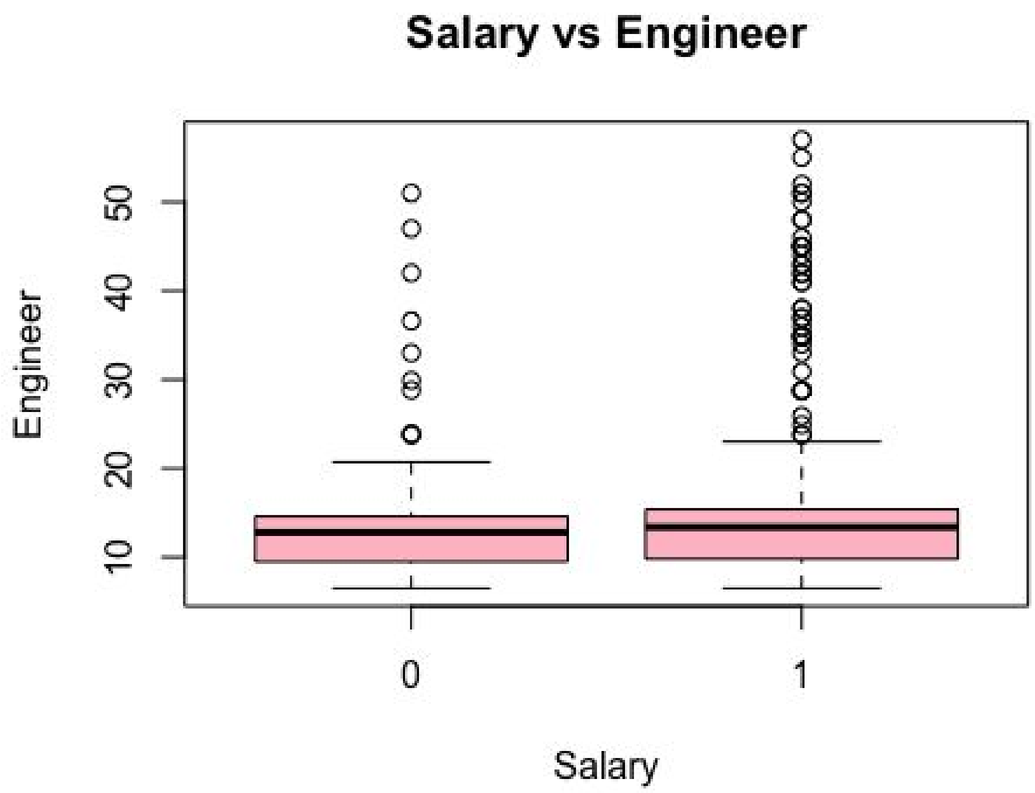
```
boxplot(cars$Age ~cars$Engineer, main = "Age vs Engineer" , xlab = "Age" ,
ylab = "Engineer",col = "red")
```




```
boxplot(cars$Age ~cars$MBA, main ="Age Vs MBA" ,xlab = "Age" , ylab =
"MBA",col = "blue")
```

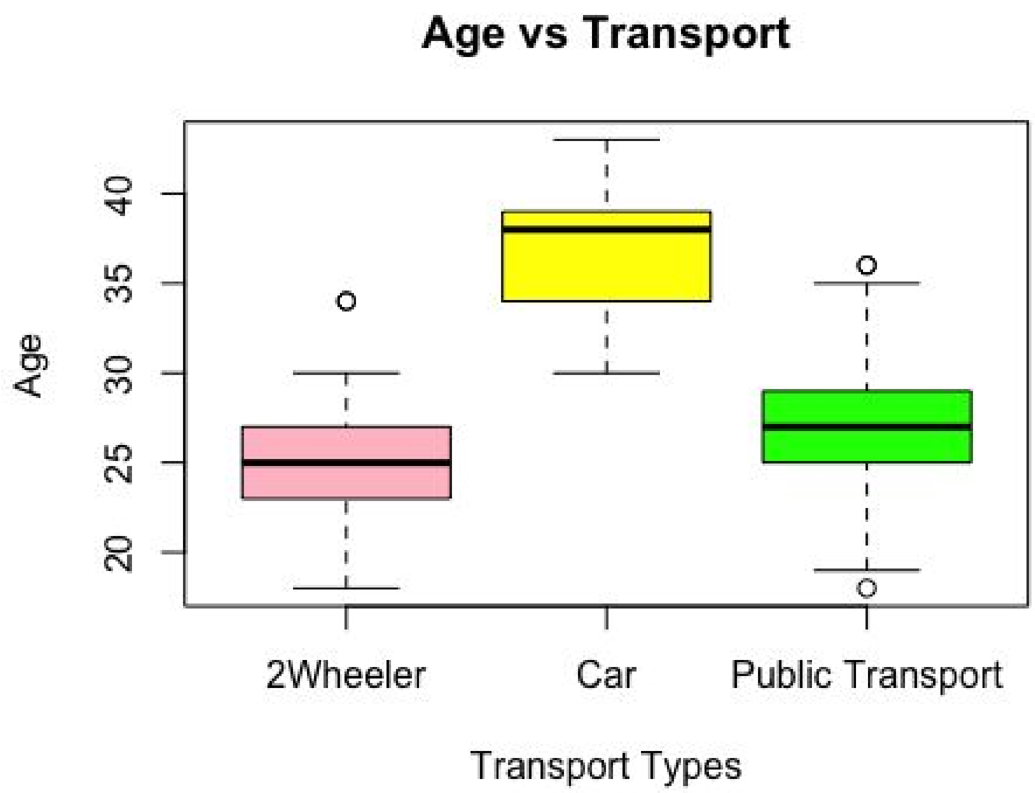


```r
boxplot(cars$Salary ~cars$Engineer, main = "Salary vs Engineer",xlab =
"Salary" , ylab = "Engineer",col = "pink")
```

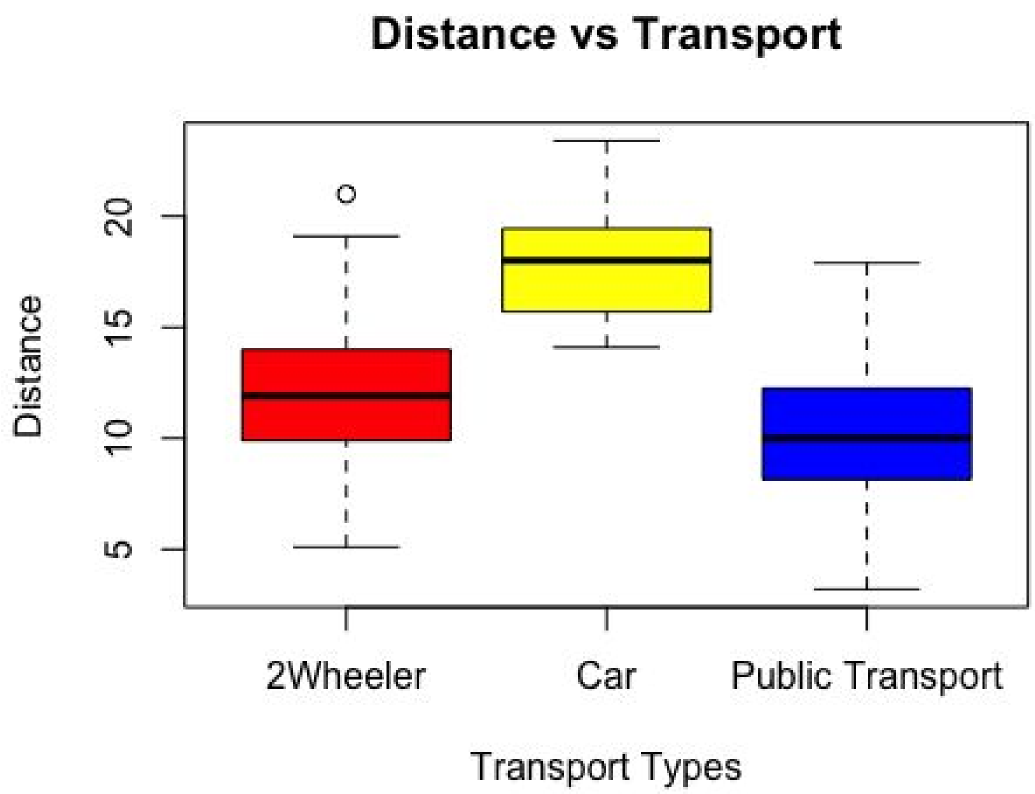



```r
boxplot(cars$Work.Exp ~ cars$Gender , main = "Work Experience vs
Gender",xlab = "Work Experience" , ylab = "Gender",col = "orange")
```

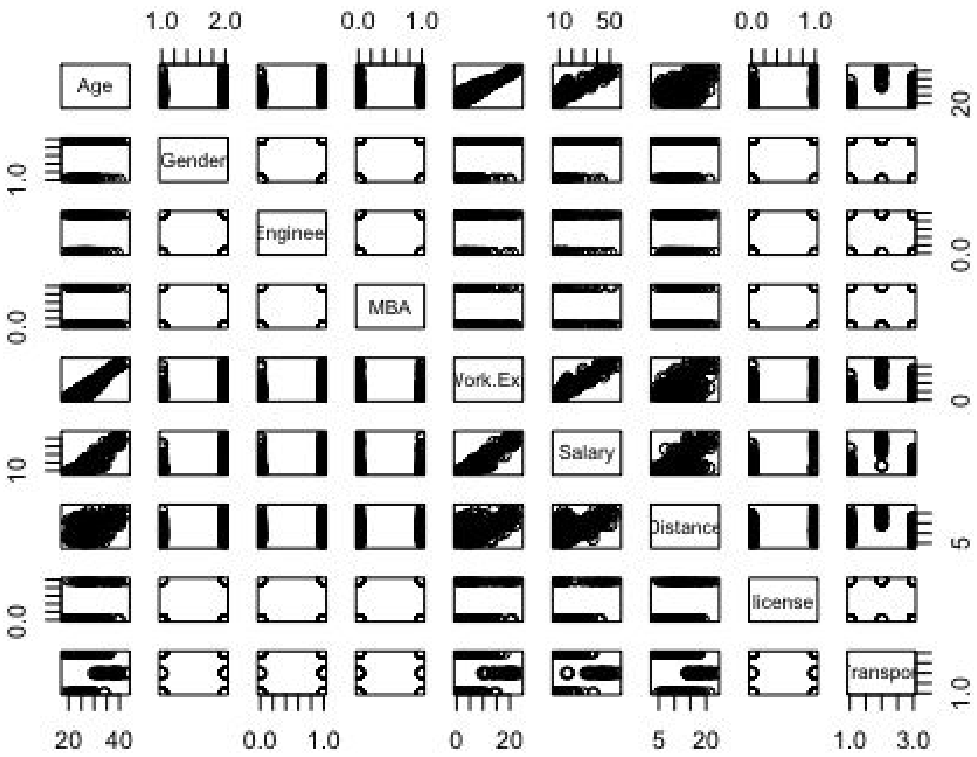
```r
plot(cars$Age~cars$Transport, main="Age vs Transport" , xlab = "Transport Types" , ylab = "Age" , col = c("pink","yellow","green"))
```




```r
boxplot(cars$Distance~cars$Transport, main="Distance vs Transport",xlab = "Transport Types" , ylab = "Distance" , col = c("red","yellow","blue"))
```



```r
pairs(cars)
```




```r
#To check the collinearity of the dataset provided
vifcor(cars[-9])
```

```
## 1 variables from the 8 input variables have collinearity problem:
##
## Work.Exp
##
## After excluding the collinear variables, the linear correlation
coefficients ranges between:
## min correlation ( MBA ~ Age ):  -0.001752158
## max correlation ( Salary ~ Age ):  0.8579114
##
## ---------- VIFs of the remained variables --------
##   Variables      VIF
## 1       Age 3.827422
## 2    Gender 1.067936
## 3  Engineer 1.012862
## 4       MBA 1.019179
## 5    Salary 4.482439
## 6  Distance 1.320710
## 7   license 1.339501
```

```r
#Remove outliers
quantile(cars$Age, c(0.95))

## 95%
##  37

cars$Age[which(cars$Age>37)] = 37
quantile(cars$Salary,c(0.95))

##    95%
## 41.915

cars$Salary[which(cars$Salary>42)]  = 42
quantile(cars$Distance,c(0.95))

##    95%
## 17.915

cars$Distance[which(cars$Distance> 18)] = 18
```

2.3.2. Insights Gained

From the above data exploration the following insights can be drawn :

1.Age,WorkExperience are right- skewed tailing at the end, giving hint of outliers.

2.Engineer , MBA , License are binary variables.

3.Salary is unevenly distributed.

4.Distance is also right-skewed, but the distribution looks good.

5.From an analysis of plot among MBA,Engineer and Age we see that all age group present in the dataset are somewhere or the other employed.

6.We do not see any appreciable difference in salary of Engr Vs Non-Engr or MBA vs Non-MBA's.

7.This is skewed towards right, again this would be on expected lines as there would be more juniors and seniors in any firm .

8.Higher the Salary the person will be using Cars rather than public commute or two wheeler

9.The collinearity between the variables were tested using the vifcor function.

10. The target variable was changed to factor and named as Car Usage.

3. Data Preparation

This is an important step since if we have imbalanced then our final implications and analysis can go wrong hence we need to prepare the data for our analysis. To prepare the data we have used the SMOTE technique to get a balanced form of the data. SMOTE enables the business to have a detailed study of the balanced data and gain important insights.

3.1. SMOTE

```
cars[!complete.cases(cars), ]

##     Age Gender Engineer MBA Work.Exp Salary Distance license
## 243  28      1        0  NA        6   13.7      9.4       0
##            Transport
## 243 Public Transport

cars_impute <-kNN(data=cars,variable =c("MBA"),k=7)

summary(cars_impute)

##       Age            Gender          Engineer            MBA
##  Min.   :18.00   Min.   :1.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:25.00   1st Qu.:1.000   1st Qu.:0.2500   1st Qu.:0.0000
##  Median :27.00   Median :2.000   Median :1.0000   Median :0.0000
##  Mean   :27.23   Mean   :1.711   Mean   :0.7488   Mean   :0.2608
##  3rd Qu.:29.00   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :37.00   Max.   :2.000   Max.   :1.0000   Max.   :1.0000
##     Work.Exp          Salary          Distance         license
##  Min.   : 0.000   Min.   : 6.500   Min.   : 3.20   Min.   :0.0000
##  1st Qu.: 3.000   1st Qu.: 9.625   1st Qu.: 8.60   1st Qu.:0.0000
##  Median : 5.000   Median :13.000   Median :10.90   Median :0.0000
##  Mean   : 5.873   Mean   :15.148   Mean   :11.19   Mean   :0.2033
##  3rd Qu.: 8.000   3rd Qu.:14.900   3rd Qu.:13.57   3rd Qu.:0.0000
##  Max.   :24.000   Max.   :42.000   Max.   :18.00   Max.   :1.0000
##             Transport     MBA_imp
##  2Wheeler        : 83   Mode :logical
##  Car             : 35   FALSE:417
##  Public Transport:300   TRUE :1
##
##
##
```

```
cars_final = subset(cars_impute,select = Age:Transport)
summary(cars_final)
```

```
##       Age            Gender          Engineer          MBA
##  Min.   :18.00   Min.   :1.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:25.00   1st Qu.:1.000   1st Qu.:0.2500   1st Qu.:0.0000
##  Median :27.00   Median :2.000   Median :1.0000   Median :0.0000
##  Mean   :27.23   Mean   :1.711   Mean   :0.7488   Mean   :0.2608
##  3rd Qu.:29.00   3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :37.00   Max.   :2.000   Max.   :1.0000   Max.   :1.0000
##     Work.Exp          Salary          Distance         license
##  Min.   : 0.000   Min.   : 6.500   Min.   : 3.20   Min.   :0.0000
##  1st Qu.: 3.000   1st Qu.: 9.625   1st Qu.: 8.60   1st Qu.:0.0000
##  Median : 5.000   Median :13.000   Median :10.90   Median :0.0000
##  Mean   : 5.873   Mean   :15.148   Mean   :11.19   Mean   :0.2033
##  3rd Qu.: 8.000   3rd Qu.:14.900   3rd Qu.:13.57   3rd Qu.:0.0000
##  Max.   :24.000   Max.   :42.000   Max.   :18.00   Max.   :1.0000
##             Transport
##  2Wheeler        : 83
##  Car             : 35
##  Public Transport:300
##
##
##
```

```
table(cars_final$Transport)
```

```
##
##         2Wheeler              Car Public Transport
##               83               35              300
```

```
cars_final$CarUsage<-ifelse(cars_final$Transport =='Car',1,0)
table(cars_final$CarUsage)
```

```
##
##   0   1
## 383  35
```

```
cars_final$CarUsage<-as.factor(cars_final$CarUsage)
```

```
set.seed(400)
carindex<-createDataPartition(cars_final$CarUsage, p=0.7,list =
FALSE,times = 1
)
train<-cars_final[carindex,]
test<-cars_final[-carindex,]
prop.table(table(train$CarUsage))
```

```
##
##          0          1
## 0.91496599 0.08503401
```

```
train<-train[,c(1:8,10)]
test<-test[,c(1:8,10)]

#Applying SMOTE on the train dataset
attach(train)
carsdataSMOTE<-SMOTE(CarUsage~., train, perc.over = 250,perc.under = 150)
prop.table(table(carsdataSMOTE$CarUsage))

##
##   0   1
## 0.5 0.5
```

3.2. Insights from above imputation :

1.As suggested earlier, there was a missing value which is treated by KNN imputation.
2.The biggest challenge in the dataset we find after creating the variable is the data imbalance between car takers and others.
3.To deal with this problem, we will use SMOTE technique which will distribute the variables in equal measure.

4. Model Creation

This step enables us to create and test the best model that would enable us to determine the best method to let the business know about the commute usually taken by the employees to their work stations.

The models have been narrowed down to 5 different Machine Learning algorithms and they are listed below :

1.Logistic Regression

2.KNN

3.Naive Bayes

4.Bagging

5.Boosting

Later in the last section of the report we will also be doing a model evaluation to conclude that which model is most efficient in letting the business in helping knowing the best commute taken by most of the employees.

4.1. Logistic Regression

4.1.1. Model Creation

```
pred_var<-'CarUsage'
predictors<-c("Age","Work.Exp","Salary","Distance","license","Engineer","MBA","Gender")
train.ctrl<-trainControl(method = 'repeatedcv',number = 10,repeats = 3)
logreg<-train(carsdataSMOTE[,predictors],carsdataSMOTE[,pred_var],method = "glm",
              family="binomial",trControl = train.ctrl)
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: algorithm did not converge
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

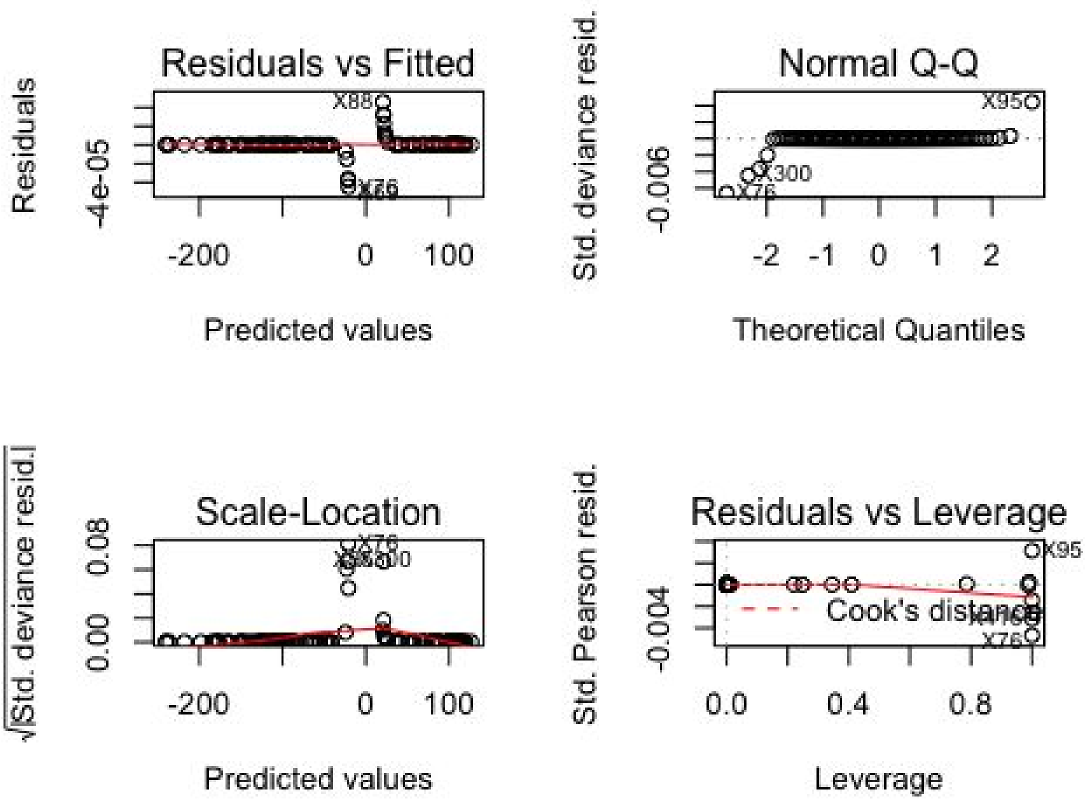
```
summary(logreg$finalModel)
```

```
## 
## Call:
## NULL
## 
## Deviance Residuals: 
##        Min          1Q      Median          3Q         Max  
## -4.454e-05  -2.100e-08   0.000e+00   2.100e-08   4.581e-05  
## 
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.917e+02  1.844e+06   0.000    1.000
## Age          1.452e+01  8.769e+04   0.000    1.000
## Work.Exp    -9.931e+00  7.615e+04   0.000    1.000
## Salary       4.132e+00  1.086e+04   0.000    1.000
## Distance     1.081e+01  1.748e+04   0.001    1.000
## license      3.163e+01  4.826e+04   0.001    0.999
## Engineer     9.412e+00  1.402e+05   0.000    1.000
## MBA          1.474e+01  1.296e+05   0.000    1.000
## Gender      -2.782e+01  1.027e+05   0.000    1.000
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 2.0794e+02  on 149  degrees of freedom
## Residual deviance: 1.0018e-08  on 141  degrees of freedom
## AIC: 18
## 
## Number of Fisher Scoring iterations: 25
```

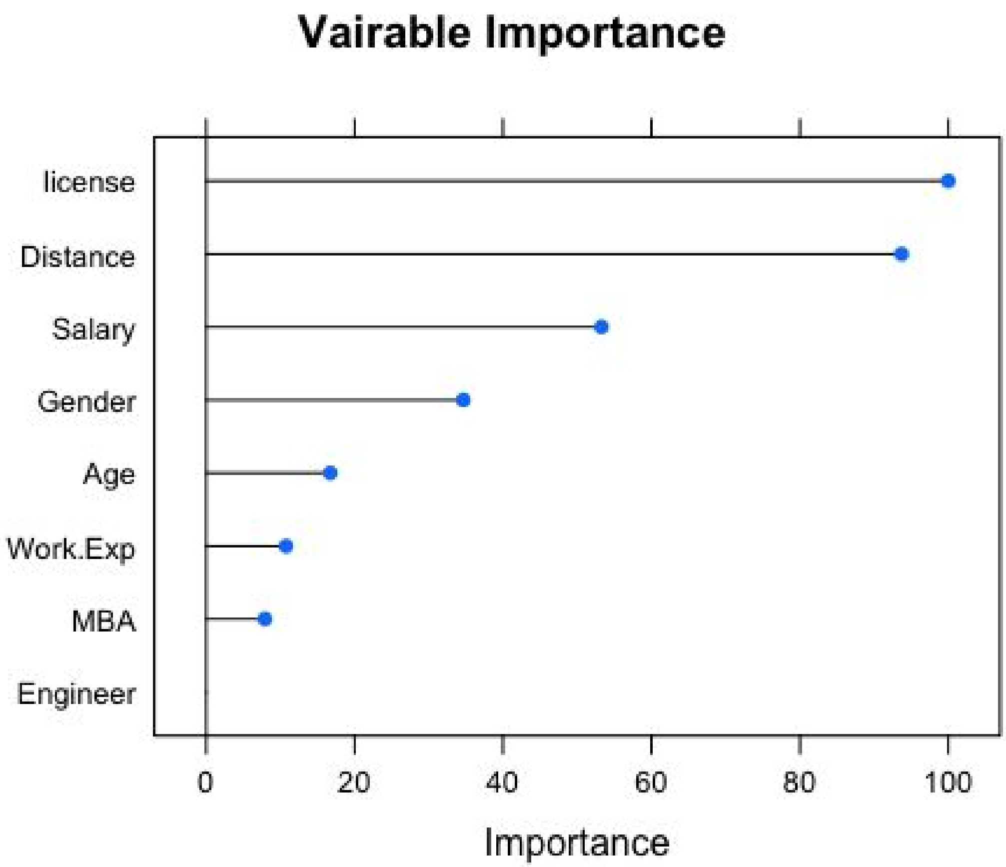
```
par(mfrow = c(2,2))
plot(logreg$finalModel)
```

```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```

```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```




```
logreg.coeff<-exp(coef(logreg$finalModel))
varImp(object = logreg)

## glm variable importance
##
##           Overall
## license   100.000
## Distance   93.689
## Salary     53.286
## Gender     34.656
## Age        16.729
## Work.Exp   10.758
## MBA         7.923
## Engineer    0.000

plot(varImp(object = logreg), main="Vairable Importance")
```




```
logreg.prediction<-predict.train(object = logreg,test[,predictors],type =
"raw")

logistic_regression_confusion =
confusionMatrix(logreg.prediction,test[,pred_var], positive='1')
logistic_regression_confusion

## Confusion Matrix and Statistics
##
##             Reference
## Prediction   0   1
##          0 112   1
##          1   2   9
##
##                Accuracy : 0.9758
##                  95% CI : (0.9309, 0.995)
##     No Information Rate : 0.9194
##     P-Value [Acc > NIR] : 0.008296
##
##                   Kappa : 0.844
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.90000
##             Specificity : 0.98246
##          Pos Pred Value : 0.81818
```

```
##          Neg Pred Value : 0.99115
##              Prevalence : 0.08065
##          Detection Rate : 0.07258
##    Detection Prevalence : 0.08871
##       Balanced Accuracy : 0.94123
##
##        'Positive' Class : 1
##
```

4.1.2. Insights

1.The accuracy is on a higher side with a good chunk of minority class predicted correctly.

2.Also Logistic regression gives fair idea about valuable predictors.

3.From the graph we can see that Age is the most valuable predictor followed by License and Gender.

4.2. Naive Bayes

4.2.1. Model Creation

```
x = carsdataSMOTE[,-9]
y = carsdataSMOTE$CarUsage
modelNB = train(x,y,'nb',trControl=trainControl(method='cv',number=10))

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 9

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 2

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 3

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 5

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 13
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 1
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 6
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 8
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 13
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 8
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 2
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 13
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 16
```

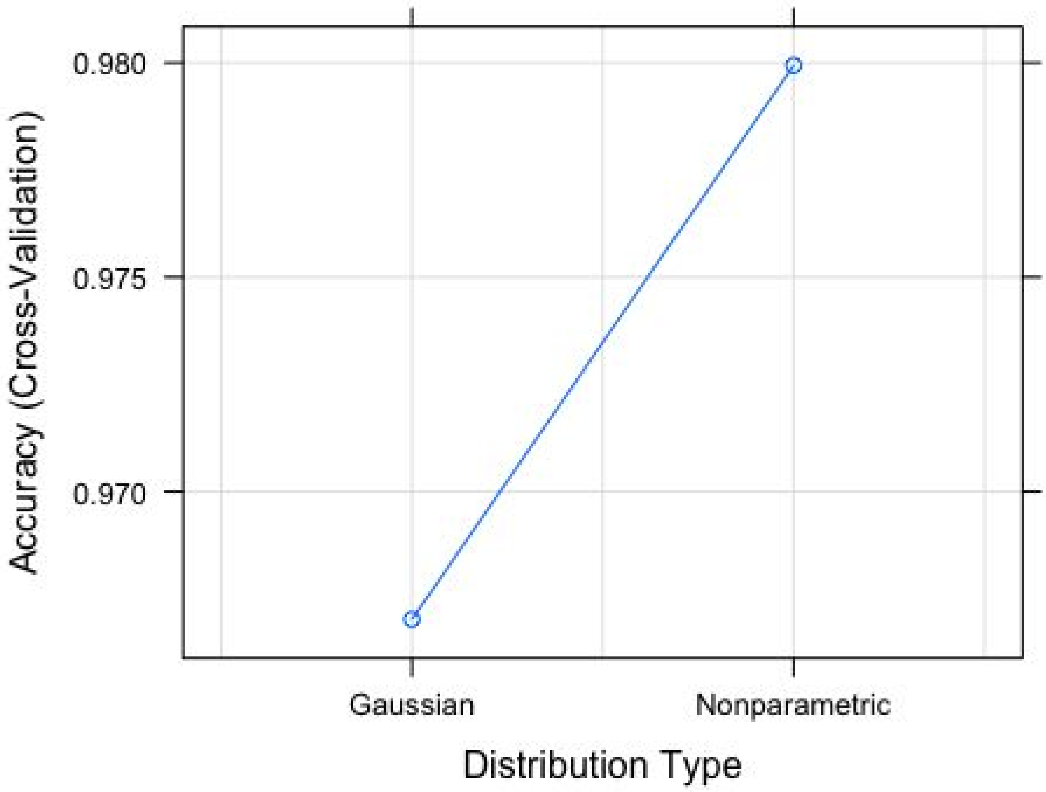
```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 13
```

```
modelNB
```

```
## Naive Bayes
##
## 150 samples
##   8 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 135, 135, 135, 134, 136, 135, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.9670238  0.9339264
##    TRUE      0.9799405  0.9599399
##
## Tuning parameter 'fL' was held constant at a value of 0
```

```
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = TRUE
##  and adjust = 1.
```

```
plot(modelNB)
```




```
predict_NB = predict(modelNB,test)

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 85

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes
with
## observation 87

table(predict_NB,test$CarUsage)

##
## predict_NB   0   1
##          0 111   1
##          1   3   9

NaiveBayes_confusion = confusionMatrix(predict_NB,test$CarUsage)
NaiveBayes_confusion
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction   0   1
##          0 111   1
##          1   3   9
##
##                Accuracy : 0.9677
##                  95% CI : (0.9195, 0.9911)
##     No Information Rate : 0.9194
##     P-Value [Acc > NIR] : 0.02476
##
##                   Kappa : 0.8006
##
##  Mcnemar's Test P-Value : 0.61708
##
##             Sensitivity : 0.9737
##             Specificity : 0.9000
##          Pos Pred Value : 0.9911
##          Neg Pred Value : 0.7500
##              Prevalence : 0.9194
##          Detection Rate : 0.8952
##    Detection Prevalence : 0.9032
##       Balanced Accuracy : 0.9368
##
##        'Positive' Class : 0
##
```

4.2.2. Insights

1.Naive Bayes like Logistic Regression gives a pretty good accuracy.

2.A good number of minority class is predicted correctly.

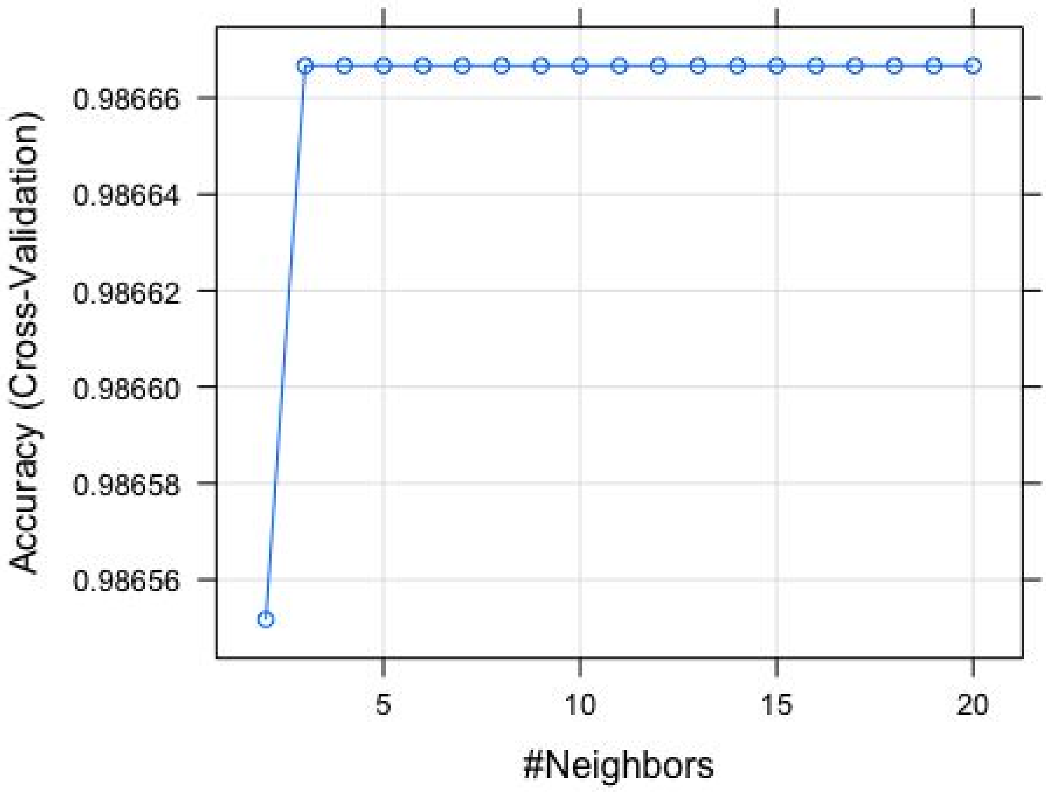
4.3 KNN MODEL

4.3.1 Model Creation

```
trControl <- trainControl(method  = "cv", number  = 10)
KNN_Model <- caret::train(CarUsage ~ .,
                       method     = "knn",
                       tuneGrid   = expand.grid(k = 2:20),
                       trControl  = trControl,
                       metric     = "Accuracy",
                       preProcess = c("center","scale"),
                       data       = train)

KNN_Model
```

```
## k-Nearest Neighbors
##
## 294 samples
##   8 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (8), scaled (8)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 264, 264, 266, 265, 264, 265, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    2  0.9865517  0.9194193
##    3  0.9866667  0.9194847
##    4  0.9866667  0.9194847
##    5  0.9866667  0.9194847
##    6  0.9866667  0.9194847
##    7  0.9866667  0.9194847
##    8  0.9866667  0.9194847
##    9  0.9866667  0.9194847
##   10  0.9866667  0.9194847
##   11  0.9866667  0.9194847
##   12  0.9866667  0.9194847
##   13  0.9866667  0.9194847
##   14  0.9866667  0.9194847
##   15  0.9866667  0.9194847
##   16  0.9866667  0.9194847
##   17  0.9866667  0.9194847
##   18  0.9866667  0.9194847
##   19  0.9866667  0.9194847
##   20  0.9866667  0.9194847
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 20.
```

```r
plot(KNN_Model)
```




```
knn_predictions <- predict(KNN_Model,test)
knn_confusion = confusionMatrix(knn_predictions, test$CarUsage)
knn_confusion

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 113   2
##          1   1   8
##
##                Accuracy : 0.9758
##                  95% CI : (0.9309, 0.995)
##     No Information Rate : 0.9194
##     P-Value [Acc > NIR] : 0.008296
##
##                   Kappa : 0.829
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.9912
##             Specificity : 0.8000
##          Pos Pred Value : 0.9826
##          Neg Pred Value : 0.8889
##              Prevalence : 0.9194
##          Detection Rate : 0.9113
```

```
##    Detection Prevalence : 0.9274
##       Balanced Accuracy : 0.8956
##
##        'Positive' Class : 0
##
```

4.3.2. Insights

Unlike Naïve Bayes and Logistic Regression, KNN couldn't provide the same amount of accuracy and high minority class prediction.

4.4. Bagging

4.4.1. Model Creation

```
cars_final1=cars_final[,-9]
cars_final1$Gender=as.factor(cars_final1$Gender)
cars_final1$Engineer=as.numeric(cars_final1$Engineer)
cars_final1$MBA=as.numeric(cars_final1$MBA)
cars_final1$Age=as.numeric(cars_final1$Age)
cars_final1$Work.Exp=as.numeric(cars_final1$Work.Exp)
cars_final1$license=as.numeric(cars_final1$license)
cars_final1$CarUsage=as.factor(cars_final1$CarUsage)
#bag.train$CarUsage=as.factor(bag.train$CarUsage)
index<-sample(nrow(cars_final1),round(0.7*nrow(cars_final1)))
bag.train<-cars_final1[index,]
bag.test<-cars_final1[-index,]

mod.bagging = bagging(CarUsage ~.,
                      data=bag.train,
                      control=rpart.control(maxdepth=5, minsplit=4))

summary(mod.bagging)

##
## Bagging classification trees with 25 bootstrap replications
##
## Call: bagging.data.frame(formula = CarUsage ~ ., data = bag.train,
##     control = rpart.control(maxdepth = 5, minsplit = 4))

## $y
##   [1] 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1
##  [36] 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0
0 1
##  [71] 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0
## [106] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
```

```
0 0
## [141] 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0
0 1
## [176] 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0
## [211] 0 0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0
## [246] 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0
0 0
## [281] 0 0 0 0 0 0 0 0 0 0 0 0 0
## Levels: 0 1
##
## $X
##     Age Gender Engineer MBA Work.Exp Salary Distance license
## 172  22      1        1   0        0    6.5      7.6       0
## 48   24      1        1   1        1    8.8     12.6       1
## 195  27      1        0   1        4   13.6      8.2       0
## 241  27      2        1   0        3   10.6      9.3       0
## 38   23      2        1   0        0    6.9     11.7       0
## 43   28      2        1   0        5   13.9     12.2       1
## 205  33      2        1   0       11   15.6      8.5       0
## 166  28      1        1   0        9   21.7      7.3       0
## 393  28      1        1   0        6   13.9     15.1       0
## 285  30      2        1   1        8   14.9     10.4       0
## 250  27      2        1   1        6   12.9      9.5       0
## 113  34      2        1   0       14   38.0     18.0       1
## 26   23      1        0   0        4   11.6     10.7       0
## 399  28      1        0   1        9   23.8     15.5       0
## 325  21      2        1   1        3    9.9     11.8       0
## 32   24      2        1   0        0    8.0     11.0       1
## 251  24      2        0   0        2    8.6      9.5       1
## 316  26      2        0   0        4   12.6     11.5       1
## 228  28      2        0   1       10   20.7      9.0       0
## 154  28      2        1   0        6   13.6      6.9       1
## 199  24      2        1   1        6   10.6      8.4       1
## 175  24      2        0   0        2    8.7      7.6       0
## 297  25      2        1   0        3   10.7     10.8       0
## 223  28      2        1   0        6   13.7      8.9       0
## 353  22      2        1   0        0    6.8     12.7       0
## 7    23      2        1   1        3   11.7      7.2       0
## 344  29      2        0   0        7   14.7     12.3       0
## 173  29      2        1   0        5   15.4      7.6       1
## 405  23      2        1   1        2    8.9     15.8       0
## 141  26      1        0   0        3    9.5      6.2       0
## 312  30      2        1   0       10   13.8     11.4       0
## 37   23      2        1   0        4   10.6     11.4       0
## 50   25      2        0   0        5   13.7     12.7       1
## 57   22      2        1   0        0    6.9     13.2       0
## 101  36      1        1   0       17   38.0     18.0       1
## 148  27      1        1   0        5   12.5      6.4       0
## 392  27      2        1   0        9   20.8     15.1       0
```

```
## 87   36        2        1   0        16   42.0     14.4       1
## 374  28        2        1   0         7   12.7     13.6       0
## 275  27        2        1   0         4   13.8     10.1       0
## 307  29        2        1   0         5   14.9     11.2       0
## 18   29        1        0   0         7   14.6      9.2       0
## 220  29        2        1   0         9   22.8      8.9       0
## 235  23        2        1   1         0    6.6      9.2       0
## 77   26        2        1   0         2   10.0     16.4       1
## 240  23        1        0   0         0    6.8      9.3       0
## 213  28        2        0   0         6   13.8      8.6       0
## 360  30        1        1   0         6   15.8     12.9       0
## 299  30        2        0   0         8   14.6     10.9       1
## 76   26        1        1   0         6   23.0     16.3       0
## 107  37        2        1   0        19   42.0     18.0       1
## 184  25        2        1   0         3   10.6      8.1       0
## 3    27        1        1   0         9   15.5      6.1       0
## 268  24        2        1   0         0    7.6     10.0       0
## 400  27        2        1   1         6   12.9     15.6       0
## 109  33        2        1   1        10   17.0     18.0       0
## 401  26        2        1   0         6   18.8     15.6       0
## 396  21        2        1   1         3    9.8     15.3       0
## 61   23        2        0   0         0    6.9     13.7       0
## 94   37        2        1   0        21   40.9     16.3       0
## 308  27        2        1   1         6   12.8     11.3       0
## 39   24        2        1   0         4   12.7     11.7       0
## 210  23        2        0   0         2    8.8      8.6       0
## 145  26        1        0   0         3    9.5      6.3       0
## 62   24        1        1   0         2    8.9     13.8       0
## 255  26        2        1   0         2    9.5      9.6       0
## 319  30        1        0   0         6   15.6     11.6       0
## 336  31        2        1   0        10   14.9     12.1       0
## 314  28        1        0   0         9   23.8     11.4       0
## 84   37        2        1   0        19   42.0     14.1       1
## 282  36        2        1   1        18   28.7     10.4       1
## 23   34        2        1   1        14   36.9     10.4       1
## 165  31        2        0   0         9   15.6      7.3       0
## 315  27        2        1   0         6   12.7     11.5       0
## 58   26        1        1   0         8   20.9     13.4       0
## 110  37        2        1   0        22   42.0     18.0       1
## 226  26        2        1   0         4   12.7      9.0       1
## 30   26        1        1   0         4   12.6     11.0       0
## 106  37        2        0   0        21   42.0     18.0       1
## 248  26        2        1   1         3   10.8      9.4       0
## 40   23        2        1   0         0    7.7     11.7       0
## 388  22        1        1   0         0    6.8     14.5       0
## 33   26        1        1   0         4   12.9     11.1       0
## 41   27        1        1   0         5   12.8     11.8       0
## 300  33        2        1   1        14   34.9     10.9       0
## 158  33        2        0   0        13   36.6      7.1       1
## 239  27        2        1   0         5   12.5      9.3       0
## 168  23        1        1   0         0    7.7      7.4       0
```

```
## 406  25      2        1   0        2   8.9     15.8       0
## 379  26      2        0   0        7  18.8     13.8       0
## 301  32      2        1   0       12  15.7     10.9       0
## 163  30      1        1   0        8  14.4      7.2       0
## 409  31      2        1   0        8  15.9     16.4       0
## 98   33      2        1   0       14  33.0     17.3       0
## 217  25      1        1   0        3  10.6      8.8       0
## 12   21      2        0   1        3  10.6      7.7       0
## 171  28      1        1   0        5  13.6      7.5       0
## 225  26      1        1   0        3  10.8      9.0       0
## 131  27      2        1   0        4  13.4      5.5       1
## 201  25      2        1   0        3   9.8      8.4       0
## 346  27      2        1   0        4  13.8     12.4       0
## 242  33      2        1   1       11  15.6      9.3       0
## 157  25      2        1   0        3  10.5      7.1       0
## 349  30      2        1   1        8  14.7     12.6       0
## 395  23      2        1   0        1   7.9     15.2       0
## 309  28      1        1   0        5  13.7     11.3       0
## 386  28      2        1   0        4  14.9     14.3       0
## 25   26      1        1   0        2   9.8     10.7       0
## 368  26      2        0   0        4  12.7     13.3       0
## 219  26      1        1   0        3  10.8      8.9       0
## 5    25      1        0   0        3   9.6      6.7       0
## 27   25      2        1   1        7  13.6     10.7       0
## 415  25      2        1   0        3   9.9     17.2       0
## 287  24      2        1   1        1   7.9     10.5       0
## 44   25      1        1   0        5  18.9     12.2       0
## 292  27      1        1   0        5  12.9     10.6       0
## 211  28      1        0   0        6  13.6      8.6       0
## 332  28      2        1   1        7  13.9     12.1       0
## 351  24      2        1   0        1   8.9     12.7       0
## 136  23      2        1   0        2   8.5      6.1       0
## 198  25      2        1   0        4  11.6      8.3       0
## 234  29      2        1   1       11  25.9      9.1       0
## 174  31      2        1   0        9  15.6      7.6       0
## 143  26      2        0   0        2   8.5      6.3       1
## 122  28      1        1   1        5  13.4      4.5       0
## 378  24      2        1   1        2   8.9     13.8       0
## 93   37      2        1   1       18  41.0     15.9       1
## 326  26      1        1   0        6  11.8     11.9       0
## 377  31      2        1   0        7  15.9     13.7       0
## 317  26      2        1   1        7  20.9     11.6       0
## 397  29      2        0   0        5  14.8     15.4       0
## 249  26      2        1   0        2   9.6      9.5       0
## 128  25      1        1   0        4  11.5      5.2       0
## 381  30      2        1   0        7  14.9     14.0       0
## 24   28      2        1   0        5  14.7     10.5       1
## 67   25      1        1   0        2   8.8     15.2       0
## 281  28      2        1   0        3  10.8     10.2       1
## 253  24      2        1   1        0   7.6      9.5       0
## 291  30      2        1   1        8  14.6     10.6       0
```

```
## 382  27       2        0   0       9   23.9     14.1       0
## 247  26       1        0   0       4   12.9      9.4       0
## 365  28       2        1   0       4   14.8     13.0       0
## 385  30       2        1   0       8   14.8     14.3       0
## 190  30       1        1   0       8   14.6      8.1       0
## 82   26       2        1   0       4   13.0     18.0       1
## 303  24       2        1   1       3    9.9     10.9       0
## 125  26       1        1   0       3   10.5      5.1       0
## 88   31       2        1   0      12   34.0     14.4       1
## 288  28       1        1   0       6   13.9     10.5       0
## 348  27       2        1   1       5   13.7     12.6       0
## 265  31       1        1   0      10   14.9      9.9       0
## 335  25       1        1   0       5   17.8     12.1       0
## 413  29       1        1   0       6   14.9     17.0       0
## 370  27       2        1   0       6   12.9     13.3       0
## 133  25       2        1   1       4   11.5      5.6       0
## 286  26       1        1   0       6   17.8     10.4       0
## 91   34       2        1   0      14   42.0     15.1       1
## 418  23       2        0   0       3    9.9     17.9       0
## 266  25       2        1   1       3    9.7      9.9       0
## 49   24       1        1   1       2    8.7     12.6       0
## 298  26       1        1   1       4   12.8     10.8       0
## 338  34       2        1   1      16   34.9     12.2       0
## 342  18       2        1   0       0    6.8     12.2       0
## 29   21       1        0   0       3    9.8     11.0       0
## 21   27       1        1   0       5   12.8      9.7       0
## 361  30       2        1   1       9   14.9     12.9       0
## 152  26       2        0   0       3    9.5      6.8       0
## 189  19       1        1   0       1    7.5      8.1       0
## 35   29       2        1   0      11   22.7     11.3       1
## 103  37       2        1   0      18   42.0     18.0       1
## 97   37       1        1   0      20   42.0     17.0       1
## 73   23       2        1   0       0    8.0     15.9       0
## 180  28       2        1   0       5   13.6      7.9       0
## 263  27       1        1   1       4   13.8      9.8       0
## 116  37       2        1   0      19   42.0     18.0       1
## 333  30       2        0   0      10   29.9     12.1       0
## 78   25       1        1   0       1    8.9     16.8       0
## 129  27       2        1   0       4   13.5      5.3       1
## 350  27       2        1   0       8   20.7     12.6       0
## 254  26       2        1   0       4   12.9      9.6       0
## 16   28       1        0   0      10   19.7      9.0       0
## 80   23       1        1   1       2    9.0     17.9       0
## 305  31       1        0   1       9   14.6     11.1       0
## 66   22       2        0   0       0    6.8     15.2       1
## 364  27       1        1   0       8   24.9     13.0       0
## 380  26       2        0   1       5   12.8     13.9       0
## 296  24       2        1   1       0    7.8     10.7       0
## 102  37       2        1   0      21   42.0     18.0       1
## 108  37       2        1   0      22   42.0     18.0       1
## 177  26       2        1   1       4   12.4      7.6       0
```

```
## 246  26       2         0   0          3    9.9       9.4         0
## 258  26       2         1   0          3   10.5       9.7         1
## 187  22       1         1   0          2   11.7       8.1         0
## 387  30       2         1   0          6   15.8      14.3         0
## 70   30       2         1   0          8   14.9      15.5         1
## 186  26       2         1   1          8   21.6       8.1         1
## 64   27       1         1   0          7   23.8      14.4         0
## 176  29       2         1   0          6   14.6       7.6         0
## 341  26       2         1   0          3   10.7      12.2         1
## 362  26       2         1   0          5   11.7      13.0         0
## 363  30       2         1   0          9   14.8      13.0         0
## 277  35       1         1   0         16   28.7      10.2         0
## 398  28       2         1   0          5   13.8      15.5         0
## 222  21       1         1   0          3    9.8       8.9         0
## 366  24       2         1   0          1    7.8      13.1         0
## 280  24       2         1   0          0    7.6      10.2         0
## 160  30       2         1   0          8   14.6       7.1         0
## 214  20       2         1   0          2    8.8       8.7         1
## 321  26       2         1   1          6   11.7      11.7         0
## 232  25       2         0   0          4   11.9       9.1         0
## 231  24       2         1   1          0    7.9       9.1         0
## 15   27       2         0   1          8   15.6       9.0         0
## 118  37       2         1   1         21   42.0      18.0         1
## 181  27       2         0   0          3    9.5       7.9         1
## 146  24       2         1   1          2    8.6       6.4         0
## 330  30       2         1   0          8   14.8      12.0         0
## 95   32       1         1   0         14   30.9      16.5         0
## 105  30       2         1   1         11   35.0      18.0         1
## 408  28       2         1   0          5   13.9      16.4         0
## 358  25       2         1   0          3   10.8      12.8         0
## 2    24       2         1   0          6   10.6       6.1         0
## 262  28       2         0   0          5   14.5       9.8         1
## 112  37       2         1   1         24   42.0      18.0         1
## 179  29       1         0   0          7   14.6       7.7         0
## 216  26       2         1   0          5   12.8       8.8         0
## 212  30       2         1   0          7   15.6       8.6         1
## 185  21       1         1   0          3    9.6       8.1         0
## 150  27       2         1   0          6   12.6       6.5         0
## 52   22       2         1   1          0    6.9      13.0         0
## 383  27       2         1   0          4   13.9      14.2         0
## 257  28       2         1   1          5   14.8       9.7         0
## 322  29       1         1   0          7   14.8      11.7         0
## 54   23       1         1   1          2    8.8      13.1         0
## 260  28       2         1   0          6   13.6       9.7         1
## 192  27       2         0   0          6   12.6       8.1         0
## 204  31       2         1   0         10   14.8       8.4         0
## 202  32       2         1   0         10   15.7       8.4         0
## 127  27       2         1   0          4   13.5       5.2         0
## 359  28       2         1   0          6   13.8      12.9         0
## 327  30       2         1   0          6   15.7      11.9         1
## 100  31       2         0   0         11   33.0      17.8         1
```

```
## 11   26       2        0   0        4   12.6      7.5        0
## 391  34       1        0   0       14   28.8     15.0        0
## 126  22       2        1   0        1    7.5      5.1        0
## 272  27       2        0   1        5   13.9     10.0        0
## 376  28       2        1   0        4   14.9     13.7        0
## 227  24       2        1   0        4   10.9      9.0        0
## 416  27       1        0   0        4   13.9     17.3        0
## 233  27       2        0   0        7   12.5      9.1        0
## 137  23       2        0   0        2    8.6      6.1        0
## 259  25       2        1   0        1    8.6      9.7        0
## 45   26       1        1   0        2    9.8     12.2        0
## 104  37       2        1   0       20   42.0     18.0        1
## 304  26       2        1   0        2    8.6     11.0        1
## 256  26       2        1   0        3   10.6      9.6        0
## 4    25       2        0   0        1    7.6      6.3        0
## 51   34       2        1   1       15   37.0     12.9        1
## 329  28       2        1   1        6   13.7     11.9        0
## 293  27       2        0   1        8   20.7     10.7        0
## 373  27       2        1   0        1    8.9     13.6        0
## 161  28       1        0   1        5   14.6      7.2        0
## 313  29       2        1   0        9   13.7     11.4        0
## 224  28       2        0   1        3    9.5      9.0        0
## 114  37       2        1   0       20   42.0     18.0        1
## 276  29       2        0   0        6   14.6     10.1        0
## 367  24       2        1   0        4   13.8     13.2        0
## 99   34       2        1   1       16   36.0     17.8        1
## 149  30       1        1   0        8   14.6      6.5        0
## 72   24       1        1   0        1    8.8     15.8        0
## 274  29       2        1   0        6   14.8     10.1        0
## 279  29       2        1   1        6   14.6     10.2        1
## 14   24       2        1   0        6   12.7      8.7        0
## 345  24       2        1   0        1    7.7     12.4        1
## 384  26       2        1   0        4   12.8     14.2        0
## 36   30       1        1   0        8   14.7     11.4        1
## 86   37       2        1   0       22   42.0     14.1        1
## 81   29       2        0   1        7   15.0     18.0        1
## 142  28       2        1   1        7   13.6      6.3        0
## 221  24       1        1   0        6   10.5      8.9        0
## 352  21       2        0   0        3    9.8     12.7        0
## 372  27       2        1   1        8   21.8     13.4        0
## 178  24       2        1   1        1    8.5      7.7        0
## 347  26       2        1   1        5   12.7     12.5        0
## 55   25       1        0   0        2    8.9     13.2        0
## 140  24       2        0   0        2    8.5      6.2        0
## 414  29       2        1   1        8   13.9     17.1        0
## 320  26       1        1   0        8   14.6     11.6        0
## 340  26       2        0   0        3    9.8     12.2        1
## 182  20       1        0   1        1    8.5      7.9        0
## 6    21       2        0   0        3    9.5      7.1        0
## 47   28       2        0   0        5   14.9     12.5        1
## 68   24       2        0   0        0    6.9     15.3        0
```

```
## 200  32      2        1   0       11   14.7      8.4       0
##
## $mtrees
## $mtrees[[1]]
## $bindx
##   [1] 176 197  90 162 288 236  64 246 192 127   8   3  30 210 288 125
250
##  [18] 122  25 179  64 166  96 184 111 229 196  40 156   3 249  52 293
257
##  [35]  57 265  38 159  62 109 291  80 147  95  68 223 251 134 108 169
202
##  [52] 208 168  21 186  98  11 221 286 117  50 109 227  34   4 225 128
88
##  [69]  32  13  78 199 190 178  74 287 245 144 201  68  99 265 132 140
101
##  [86] 260  57 290 148  95 200 198 155  79 165  20 200 246 118 263 228
272
## [103] 138 220 208 148  65 203  31  74 204 186 151 130  50  42 204  44
41
## [120]  38  19 123 262  24 215 266  94  75  57 108 164 106  92 114 288
106
## [137]  26 137 116 178  39 234  59  98  50 252  15 154 227 131 256 155
263
## [154]  91 168 221  68 222  66 223 275  34  65 103 174 242 209 177 265
29
## [171] 166  27 252 106  95 167  91 185 230 118 122 248 175  38 227 285
57
## [188]  40 257  60  59 277 219 228 217  60  60  59 210   9  39 208 119
187
## [205] 196  33 143   6  96  53 111 154  68 249 198 248  11 199 235  73
271
## [222]  95 164  79 124 250 111 265 239  84  98  19 188  94 108 267  26
139
## [239] 293 108  11  38 225 147 163 119  58 276 163 234 222  92 154 241
193
## [256] 110 157 283  29 239 282 116  84   2 164  52 207 212   3 215 169
9
## [273] 146 157  36 203 213 284 202 133 213  32 239  67  22 207 188  14
224
## [290] 238 197 271 109
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 27 0 (0.90784983 0.09215017)
##   2) Salary< 30.4 263  0 0 (1.00000000 0.00000000) *
##   3) Salary>=30.4 30  3 1 (0.10000000 0.90000000)
##     6) Distance< 13.5 3  0 0 (1.00000000 0.00000000) *
```

```
##     7) Distance>=13.5 27  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[2]]
## $bindx
##   [1] 273 241 261 147 239 251 279 174 203 252  96 133  37 247 231 233
274
##  [18] 177 182  93 268 172 222 152 227 180  71 170 189 186  71 250 246
102
##  [35] 263 171 173 220  78 194 135 116 257  68 184  57 247 231 278 116
8
##  [52] 238  30 285  70  55 121 209 100  11 228 264 126  17 240 145  67
84
##  [69] 214 107 268 242 198  66 262 122 292  81 235  48  98  86 120 108
173
##  [86] 244 211 252 144 184 144  43  78 238 244  61 195 207 269  76 162
89
## [103]  47  42  35   3 292 203 222 217  34  85 228  48 119  75 151 255
28
## [120] 190 246 292 167  62 160 288 166  25 147 250  96 277 127 185  64
218
## [137] 168  89 271 104 140 154 203 117 276 247  98 241 119  43 200 140
186
## [154]  50 145 164 269 168 144 167 158 175 147 235  24  57 140 232 282
276
## [171]  42 115 190 232  66 122 103  91 110 221  28 203  60 269 100 179
268
## [188] 288 189 193 131 161 292 199 102  90 204  99 113 151  28 284 244
36
## [205] 152 154 152 100 175 271 266 199 262 136  59 113 167  53 192 188
190
## [222] 270  99  32 205  67   9  28 232 195 169  54 241  73 160  65 272
268
## [239]  55 252 185 253 236 123 104 149 206  95 277  58   9  56 125 232
289
## [256] 152 109 233 237  36 155 225  92 259  94 281 129  83 176 252 178
78
## [273]  49  96 239 259   5  70  90 164  69 121  99 121  44 217 165 245
6
## [290] 169 110 272 288
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 25 0 (0.914675768 0.085324232)
```

```
##    2) Salary< 30.4 265  1 0 (0.996226415 0.003773585)
##      4) Age< 32.5 260  0 0 (1.000000000 0.000000000) *
##      5) Age>=32.5 5  1 0 (0.800000000 0.200000000)
##       10) Work.Exp>=10.5 4  0 0 (1.000000000 0.000000000) *
##       11) Work.Exp< 10.5 1  0 1 (0.000000000 1.000000000) *
##    3) Salary>=30.4 28  4 1 (0.142857143 0.857142857)
##      6) Distance< 13.5 4  0 0 (1.000000000 0.000000000) *
##      7) Distance>=13.5 24  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[3]]
## $bindx
##   [1]  78 171 284 228 273 281  47  69  23  92 160  57 132 187 113 188
29
##  [18]  55   6 176 256 143 103  89  35 177  90 169 234 135 109  85 167
196
##  [35] 168   5 238 225 157   8 198 219 198 155 212 137 159  34 210 227
55
##  [52] 161  50  25 138 175  63 260  94 235  10  79  40 236 239  68 292
237
##  [69] 291  46   9 146  75  18  36  29 140 290  26 158 239 152 260 199
175
##  [86] 270 229 231 217 240 203 189 168 255 156 207  30   5 211   3 183
196
## [103] 157  23 138 187 285  83  53 216  32 109  79 163 248 266 267  25
202
## [120] 103  94 127 205  60 206 132 199   9 117 227  43  74 227 267 290
233
## [137]  53  57   5 241 211  71 159 253  77  38  89 250  98 158  60 126
201
## [154] 243  81  97  53 125  61 120 168 115  54 118 250 113 191 238 125
158
## [171] 285  14 163 225 164  65 125  62 173  56 146 253 288 159 288 147
252
## [188]  31 122 113 247 292 216  21  49 127  31 261  20 117  51 184 127
168
## [205] 224 158  34 168  94 111 220  91 288 236  51 248 190  32 248  17
147
## [222]  70 255 246 255 212 286  85  52  64 203 117  58 185 208 267 111
9
## [239] 235  67 136 173  50 185  97 264 181  86 289 283 235 158 278 207
176
## [256]  63 152 289  93  12  25  60 140  18 259 178  49 242 262  25  97
11
## [273] 240 184 198  32 205 137 216 174 209 188 239  98  45 168  74 142
31
## [290] 163 140 126 212
##
```

```
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 34 0 (0.883959044 0.116040956)
##   2) Salary< 30.4 257  1 0 (0.996108949 0.003891051)
##     4) Distance< 17.95 256  0 0 (1.000000000 0.000000000) *
##     5) Distance>=17.95 1  0 1 (0.000000000 1.000000000) *
##   3) Salary>=30.4 36  3 1 (0.083333333 0.916666667)
##     6) Distance< 12.5 3  0 0 (1.000000000 0.000000000) *
##     7) Distance>=12.5 33  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[4]]
## $bindx
##   [1] 268  73   8 286 158 221  58 275  54 175 219 151 138  25 156 122
279
##  [18]  44 285 224 180  50 153  16 237 190 269 130   8 114  25 282  32
288
##  [35] 181 229  10 118 293 150 102 106  79 242 270 223 272  46  80 209
229
##  [52]  99 243 210  47 273 260 165  87 205  36 291 105 134 207   7 147
158
##  [69] 260 125 113 118 282 269 118 145 181  42 169  95 227 270 264  19
80
##  [86] 111  96  90 153 118 238 187 100  32  31  27 256  44 235 226 273
206
## [103] 108   4 291  41 117  27 178  29 156 203 253 189 139 270  12 116
131
## [120]  30 281 266  61  39  43 114  61 120 150 162 163 158 143  53 230
264
## [137] 219 245 113 147 202 198  81 253 132  42 183 264 249  67  44  82
134
## [154] 169 221 124 265 236 172 151 276 189 161 144 213 236 149 129 241
197
## [171] 241 237  18  82 176 176 224 173  42 247   2 187  63  28 149   1
240
## [188] 200  56 286 250 158 109 252  93 233 243 187 270 205 192 262  59
213
## [205] 182  79  66  16 183  88 241 244 276 275 130 236  76  72 121 170
63
## [222] 194  61 110  91  90 187  23 226 185  31   5 110  72 135 101 216
163
## [239]  12 138  69 203 274 268 290  85 148 242 220 109  37 231  68  10
280
## [256] 285 162 267  27 265 272  38  48  84 134  25  97 187  44  94 290
```

```
170
## [273] 100  46 224 184  63  46   9 237 190 129 115  84 256  15 187 207
235
## [290] 238 266 117  27
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 28 0 (0.904436860 0.095563140)
##   2) Salary< 31.45 261  1 0 (0.996168582 0.003831418) *
##   3) Salary>=31.45 32  5 1 (0.156250000 0.843750000)
##     6) Distance< 13.15 5  0 0 (1.000000000 0.000000000) *
##     7) Distance>=13.15 27  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[5]]
## $bindx
##   [1]  60  58  69 206 117 117 124 125 217  31  61  49  65 158 129 102
216
##  [18]  89 222 167 262 188 165 239 254 223 270 239  65   3 137 210  88
129
##  [35]  46 260  61 259 116  28 103 191 145 104  76 130  70 181 113  36
207
##  [52] 180  15 130  18 131 166 162 193 183 227 180 114  46  58 230  66
119
##  [69] 115  46  31  20  47  58  51 225 125 198  89 147 282  96 252 285
88
##  [86] 161 253 188 212 198 110  33 277 182 248  53 198 175 235 192 173
178
## [103] 260  87 225 237  44  28 212  98 241 153 121 163  60 141 169 174
226
## [120] 256  82 282 254 245  97 107 248  58 204 263 145 138 166 240 269
213
## [137]  77  70  83 164 133 148  54 170 224 153  51  24 137 236 237 227
161
## [154]  86 211 237 201 282  57 119  66   4 272 169  19 211 286 200 158
141
## [171] 238  31 193 194 289 221 227 235  77 176  53  76 114  76 114  13
187
## [188]  40 279  85 167 137 224  61  22 119 162 252   6   7 197  29 200
241
## [205] 137  69  81 214   8  37 284   2  46   8 241  55 251   5 284 186
176
## [222] 228 151 167  93  34 291 230 245 177 123 275  77  21 284  51  62
7
```

```
## [239] 171  22 191  47  84 192 252 255  67  31  42  35  80  39 134 198
36
## [256] 207 112 142 147 209 101 144   7 128 114 216  46  62  60  63  74
49
## [273] 283 126  66 130 205  26 280 229 108 255 247 123  17 235 214  93
188
## [290]  24 113  42  95
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 26 0 (0.9112628 0.0887372)
##   2) Salary< 30.4 263  0 0 (1.0000000 0.0000000) *
##   3) Salary>=30.4 30  4 1 (0.1333333 0.8666667)
##     6) Distance< 13.15 4  0 0 (1.0000000 0.0000000) *
##     7) Distance>=13.15 26  0 1 (0.0000000 1.0000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[6]]
## $bindx
##   [1] 215 168 156 127  21 112 170  72 154   7 190  50  17 286  81  55
283
##  [18] 206  38 135 156  15 186 276  27 218   9  87 258   8 184 246 258
233
##  [35] 102  40  36 191  27 112  25  75   5 215 242  64  70  50  23  55
256
##  [52]   9   7 199  38  18 219  99 232 195 264 143 156 203  72 272  81
182
##  [69]  99  68  94 103  18  33 177 102 237 187 149 235 257  34 149 133
97
##  [86] 224  49 134 106 233 135 157 113  60 126 204 170 160 193 152 250
95
## [103]  61 282 246 153 133 253  35 253 195 101 197 225  18  91 122  59
234
## [120]  48 255  39  62 104 111  56   8 271 249  44  63 271  28 191 163
98
## [137]  45 152 121  39 122 174 210  72   1 167  68 159   7  69  12  73
110
## [154] 289  13 141  16  86 255  38 210 128 177  11  11 149 183 290 127
122
## [171]  95 176 259 193 246 147 187  45  44 110  26 191   6  50 191 289
132
## [188]  13  90 150  45 249  17 128  49  20  44 117 264 205  81  73  59
251
## [205]  87 281 117  46 149 162 194 212  79  93  98   5 237 175  96 201
```

```
101
## [222] 285 222 235  66  93 253 182 162 229  27 264  26 192 129  60 155
133
## [239] 210 293 244 184  22  64 287 257  29  38 130 166 162 292   5 131
218
## [256] 124  43  18 147 260  65 249 195 260 159 128 217 118 191 171  95
244
## [273]  39 169  63  85 196  31 264  41 200 146 237 274  54 104 186  74
70
## [290] 192  19 214  56
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 32 0 (0.890784983 0.109215017)
##    2) Salary< 37.5 267  6 0 (0.977528090 0.022471910)
##      4) Distance< 17.95 263  2 0 (0.992395437 0.007604563)
##        8) Distance< 16.45 254  0 0 (1.000000000 0.000000000) *
##        9) Distance>=16.45 9  2 0 (0.777777778 0.222222222)
##         18) Age< 30.5 7  0 0 (1.000000000 0.000000000) *
##         19) Age>=30.5 2  0 1 (0.000000000 1.000000000) *
##      5) Distance>=17.95 4  0 1 (0.000000000 1.000000000) *
##    3) Salary>=37.5 26  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[7]]
## $bindx
##   [1] 253 133  36  79 155  34   8 146 204 202  38 148 168 114 204  40
247
##  [18]  50 162  94 120 276 190 253  22 149 196 288  25  24 102  67 143
32
##  [35] 136  79 217 221 173 120 272  23  57  81 131  64 197  16  54 110
109
##  [52] 147 257  47  51 197 149 185 165  93 182 214  25  27 204  12 290
52
##  [69] 226 165 122 125 237 151  21 293 171  56  34 215 277  28 244 215
205
##  [86] 289 237  61 166 133 272 150  67 172  91 136 163  67  51 246 262
1
## [103] 112  71 105 263 236 259 110  17 113 193 135 177 250 251   6 202
70
## [120] 127  42  60 168 259 188 272 245 147  53 234 184  18 125  39   6
98
## [137] 175 192  12 150 235  19  65  52  18  79 219 203 224 199  54 148
193
```

```
## [154] 285 208 228   6  87 244 172  79  47  37   9 285   6  39  23 176
139
## [171] 168 195 222 207  81 253  27 286 172 181 203 153 182 174 254 183
87
## [188] 159 205  14 118  41  81 205 186 166   4   5  46  45 255  21 112
93
## [205] 183  53 138 237 279 169 283 125 109  39 287 257 254  60 189 114
176
## [222] 211 227 102 146  87 257 206 241 201  78  59  82 228 290 245 272
225
## [239]  13  54 211 281 148 143  74 177 248 145 144  15 246 185 193  37
231
## [256] 266 269 170 159  57  26 156 291   9 261 187 253  91   7 218   5
212
## [273] 278  29 247 140  27 258 111  30 188 135  64 179  53 113 151  50
126
## [290] 224  47 233 245
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 32 0 (0.890784983 0.109215017)
##   2) Salary< 30.4 258  1 0 (0.996124031 0.003875969) *
##   3) Salary>=30.4 35  4 1 (0.114285714 0.885714286)
##     6) Distance< 13.5 4  0 0 (1.000000000 0.000000000) *
##     7) Distance>=13.5 31  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[8]]
## $bindx
##   [1] 204 281  52 197  71   5  58 257  92 262 274 275  46  48  57  75
139
##  [18]  33 205 278  67 173 234 104  57  88 138 190 156   6 205 223 245
7
##  [35] 263 280 238 201 198 177 274 280  93 115 259 198 281 250 210 281
288
##  [52]  32   3 169 144  19  30 206  84 105 121 185  57 287 196  32  19
190
##  [69] 111 233 261 153 117 290  25 206 174  93  32 235 263 258 278 276
164
##  [86] 269 169  83  51  25 273 155 215  70 110  45 166  58 269  14 219
270
## [103] 290 279  27 261 148 130 128  35 266  16 289 137 213  37 169  43
90
## [120]  12 143 282 207 281 132 128 210 136  36  22 159 257 112 246  91
```

```
80
## [137] 270 141 139 225 152  29 271 210 195  16 170  55 134  90   1 175
123
## [154]  67 189 237 273 224 179  90 221 146  53 193 170 129  16  58 221
11
## [171] 227  14 144 199  30 161 280 242 147 213 234 103 146 246 164 286
49
## [188] 119  55 238  95 185  11 111 201 246  74  61  67  14  78 214  91
282
## [205] 136 283 237 275 177  34  44 216 161 138 112 249 245  94  39  37
79
## [222]  62 250  75  10 264  24 111 105  25   7 151 279  99 159 263  32
56
## [239] 191 186 173 133 200   7  33 122 205   9 120  20  14 157 181  88
16
## [256] 221  73 283 120 135 208  10 159 184  76 165  83 250  72 242   4
292
## [273] 110  38  43 220 202 111 161  15 230 167 118  12 197 181  88  76
255
## [290] 201 122 233   9
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 22 0 (0.924914676 0.075085324)
##    2) Salary< 37.5 274  3 0 (0.989051095 0.010948905)
##      4) Distance< 17.2 272  1 0 (0.996323529 0.003676471)
##        8) Salary< 31.35 268  0 0 (1.000000000 0.000000000) *
##        9) Salary>=31.35 4  1 0 (0.750000000 0.250000000)
##         18) Age>=32.5 3  0 0 (1.000000000 0.000000000) *
##         19) Age< 32.5 1  0 1 (0.000000000 1.000000000) *
##      5) Distance>=17.2 2  0 1 (0.000000000 1.000000000) *
##    3) Salary>=37.5 19  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[9]]
## $bindx
##   [1]  80 236 169 116  81 161  20 195 266 216  75 131  83 173  90  18
265
##  [18]  90 287 283  44  51 105 131 205  18 253 256 249  58 136 187 218
182
##  [35]   2 292  42  20  81 218 232 130   8  46  55 157  54  66 195 240
60
##  [52] 120 191 155  64 196 192 117 205  81 176  31  50 267 244  75 148
197
```

```
##  [69] 114 252 278  83 259 255 148 103  28 204 147  50 113 244 203   9
219
##  [86] 114  68  79 174  58 282 213   6  31 130  99 181 118 123 207  70
17
## [103] 242 176 128 283 218  80 179 144 260  15 101 148 213  75  25 276
97
## [120]  35 143  30 276   8 140   5 236 262   3  11 253  89 123 260 214
143
## [137] 232 180 254  93 163 277 289  39 185 272  81 102 248  47 185  57
47
## [154] 150 148  39  73 286 252 147 122 213  51 103 150  55 190  70 228
178
## [171]  91  66 182 140 277 273 255  52 249 170 144 162 138 145 102  29
207
## [188] 146 253 293  26   1  42   4 205 238 196 181   4 127  77   8 231
60
## [205] 225 133   3  19 116 182  40 291  72 210 163   9 115  65  11 135
166
## [222] 210  41 292 111  10 263 290 249  19  30  83 277 267 161 255 241
241
## [239]  76  84  79 273 247 223 160  89  87  24  84  60  22 220 152 120
287
## [256] 169 110   9 114 262  90  68 240 112   2 161  76 221 254  47 124
192
## [273] 249 208 205 285 112 154 239  15  91 141  85 281 176 247 166 154
103
## [290] 110 249 147 113
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 35 0 (0.88054608 0.11945392)
##   2) Salary< 31.45 255  0 0 (1.00000000 0.00000000) *
##   3) Salary>=31.45 38  3 1 (0.07894737 0.92105263)
##     6) Distance< 13.15 3  0 0 (1.00000000 0.00000000) *
##     7) Distance>=13.15 35  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[10]]
## $bindx
##   [1] 220 121  74  86 109 279 229  27 132  63  47 171 222 118 226  69
103
##  [18] 176  85   3  37   4  11  55  96 124 162 184  37 247 248 109  51
128
##  [35] 158  72 242 100  94 256  26 209  44 241 157  36   2 216  10 151
```

```
78
##  [52]   8  95 162 288 292  19 278 278 166 142 272 269 210  55 105 131
219
##  [69]  16  81 171 140 205  72  71 286 138 109 164  27  19 225  43 172
161
##  [86]  66   3 242  12 120 152   8  81 111 237 134 237  28 283 257  82
141
## [103] 154 166 103 275 101 152 205 174  94 145 209 156  74 233  47  31
262
## [120] 143  31 177 179 265  16 117 256  50  26 165 173  34  65  99 206
40
## [137] 252 264  93 210 201 211 269 260 160 237 127 156 124 243  25 190
279
## [154] 257  22 257 260  96  65  24  19  61 159 240 132 162 134 160 260
189
## [171]  86 113  86  86  98 167 155 126 201  23 290  81  65 108 193  74
167
## [188] 249 260 232 107 265 164 184 233 119 202 156 182 126   2 173 137
259
## [205]  38  24 206 246 247 184 258 110 144 160 205 168 106 132 185 199
47
## [222] 241 231 286  14 238 270 234  39 280  49 220  19 277 185  81  35
75
## [239] 292  92  78 265  57 292  30  49 149 131 157 128  92 235  33  89
280
## [256] 132 156 197  34 175  84 179 225 204 119 219 146  36 266 253  36
33
## [273]  34 224 284  78 252 137  59 106 136 275 225 120  16  26 260  12
21
## [290] 284 190 104 151
## 
## $btree
## n= 293 
## 
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
## 
##  1) root 293 18 0 (0.93856655 0.06143345)  
##    2) Salary< 37.5 279  4 0 (0.98566308 0.01433692)  
##      4) Distance< 17.25 270  0 0 (1.00000000 0.00000000) *
##      5) Distance>=17.25 9  4 0 (0.55555556 0.44444444)  
##       10) Age< 30 5  0 0 (1.00000000 0.00000000) *
##       11) Age>=30 4  0 1 (0.00000000 1.00000000) *
##    3) Salary>=37.5 14  0 1 (0.00000000 1.00000000) *
## 
## attr(,"class")
##    class 
## "sclass" 
## 
## $mtrees[[11]]
## $bindx
```

```
##   [1] 122  51 246  31  16 158 267 216 275 198 120 271 149 167 278 195
208
##  [18] 132 117 190 106 192 290  56 291 107 258 241 111 243  95  37 225
259
##  [35]  93 231 193  32 161  35  25 103  29 261   1 113 122 209 270   2
261
##  [52]  31 117 127  39 263 215  15  68  24  80 281 216 104 280  33 149
53
##  [69] 218 165 260 225 174 182 255 108   7 221 142  25 104  29 274  84
143
##  [86]  70 131 170  67 246  77 161 262 245 118  20  50 139 211 116 188
54
## [103]  34 173  43 148 110 248  30  23 230  16  20 220 134  55 288 273
74
## [120]  97  16 132 129  51 183 250  42   7 116  70 130 143 202 156 202
293
## [137]  50  41  79  15 157 134 185  58  20  21 238  28  50 217 134  17
257
## [154]  97 128 177  25  20 277 113   2 111  16 287 153 170 145 273  91
104
## [171]  44 244  64 277 127 135 293  38  56  69  57  14 272 107 182 129
213
## [188]  24 124 262 286  27 130  77 160 280  66 207  63 269 224  85 201
185
## [205]  57 162 175 242 223  35 229 112 119  36 208 168 234 279 258  72
194
## [222]  59 187  41 102  98  24 155 135 273 252  84 226 233 249 142  62
105
## [239] 245 189 250 185 110  99  23 143 293 203  69  47  49  26  60 167
217
## [256]  89 238  73 123  71  35 158 266 205  99 125 131 262 217  43  59
280
## [273] 173  39 245 165 154 251  24 207 206 245 154 127  45 194 239  82
98
## [290] 272  16  85  11
## 
## $btree
## n= 293 
## 
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
## 
##  1) root 293 30 0 (0.897610922 0.102389078)  
##    2) Salary< 29.85 260  2 0 (0.992307692 0.007692308)  
##      4) Distance< 17.95 255  0 0 (1.000000000 0.000000000) *
##      5) Distance>=17.95 5  2 0 (0.600000000 0.400000000)  
##       10) Age< 31 3  0 0 (1.000000000 0.000000000) *
##       11) Age>=31 2  0 1 (0.000000000 1.000000000) *
##    3) Salary>=29.85 33  5 1 (0.151515152 0.848484848)  
##      6) Distance< 13.5 5  0 0 (1.000000000 0.000000000) *
##      7) Distance>=13.5 28  0 1 (0.000000000 1.000000000) *
```

```
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[12]]
## $bindx
##   [1]  53  81 122 112   5 141  90  72 151   1 130 212 137 166 250  41
93
##  [18]  47 285 149 177  92 101 282 121 215  45 209  59 160 274 113 104
94
##  [35] 260  50 241 111 207 237   3 236 149  10 217  15 121 145  98  63
274
##  [52]  76  76 145  31 286 121  95  19 248 265 200 110  60 249 142 265
63
##  [69] 158 134 194 184  35  79  34 200 273 273 178 263 286 144 268 195
104
##  [86]  50 122 256 185 124 138 235  77 185 216 125 238 237  74 150 278
225
## [103]  56   9 116  91 248  17 217 124 201 128 275 224 275 269  41 289
252
## [120]  33 110 165  33   3  78  83 266 209  73  49 293 160  63  83 137
102
## [137]  44 125  84 267 145 282  25  11 100 135  63 145 175 117 209 114
198
## [154] 103 237 230 182 132  21   8 171  18 287 105  49 182 247 134 172
33
## [171] 187   3 125  82 157 177  82 240 141  63 112 212  23 293  32  34
159
## [188]  17 128 141 120  94  81 244   7 144 272  94 208  12   2 124 236
284
## [205]  54  41 215 116 189 233 218  39 222 204 153 174 247 162 233 274
167
## [222]  90  15 214 129 263 212 268 258  11 158 284 287  52 201  70  52
6
## [239] 127 106 162 224 220 141  61  89  36  37 236 250 268  98  71  37
264
## [256] 277 285 164 218 166 148  76 170  32  11   5 225 244 228  20  21
280
## [273] 214 255 291  88 291  24 212 278 240 268 192 153  23 194  13 192
2
## [290] 279  30  75 139
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 26 0 (0.911262799 0.088737201)
##    2) Salary< 29.8 265  1 0 (0.996226415 0.003773585)
```

```
##      4) Age< 32.5 261  0 0 (1.000000000 0.000000000) *
##      5) Age>=32.5 4  1 0 (0.750000000 0.250000000)
##       10) Work.Exp>=10.5 3  0 0 (1.000000000 0.000000000) *
##       11) Work.Exp< 10.5 1  0 1 (0.000000000 1.000000000) *
##    3) Salary>=29.8 28  3 1 (0.107142857 0.892857143)
##      6) Distance< 13.15 3  0 0 (1.000000000 0.000000000) *
##      7) Distance>=13.15 25  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[13]]
## $bindx
##   [1] 146 281 217 258  21 292  46 293 222  62 220 167 160 154  56  87
285
##  [18] 196 165 178 135  27 231  95 233 109  46 189 156  99 141  30 218
51
##  [35] 127  67 202 261  47 146 105 291 208 178   3 150 153 213 209  94
122
##  [52] 291   7 238 293 252  56 264 238  19 217  36 260 148 115 256 217
29
##  [69] 166 118  73  61 184 173 178 153  17 197  57 227 232 177 256  14
102
##  [86] 114   6  21  20 106 275 141 150 161 189 143   4 240 282 193   3
278
## [103] 176  27  56  95 105 176 284  52  26 260  48  19 157 147 270 285
60
## [120] 280 149 268 218  14  97 202 142  48 164 198 231 102  85 217  16
70
## [137]  56  80  51 270  68 171 288 271 178 165  90 228 143  98  25 212
114
## [154] 269 287 286 280 121 170 211 268  86 198 124 270  36 193  38 202
112
## [171]  36 138 128 211 202 114  91 194 138 273 105 105 230 215  71 243
160
## [188] 237 172 181 190   5  19 268 190 207 141   8 274  10 202 187 259
262
## [205] 133  74 287  21  17 209 134 225 155 283 211  53  83 173  66 226
284
## [222] 129 288 155  59  76 122 149 180 232  64  28  78 115 191 114  53
270
## [239]  10 142 143 283  54 106  94  29 220  46 116 169 184 224 107 107
139
## [256]  70 237 161 112  83 148  35   5 140  20 276 130  25 144  30 130
48
## [273]   1 209 212  76 111   2 127 241 200 145 199 227 218 154 254 120
36
## [290] 199  91 101 253
##
## $btree
```

```
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 36 0 (0.87713311 0.12286689)
##    2) Salary< 30.4 259  4 0 (0.98455598 0.01544402)
##      4) Distance< 17.55 254  0 0 (1.00000000 0.00000000) *
##      5) Distance>=17.55 5  1 1 (0.20000000 0.80000000)
##       10) Age< 29.5 1  0 0 (1.00000000 0.00000000) *
##       11) Age>=29.5 4  0 1 (0.00000000 1.00000000) *
##    3) Salary>=30.4 34  2 1 (0.05882353 0.94117647)
##      6) Distance< 12.5 2  0 0 (1.00000000 0.00000000) *
##      7) Distance>=12.5 32  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[14]]
## $bindx
##   [1] 110 183 105 222  61 119 228 215   5 127 157 190 282 253 233  54
36
##  [18] 118 182  89  78  23 143  45 115 278 283 212 282 207  86 220 139
284
##  [35]  55 272 247 258 275  10 171  39  13 236 137  95  14 235   9 210
287
##  [52] 198 202 275 172 204 171 262 268  60 156 284   3 159  59  94  79
107
##  [69] 100 151 150   5 268  14 158 133  38 205  13 223  51 179 226 210
30
##  [86]  80 165 138  58 226  88 241 209  43 163 193 263  95  76  26 254
108
## [103] 146  33 254 256  30 251  23 137 251  98  65  14 276 263 206 189
30
## [120] 221  93  15 226  34  11 239  89 133  39  53 109  65  30 157 236
56
## [137] 167 187  33 148 183 268  68 235 164 170 274 149 228 186 226 283
213
## [154]  76 271  33 282 183 168  30 110 145 183  93 280 208 150   9 233
248
## [171]  44 167  51 218  25 219  55 109  98 209 279 127 205  22 191 157
237
## [188]  92  75 214  22  37  81 263  41 280 138  21  31  38  52 138  33
218
## [205]  75 175 133 255 118 164 114 136  34 278 165 270  24 246  82 184
38
## [222] 140 144 107 106  41  51 260  12  94 161 231 225  19  75 247 115
2
## [239]   4 117 231 240 211 232  53  93 111 175 136 286  17  63 206 104
262
```

```
## [256]  32 234   8  68 236  33 159 221 222 275 186 160 132  14  33  72
36
## [273]  54 215 155   6 248  62   8 152 292 198 277  16 146 150  59 131
206
## [290] 180 291  34  47
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 33 0 (0.887372014 0.112627986)
##   2) Salary< 30.85 259  1 0 (0.996138996 0.003861004) *
##   3) Salary>=30.85 34  2 1 (0.058823529 0.941176471)
##     6) Distance< 12.25 2  0 0 (1.000000000 0.000000000) *
##     7) Distance>=12.25 32  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[15]]
## $bindx
##   [1] 156 168 178  71 189   3 128  41 134  33  42 216  15  85 235 189
281
##  [18] 282 233  61  51  79 102   8 174 205  75  39  27  56 223   8 150
78
##  [35] 150   4  69 287 100  34 151 249 263 280  57 200 113  26   5 107
74
##  [52] 107 134 210 260 103  28 238  18 100 106  74  35  11 189  58 167
30
##  [69] 129  30 187 137 121  29  10  43 128 174  20 230 259 288  78 142
219
##  [86] 112   1 130 133 253 218 103  55  47 248 274 234 194 167 244 153
117
## [103] 213 113 219 182 140 176  47 213 110 146  90 163  37 286 158 224
289
## [120] 163 229 222 243 260 282 197 217 114 233   6 259  43 201  62 279
288
## [137] 183 167 118 197  75 279  58 188 108 172 265 128  81 267 170 212
139
## [154] 284 182 194  74 215  67 100 218  62  50  39  90  36 224  23  63
9
## [171] 171   9 288 178 284  12  63  90  31 219 210  81 251  41 233  15
161
## [188] 135 261 175 216 251 165 232  70 121 146 251 172 164 151 291 189
211
## [205]  72  28 109  81 127 275  24  51 270  47 155 231  98 292 129 215
3
## [222] 258  10 215 107 218 261 202  57  98 285  80 197 189  52 105 105
```

```
138
## [239] 189  56 191  15 205 164  18 105 290  29 164  14 182 256  23 105
90
## [256]   4 198 139  44  18 144  61 203 275  74 289 253  25 140 246 216
12
## [273] 156  56 263 223 106   4 136 128 153 105 229 154 166 199 219 178
232
## [290]  46 197 189 224
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 33 0 (0.88737201 0.11262799)
##   2) Salary< 30.4 261  3 0 (0.98850575 0.01149425)
##     4) Distance< 17.95 258  0 0 (1.00000000 0.00000000) *
##     5) Distance>=17.95 3  0 1 (0.00000000 1.00000000) *
##   3) Salary>=30.4 32  2 1 (0.06250000 0.93750000)
##     6) Distance< 12.5 2  0 0 (1.00000000 0.00000000) *
##     7) Distance>=12.5 30  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[16]]
## $bindx
##   [1]  83 278 196  32 154  68 214 152 255  58 266 221 114  81  40 291
224
##  [18]  97  23  32 214 129  10  94 176 202  98 122 276  28  71 270 287
121
##  [35]  98 148 260 206  36 283 200  28 153  71  84  10 246 126 255 138
97
##  [52] 212  58 230  71 285  69 267 158 280 231  39 158 139 201 134  51
71
##  [69]  63 169   4  34 188  35 221  42 292 199  80 117 160  14 166 205
273
##  [86] 245  95  22 136 274 227 243 249  31 220 105  16 155 204 149 272
105
## [103]  66 189 293  30  24 289 209  37   4 244 141 255 171  93 189 153
48
## [120] 263  70 162  68  63  33 194 212 177  96  85 126 198 211 212  59
48
## [137] 163 156 118 207 255 240 233 246 267 280 243  77 277  18 202 219
151
## [154]  99 210 147 254 113  96 278 259 135 138  41 226 215 244  49 207
114
## [171] 265  70  28  66  38 118  64 213  48 228 147 229 157  53 210   4
49
```

```
## [188] 119 242  79 199 290 214  85 162 139 126 192 221  85 181 279  70
275
## [205]  23 216 169 124  98 290  72   6   4 209 260 139 175  47 258 210
196
## [222] 237 285 116 222 173 227  80 221 266 290 244 264 152 258 182  13
209
## [239] 280 249  74 124  68 171  99 122 252  36 226   7 232 168 262 129
127
## [256] 128 260 237 262 148  31 245   7 241 104 175 180  73 261 111 158
275
## [273]  50 125  69 244 118 264  56 160 166 143 146  66 191  67 125 205
226
## [290]  22  77  81 217
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 28 0 (0.904436860 0.095563140)
##   2) Salary< 30.4 260  1 0 (0.996153846 0.003846154) *
##   3) Salary>=30.4 33  6 1 (0.181818182 0.818181818)
##     6) Distance< 13.15 6  0 0 (1.000000000 0.000000000) *
##     7) Distance>=13.15 27  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[17]]
## $bindx
##   [1] 249 180 185 243  48 114  10 221  27  63  27 189   2 229 123  55
219
##  [18]   6 198 146 115 124 220   6 192 229 256 113  78  27 165  55  27
165
##  [35] 102 228 239  65 124 168 185  38 206 207  65 149  75 289 266 231
210
##  [52] 209 211 104 148  58 246  32 170 146 222 138  91 180  20 260 214
19
##  [69] 278 169  39 139  51 285 134  41  78 242  65 285  65 133 131 195
235
##  [86] 239 180  95 160 187  48   5 146 166 263 107 266 170 268 216  48
174
## [103] 276 132 172 172 110 159 154  78 259 213 199  60 210  79  29 230
202
## [120]  10  48 103  88 178 226  76 130  90 202  83 132 269 217  85 115
60
## [137]  70  31  57 128 188  61 199 172 100 155 152 195  55  78 134 293
167
## [154] 278 155 176 108 258  33 149 123 247  91 247 189  43 123  17 151
```

```
190
## [171] 249 152 284 141  67 162  72  29  28  91  46  52 236  85 192 123
177
## [188]   8 156 230 178  89 225  57  14 272  84  30 144  43  67 248 277
254
## [205]  62 150  38 161 287 115 205 248 111 212 181 218  69  46 145 234
145
## [222] 273 126 211 276 209 156 241 110 132 165  41 170 248  68 239 268
108
## [239] 146 264 179  72  49 230 253  34  65 186 269  24  40  49  14 291
117
## [256] 258  59 199  14  49  15 141 287 190  48 159 174  50 210 212  81
240
## [273] 274 151 132 212 196 149 104  80 278 231 171 150 133  70 149 133
42
## [290] 112  30  28  74
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 25 0 (0.91467577 0.08532423)
##    2) Age< 35.5 272  4 0 (0.98529412 0.01470588)
##      4) Salary< 30.4 263  0 0 (1.00000000 0.00000000) *
##      5) Salary>=30.4 9  4 0 (0.55555556 0.44444444)
##       10) Age>=32.5 5  0 0 (1.00000000 0.00000000) *
##       11) Age< 32.5 4  0 1 (0.00000000 1.00000000) *
##    3) Age>=35.5 21  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[18]]
## $bindx
##   [1]  57 253   1 270  78 115 192 241 121   8 101 180  33  78   9  61
137
##  [18]  59 247 288 227 108 146 244 126  98 238 127 168 194 232 260 241
253
##  [35] 156 213  14 270  46 212 151 229  44  79  90  69 192  11  29 203
110
##  [52] 222 246 101 290 102  87 185 201 247  37  63 179  80  87  26 136
179
##  [69]  11  14 101  66 202 235 153 147  27  70 291 114 170   8  81  79
225
##  [86]  81 214 261  53 257 264  49 113 107 243  67 214 173 248 284  50
48
## [103]  20 196 220 220 126 181 224 116  96 108  30  35  66  33 182  70
220
```

```
## [120]  54 255   6 135 122  18 147 143 227  57 142  42 147 224 239 240
114
## [137]  18 246 179  72 195 177 159  15 150 109 123 231 174 187 237  40
61
## [154] 205  71  48 209 168 132   8 254  98 172  67 144  83  37  84  32
100
## [171]  89 147 214  18  34 167 124   8 102 136 230  97 254 239 147  38
163
## [188] 274 202  79 163 173 245 229 270  47 150 255  12  49  43 126  94
246
## [205] 232  71 175 213 253 119 132 110  82 265 191  50  31 115  50   2
11
## [222]  84  19 283 134 107 113 209  70   8 151 291 186 229 160  27 168
264
## [239]  73  24 177 216 288 207 244  89 155 262  18 202 109 191  97 286
136
## [256] 175 129   4 144  55 187 281  98  45 148 267  94   6  43 276  61
241
## [273] 195 279  50 235 267 176   1  31  25  25 163 256 236 277 244 102
289
## [290]  64 156  68 227
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 29 0 (0.90102389 0.09897611)
##   2) Salary< 31.45 262  0 0 (1.00000000 0.00000000) *
##   3) Salary>=31.45 31  2 1 (0.06451613 0.93548387)
##     6) Distance< 13.5 2  0 0 (1.00000000 0.00000000) *
##     7) Distance>=13.5 29  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[19]]
## $bindx
##   [1]  83 221 289  72  77  60 213  50 120  82 289 200 117  93 122 136
35
##  [18] 138  78 180 156 257 264  67 208 275 133 124 253 178  68 286 236
292
##  [35] 255  12 284 283 240  42 134 240 134  87 141 115  25  83 257 138
53
##  [52] 106 175  23 158 281 256  66  20 121  34 115 182 256   3  95 133
118
##  [69] 236  92 113  20 253 242 154  92  28  20 166 130 239  15 113 227
23
##  [86]  98  75 278 285 200 113   7 236 120  80  15 128 181  27 134 230
```

```
255
## [103] 117   1  37 152 147 156 256 267  11 209 146 127  16  90   4 178
203
## [120] 151 225  61 151 184 254  36  47 175 204 165  13 265 235 226 127
98
## [137] 167 162 211 261 255   6 182 132  99  20 253 110  89  67 289 145
173
## [154]  56 126 273  18 180 193 210  62 114 113 270 280  37  13 112 263
216
## [171] 217 111  92 168 259 280 268  65 214 242 164 190 203 127 283  30
49
## [188] 280  33 161  28 251 206 116 253 176  19 242 244 199  75 143  15
189
## [205] 284 141 178  84  42 236 226 141 220 135 115 281 190 133 262  67
179
## [222] 236 245 220 206 145 242  38  72   2  21 255 100 235   5  41  51
15
## [239]  30 172 276 189  50 226 175 229 276 122  71 224 290 150 118 228
214
## [256]  69 255 200  71 174 221  33  74  36 260  60 193 165  89 226 196
232
## [273]  12 261 165  82 118  59 173 193   9  20 112  37  59  76  70 254
111
## [290] 253 143 184 111
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 293 29 0 (0.90102389 0.09897611)
##    2) Salary< 37.5 267  3 0 (0.98876404 0.01123596)
##      4) Age< 31.5 256  0 0 (1.00000000 0.00000000) *
##      5) Age>=31.5 11  3 0 (0.72727273 0.27272727)
##       10) Distance< 14.7 8  0 0 (1.00000000 0.00000000) *
##       11) Distance>=14.7 3  0 1 (0.00000000 1.00000000) *
##    3) Salary>=37.5 26  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[20]]
## $bindx
##   [1]  34 219 213 217 113 277 213  45 292 247 137 229 254  18  21 187
122
##  [18]  26 114 270 275 256 203 253 228 192  64  46  12  79 137  71  42
111
##  [35]  75  77   7 156  80 173  88  55  36 224  83 105 254 197 238 263
287
```

```
##  [52]  66 277 208 212 153 153  29 271 144 115 128  72  17 256 272 240
257
##  [69]  48 177  56 150  45  84 207  80 165  77 133 199 195 176 140 184
39
##  [86] 195 241 140  42 193  90 190 281 131 137 283 141  58   1  14  20
149
## [103] 281   6  75  60  80 249 133 203 222  76  89 221   6 248   7 236
84
## [120] 209 217 179  70 269 274 288 280 133  25  82 173  86 154 149 141
220
## [137] 181 165 206 180 144 269  40 289 135 177 157 149 212 201 181 265
276
## [154] 271 279 170  69 158  97 286 105 128 108 195 138  58 239  72   9
87
## [171] 245 230 198  83  77 121 128  94  28 185 209 202 190  41 173 122
214
## [188] 198 133  28 218 207 107 128 161 150 139 134  84  11  99 225 160
7
## [205] 151 283 291  38 116  43 279 170 172  37 139 288 293  41 144  55
41
## [222] 275  67 290  58 290 290  35 143 196  98 241  88   9 161 203  37
197
## [239] 213 242 246 129  40 141 107 208 246  46 165  90 187 136 117 275
44
## [256]   4  98 103 160 245  91  93 292 111 143 211  94  61 143  93 109
285
## [273] 157 160 101  56  50 110 122 108  58  33 154  86  21 130  24 218
59
## [290]   3  40  30 252
## 
## $btree
## n= 293 
## 
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
## 
##  1) root 293 26 0 (0.911262799 0.088737201)  
##    2) Salary< 30.4 264  2 0 (0.992424242 0.007575758)  
##      4) Distance< 17.95 260  0 0 (1.000000000 0.000000000) *
##      5) Distance>=17.95 4  2 0 (0.500000000 0.500000000)  
##       10) Age< 31 2  0 0 (1.000000000 0.000000000) *
##       11) Age>=31 2  0 1 (0.000000000 1.000000000) *
##    3) Salary>=30.4 29  5 1 (0.172413793 0.827586207)  
##      6) Distance< 13.5 5  0 0 (1.000000000 0.000000000) *
##      7) Distance>=13.5 24  0 1 (0.000000000 1.000000000) *
## 
## attr(,"class")
##    class 
## "sclass" 
## 
## $mtrees[[21]]
```

```
## $bindx
##   [1]  59 232 146 107 228 202 235 182 242 274 135  98 186 117 201  87
187
##  [18]  49 239  71 239 161 159  76 195 225 130 191 123  90 168  17 211
119
##  [35] 160 274 231 159 240 280 183 117 175 117 154 287 238 186  13 217
159
##  [52] 100 159 199  75  39  76 274 244 237   6  25 264 282  73 229 106
97
##  [69]  41 238  20 273 179 126 135  48  16 288  14 272 121  32  67 212
281
##  [86] 111 169 121  64 270 264 234  69 137 176  35  73  49   2  60 242
189
## [103] 113 228 263 167 241  31 185 156   2  18 290 105 112 278  82 180
138
## [120] 141 140 264 211 159  75  63   5 116  28  51 293  65 247 246 137
23
## [137] 263 247 263  24 259  80  41 117 288 275 141 215 142 241 134 290
128
## [154] 101 114  88 198  94 184  97  85 267 272 101 127  82 218  30 146
181
## [171] 129  33 239  74 176   4 202  89 233 279 176 227 188 105 197 200
211
## [188]  67 165  13  40 191  13  57  25 154 164 253  93  10  87  96 227
113
## [205]  51 197  85   1 128  64  99  77 198 233 102  26 175  40  27 119
75
## [222] 292 166  13 143  63 145 135  81 214 152  89  39 162  44  54 282
259
## [239] 252 173  26 194 289 104 166  63 257 204 245 148 168 290  24 251
280
## [256] 232 142  40 218  35 277 262 224  80 219  73  53 250 145 196  88
176
## [273] 151 204  46 203  87 114 247 208  64 216 155 258 223  86 161  45
38
## [290] 141  49 188  51
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 28 0 (0.90443686 0.09556314)
##   2) Salary< 30.4 260  0 0 (1.00000000 0.00000000) *
##   3) Salary>=30.4 33  5 1 (0.15151515 0.84848485)
##     6) Distance< 13.65 5  0 0 (1.00000000 0.00000000) *
##     7) Distance>=13.65 28  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
```

```
## "sclass"
##
## $mtrees[[22]]
## $bindx
##   [1] 121  80 136 195 189 154  17 117 250 286 241 219 276 139 174   3
246
##  [18] 155 138 287 171  18  10  85 116  34 179 190 137 260 161 144 235
2
##  [35] 127  99  14 272 173  92 178 120 127 236  24  75  81  14  63  34
200
##  [52] 229 184 170 172 238 111 264 270 106 270 205  39 119 133  63  73
216
##  [69] 112 112 197  30 135 215 190  57  17 176 222  59 173 167 213  11
52
##  [86]  38  45  41 189 291 148 192 148  11 127   2 290 103 144  83 250
47
## [103] 165  97 117 164  92  47 153 209   6 193  85 178   8 103 175 170
32
## [120] 140 134 263 151  77 281  30  19 153 124 140  78 222 185  96  16
49
## [137] 225  68 103 132 219  60 127 242  35 281 228   3  14 199  43  31
285
## [154] 274 279  94  94 285 189 133 154 252 261 121 254 137 280 112  69
159
## [171]  60 214  31 239  80  19  81 161  74 160 148  78   5 144 166 218
230
## [188]  60 193  32 155  48 290 207  21 156 180 288 119  44 262 119 289
64
## [205] 252  94 220  44 273 209 163 260  98 128  16 200 236 262 251  43
69
## [222]  75 219  42 133 267  86   6 114 104 136 212 265 197 160 173 283
210
## [239] 127 172 169 249 137 203 258 129 168 281 188 292 224 280 285 229
75
## [256] 248 235  69 284  50 114 173 115  94  69 261 287  28  63 114 140
103
## [273]  86 176  32  84 274 280 240 235 277 163 174  64  90  35 181 208
133
## [290] 209  24 188 119
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 33 0 (0.8873720 0.1126280)
##   2) Salary< 31.45 256  0 0 (1.0000000 0.0000000) *
##   3) Salary>=31.45 37  4 1 (0.1081081 0.8918919)
##     6) Distance< 12.5 4  0 0 (1.0000000 0.0000000) *
##     7) Distance>=12.5 33  0 1 (0.0000000 1.0000000) *
```

```
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[23]]
## $bindx
##   [1]  73  32  83  24 222  89 188  38 193  68 221 147 216 145  32 251
106
##  [18] 120 134 207 195 276 218 146 127 140 139 138 144  83 195 143 225
185
##  [35] 138 188 143  44 110  28 169  84  36 244 243 290  55 282 218 216
222
##  [52] 168  42 125 234  70 224 218 118 172 258   5 159 188 275  71 128
138
##  [69]  53 205  53  33 222  13 147 207 104 158  51   2 198  62  58  34
4
##  [86]  59 245 231 285 100 236 118 236  67 153 111 102  62 214 120  17
119
## [103]  35 205  92  96 249 193  82 127 253  66 142  45 240 210 135 167
290
## [120] 197 194  24 144  22   6  83 166  69 118 279 220 120 167 138  30
101
## [137] 109  67   2  34 216 191  34 181   3 171  53 170  92  39 287 122
121
## [154]  89 154  51 172 163 140 107  23  37  91 199 183 181 103  93 269
133
## [171]  63 108 141 101  47 180 263  12 164  24 124 255 286 121 180 230
48
## [188]  49 155  39 191 218  20 144 148  32  91 177 277 118 134  17 142
62
## [205] 112 191 133  48  62 166 259 246 200 105 120 160 120  38 174  33
187
## [222]  83  26 115 219 116 168 122  78  98 249 267 289  53 114 268  73
203
## [239] 172 121  87  70  30  62 252 219  82 117   9 290  61 213  41 135
27
## [256] 240  15 235 213 107 251 206  46 154 155 240 110 195 258 189 159
127
## [273] 268 120 199  69 269  68 241  93 181  43  24 292 152 132 151  81
29
## [290] 159 139 202 228
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 28 0 (0.90443686 0.09556314)
##   2) Salary< 30.9 265  0 0 (1.00000000 0.00000000) *
```

```
##   3) Salary>=30.9 28  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[24]]
## $bindx
##   [1]  14   2  83  11  15 104 167 265 195  97 173  65 123 177 205 145
15
##  [18] 151 255  99 292 272 224 282 272   6 148 111 274 227 111 142 286
28
##  [35] 130 184   4  58 124  14 178  30 229 142 225 105 225 107  80 218
174
##  [52] 192 230 265 229 165  55 262 270 159  49  93 239 106  54 217 267
28
##  [69] 243 281 198 104 240 272 108 181 277 260  90 271 174 248 114 268
114
##  [86]  38 189  54 151 275 273  53 269 221 211 151  17  17 268 152  83
213
## [103] 188 139 130 280  53  52 269 289 289 230 243 177  26 140  35  55
30
## [120]  71  60 182 202 198 232 168 273  21  55 233  20   7  41 276 141
74
## [137] 185 273 126 185  65 155 184 102 205   2 283 155 204 118 216 164
257
## [154]  22 278 244 246  67  68 148 164 102 287  16  25  20  59  83  33
104
## [171] 121 210 235 110 261   9  80  87 139  97 184 122  14 233  58  63
238
## [188] 177 127  73 274  54 105 139 269  46 231 240 211 275  28 201 184
57
## [205] 267 141 175 154  50  46 214 144 152 155 116   9 130  80 150 285
214
## [222] 179 193 241 283 293 212 106 126 128 129 106  24  29  82 237 119
60
## [239] 203  24 190 146 136  52 177 142 148   4 158 231  90  58  92 169
99
## [256] 164 226  90  43 164  55 285 214  70  75 265  56 284   8  51 100
100
## [273] 233 217 262 222 152 252 155 142 204 156 264 140   4 248   3 124
275
## [290] 161  89 104 157
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 293 24 0 (0.918088737 0.081911263)
```

```
##   2) Salary< 29.85 269  1 0 (0.996282528 0.003717472) *
##   3) Salary>=29.85 24  1 1 (0.041666667 0.958333333)
##     6) Distance< 13.5 1  0 0 (1.000000000 0.000000000) *
##     7) Distance>=13.5 23  0 1 (0.000000000 1.000000000) *
##
## attr(,"class")
##    class
## "sclass"
##
## $mtrees[[25]]
## $bindx
##   [1] 151 188 240   3 129  24 101 244 183 204 218 236  86 291  18  95
161
##  [18] 129 254 138  81  26 290 200 195 253 283  84  74  13 119 103  71
272
##  [35] 171 254 152 207  53  95  86 149 125  99  36 183 255  63  50 155
285
##  [52] 281  76  41  25 227 273 109 220 252 191 180  24 274  73 290 247
246
##  [69]  17 189 101 156 198 191 184 115 230  59 137 150 265 113 253 147
290
##  [86] 196 121 225 278  23  97  85  41 198  99  65 150 201 101 256 262
46
## [103] 114 214 226  95 231  70 119  20 229  11 200 239 118  92   6 250
20
## [120] 112 228  72 216 103  29 240  24  35 214 285 112 257 165  97  90
35
## [137]  45  86   5 146  59  14  26 293 197 292  29 266 117 223 151 142
185
## [154] 130  23 267 153 290 150 284 147 194  59 288 155 290 132 220 213
221
## [171] 206  71 271 151  97  70  66  51 206  14 211 219 259  31 277 233
4
## [188] 223 157 229 231 206 102  43 141  59  38 276  40 268 152 196  83
292
## [205]   4 277 234 287  33 259 219 202 219 208 106 241  57 126 241  40
162
## [222]  64  48 133 182  18 243 280 125 104 275   7  72  86  61  58 153
241
## [239] 183 212 204  72  47 280 246  94  20 158 123  73  27 133 187 115
108
## [256] 212 257  91  82  78  80 168 238 133 113 208  85 120 180  87  34
57
## [273] 228 286 112 277 222 149  97 290 155 267 222 244 237  71 118 267
37
## [290] 146 205  37 166
##
## $btree
## n= 293
##
## node), split, n, loss, yval, (yprob)
```

```
##       * denotes terminal node
##
##  1) root 293 25 0 (0.91467577 0.08532423)
##    2) Salary< 37.5 276  8 0 (0.97101449 0.02898551)
##      4) Distance< 17.25 263  0 0 (1.00000000 0.00000000) *
##      5) Distance>=17.25 13  5 1 (0.38461538 0.61538462)
##       10) Age< 29.5 5  0 0 (1.00000000 0.00000000) *
##       11) Age>=29.5 8  0 1 (0.00000000 1.00000000) *
##    3) Salary>=37.5 17  0 1 (0.00000000 1.00000000) *
##
## attr(,"class")
##    class
## "sclass"
##
##
## $OOB
## [1] FALSE
##
## $comb
## [1] FALSE
##
## $call
## bagging.data.frame(formula = CarUsage ~ ., data = bag.train,
##     control = rpart.control(maxdepth = 5, minsplit = 4))
##
## attr(,"class")
## [1] "summary.bagging"
```

```
bag.pred = predict(mod.bagging, bag.test)

confusionMatrix_bagging = confusionMatrix(bag.pred,bag.test$CarUsage)
confusionMatrix_bagging
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 117   2
##          1   0   6
##
##                Accuracy : 0.984
##                  95% CI : (0.9434, 0.9981)
##     No Information Rate : 0.936
##     P-Value [Acc > NIR] : 0.01175
##
##                   Kappa : 0.8489
##
##  Mcnemar's Test P-Value : 0.47950
##
##             Sensitivity : 1.0000
##             Specificity : 0.7500
##          Pos Pred Value : 0.9832
```

```
##          Neg Pred Value : 1.0000
##              Prevalence : 0.9360
##          Detection Rate : 0.9360
##    Detection Prevalence : 0.9520
##       Balanced Accuracy : 0.8750
##
##        'Positive' Class : 0
```

4.4.2. Insights

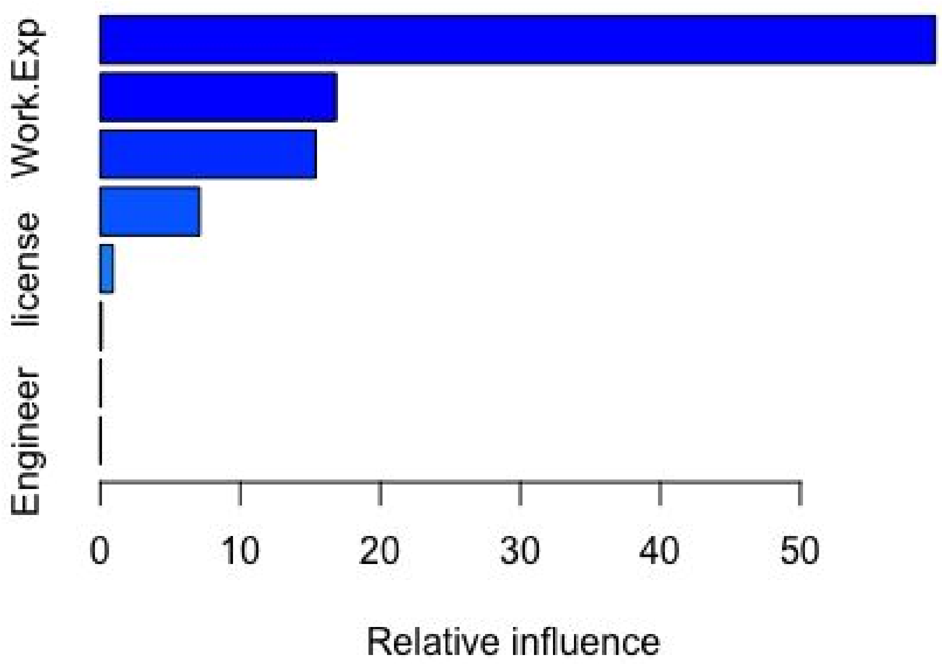
1.Bagging gives us a tremendous results in terms of the accuracy.
2. Apart from that, a higher number is achieved in terms prediction of minority class.

4.5. Boosting

4.5.1 Model Creation

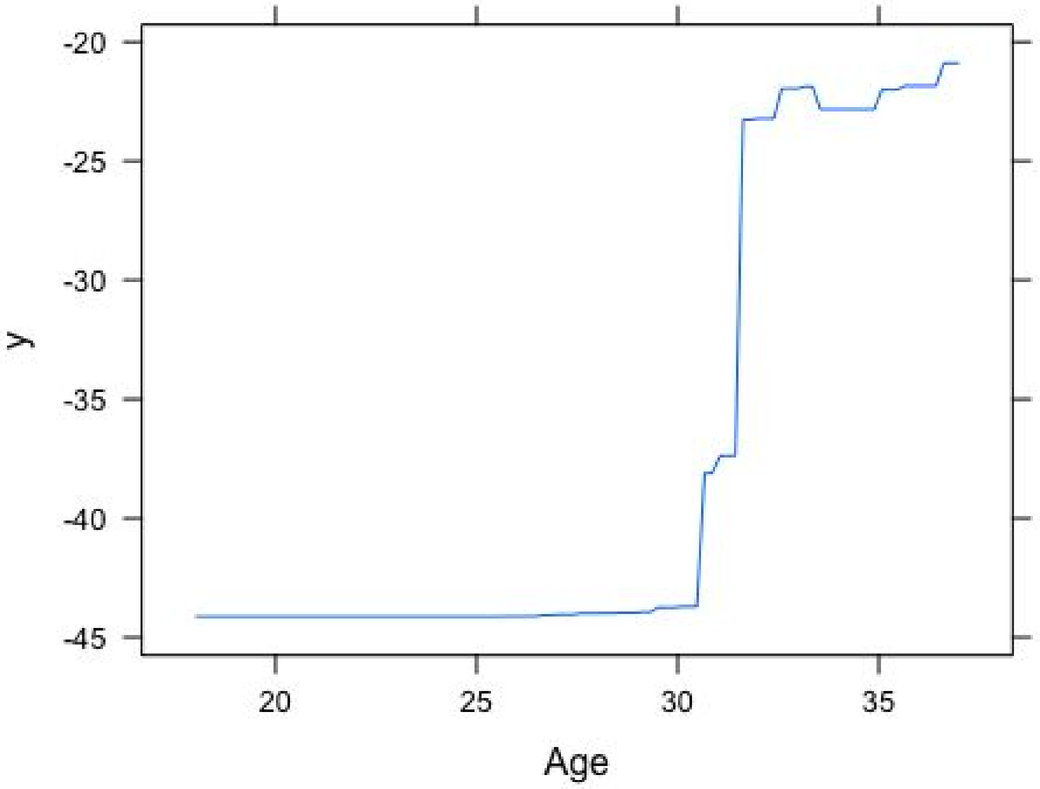
```
train$CarUsage = as.character(train$CarUsage)
mod.boost = gbm(CarUsage ~ .,data=train, distribution=
                      "bernoulli",n.trees =5000 , interaction.depth =4,
shrinkage=0.01)
summary(mod.boost)
```

```
##
var
```



```
rel.inf
## Salary      Salary 5.963819e+01
## Work.Exp Work.Exp 1.684673e+01
## Age            Age 1.538024e+01
## Distance Distance 7.102625e+00
## license    license 9.110294e-01
## MBA            MBA 7.808544e-02
## Gender      Gender 4.276411e-02
## Engineer Engineer 3.367514e-04
```

```
plot(mod.boost)
```




```
boost.pred <- predict(mod.boost, test,n.trees =5000, type="response")

y_pred_num <- ifelse(boost.pred > 0.5, 1, 0)
y_pred <- factor(y_pred_num, levels=c(0, 1))
table(y_pred,test$CarUsage)
```

```
##
## y_pred   0   1
##      0 113   1
##      1   1   9
```

```
boosting_confusion = confusionMatrix(y_pred,test$CarUsage)
print(boosting_confusion)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   0   1
##          0 113   1
##          1   1   9
##                                          
##                Accuracy : 0.9839         
##                  95% CI : (0.943, 0.998)
##     No Information Rate : 0.9194         
##     P-Value [Acc > NIR] : 0.002091       
##                                          
##                   Kappa : 0.8912         
##                                          
##  Mcnemar's Test P-Value : 1.000000       
##                                          
##             Sensitivity : 0.9912         
##             Specificity : 0.9000         
##          Pos Pred Value : 0.9912         
##          Neg Pred Value : 0.9000         
##              Prevalence : 0.9194         
##          Detection Rate : 0.9113         
##    Detection Prevalence : 0.9194         
##       Balanced Accuracy : 0.9456         
##                                          
##        'Positive' Class : 0              
## 
```

4.5.2. Insights

1.The accuracy and kappa values are quite good at count.

2.Balanced accuracy of the data is also quite good in account.

5.Model Evaluation

5.1. Evaluation Prospects

This part of the report helps us evaluate the above models created in part 4. The various metrics taken are accuracy , miss classification error , specificity , sensitivity , kappa values.Below table gives a brief study of the values upto 4 decimal places.

```
ModelType = c("Logistic Regression", "K Nearest Neighbour", "Naive
Bayes","Bagging","Boosting")
```

```r
# Training classification accuracy
Accuracy = c(0.9274,0.9758,0.9516,0.952,0.9839)

# Training misclassification error
Missclass_Error = 1 - Accuracy

# validation classification accuracy
specificity_values = c(logistic_regression_confusion$byClass[2],
knn_confusion$byClass[2],

NaiveBayes_confusion$byClass[2],confusionMatrix_bagging$byClass[2],boostin
g_confusion$byClass[2])

sensitivity_values = c(logistic_regression_confusion$byClass[3],
knn_confusion$byClass[3],

NaiveBayes_confusion$byClass[3],confusionMatrix_bagging$byClass[3],boostin
g_confusion$byClass[3])

kappa_values = c(logistic_regression_confusion$overall[2],
knn_confusion$overall[2],
NaiveBayes_confusion$overall[2],confusionMatrix_bagging$overall[2],boostin
g_confusion$overall[2])

metrics <- data.frame(ModelType, Accuracy, Missclass_Error,
specificity_values,
    sensitivity_values,kappa_values)  # data frame with above metrics

knitr::kable(metrics, digits = 4)  # print table using kable() from knitr
package
```

ModelType	Accuracy	Missclass_Error	specificity_values	sensitivity_values	kappa_values
Logistic Regression	0.9274	0.0726	0.9825	0.8182	0.8440
K Nearest Neighbour	0.9758	0.0242	0.8000	0.9826	0.8290
Naive Bayes	0.9516	0.0484	0.9000	0.9911	0.8006
Bagging	0.9520	0.0480	0.7500	0.9832	0.8489
Boosting	0.9839	0.0161	0.9000	0.9912	0.8912

5.2. Insights

We explored the data completed the steps of data wrangling as per the necessity. We removed the missing values, capped the outliers, and detected the multicollinearity as well. We created 5 models namely Logistic Regression, Naive Bayes, KNN, Bagging and Boosting. From the validation outcomes of confusion matrices of the models created it can be inferred that Boosting has the highest accuracy followed by Bagging.

Further more below are some more insights that might be useful for the business people to take in the model choice in consideration.

Let us summarize the conclusions from analysis and models for employee's decision whether to use a car or not:

1.Important variables are Age, Work.Exp, Distance and License

2.Age and Work.Exp are correlated hence we can conclude that employees with work exp of 10 and above are likely to use car

3.Employees who must commute for distance greater than 12 are more likely to prefer car.Again, people with higher salaries (>20) are likely to use cars

4. Also , before putting the model into production we need to do a final test of overfitting and underfitting issues and act accordingly with the needs like appropriate measures for the skewness that appeared at times to the data values.