Project -5 (Machine Learning)

Employee mode of commuting

By Saleesh Satheeshchandran
PGP-BABI 2019-20 (G-6)

Objective

This project requires you to understand what mode of transport employees prefers to commute to their office. The attached data 'Cars.csv' 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?

Assumptions

There are no particular assumptions.

Tool used for the analysis

RStudio Version 1.2.1335
R Version 3.6.0

Input Data

The data is available in a spreadsheet format with .csv file extension.
The same is uploaded into the R Studio with the help of the function "read.csv.

Steps involved in the analysis

1.1 EDA - Basic data summary, Univariate, Bivariate analysis, graphs, Check for Outliers and missing values and check the summary of the dataset

1.2 EDA - Illustrate the insights based on EDA

1.3 EDA - Check for Multicollinearity - Plot the graph based on Multicollinearity & treat it.

2. Data Preparation (SMOTE)

3.1 Applying Logistic Regression & Interpret results

3.2 Applying KNN Model & Interpret results

3.3 Applying Naïve Bayes Model & Interpret results (is it applicable here? comment and if it is not applicable, how can you build an NB model in this case?)

3.4 Confusion matrix interpretation

Loaded Libraries

```
> library(dplyr)
> library(forcats)
> library(mice)
> library(corrplot)
> library(psych)
> library(car)
> library(caTools)
> library(ROCR)
> library(DMwR)
> library(e1071)
> library(class)
> library(caret)
> library(gbm)
> library(xgboost)
> library(data.table)
> library(scales)
> library(ineq)
> library(ipred)
> library(rpart)
> #EDA
> setwd("/Users/saleesh/Desktop/R Programming")
> Transportation=read.csv("carsedited.csv")
>
```

Exploratory Data Analysis

- There are a total of 9 variables.
- There are 444 observations
- The data has a mix of continuous and categoric variables.
- Continuous variables – Age, Salary, Work Experience, Distance
- Category variables – Gender, Engineer, MBA, License, Transport
- The target variable, “Transport” is a categoric variable with 3 classes.
- Engineer, MBA, License variables are integers and need to be converted to factors.

	Age	Gender	Engineer	MBA	Work.Exp	Salary	Distance	license	Transport
1	28	Male	0	0	4	14.3	3.2	0	Public Transport
2	23	Female	1	0	4	8.3	3.3	0	Public Transport
3	29	Male	1	0	7	13.4	4.1	0	Public Transport
4	28	Female	1	1	5	13.4	4.5	0	Public Transport
5	27	Male	1	0	4	13.4	4.6	0	Public Transport
6	26	Male	1	0	4	12.3	4.8	1	Public Transport
7	28	Male	1	0	5	14.4	5.1	0	2Wheeler
8	26	Female	1	0	3	10.5	5.1	0	Public Transport

```
  Age Gender Engineer MBA Work.Exp Salary Distance license        Transport
1  28   Male        0   0        4   14.3      3.2       0 Public Transport
2  23 Female        1   0        4    8.3      3.3       0 Public Transport
3  29   Male        1   0        7   13.4      4.1       0 Public Transport
4  28 Female        1   1        5   13.4      4.5       0 Public Transport
5  27   Male        1   0        4   13.4      4.6       0 Public Transport
6  26   Male        1   0        4   12.3      4.8       1 Public Transport
> str(Transportation)
'data.frame':   444 obs. of  9 variables:
 $ Age      : int  28 23 29 28 27 26 28 26 22 27 ...
 $ Gender   : Factor w/ 2 levels "Female","Male": 2 1 2 1 2 2 2 1 2 2 ...
 $ Engineer : int  0 1 1 1 1 1 1 1 1 1 ...
 $ MBA      : int  0 0 0 1 0 0 0 0 0 0 ...
 $ Work.Exp : int  4 4 7 5 4 4 5 3 1 4 ...
 $ Salary   : num  14.3 8.3 13.4 13.4 13.4 12.3 14.4 10.5 7.5 13.5 ...
 $ Distance : num  3.2 3.3 4.1 4.5 4.6 4.8 5.1 5.1 5.1 5.2 ...
 $ license  : int  0 0 0 0 0 1 0 0 0 0 ...
 $ Transport: Factor w/ 3 levels "2Wheeler","Car",..: 3 3 3 3 3 3 1 3 3 3 ...
```

```
> summary(Transportation)
      Age            Gender        Engineer           MBA            Work.Exp         Salary
 Min.   :18.00   Female:128   Min.   :0.0000   Min.   :0.0000   Min.   : 0.0   Min.   : 6.50
 1st Qu.:25.00   Male  :316   1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.: 3.0   1st Qu.: 9.80
 Median :27.00                Median :1.0000   Median :0.0000   Median : 5.0   Median :13.60
 Mean   :27.75                Mean   :0.7545   Mean   :0.2528   Mean   : 6.3   Mean   :16.24
 3rd Qu.:30.00                3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 8.0   3rd Qu.:15.72
 Max.   :43.00                Max.   :1.0000   Max.   :1.0000   Max.   :24.0   Max.   :57.00
                                               NA's   :1
    Distance         license                  Transport
 Min.   : 3.20   Min.   :0.0000   2Wheeler        : 83
 1st Qu.: 8.80   1st Qu.:0.0000   Car             : 61
 Median :11.00   Median :0.0000   Public Transport:300
 Mean   :11.32   Mean   :0.2342
 3rd Qu.:13.43   3rd Qu.:0.0000
 Max.   :23.40   Max.   :1.0000
```

Identifying missing values and treating the data

- There is one missing value in the dataset
- The raw containing the missing value is omitted.

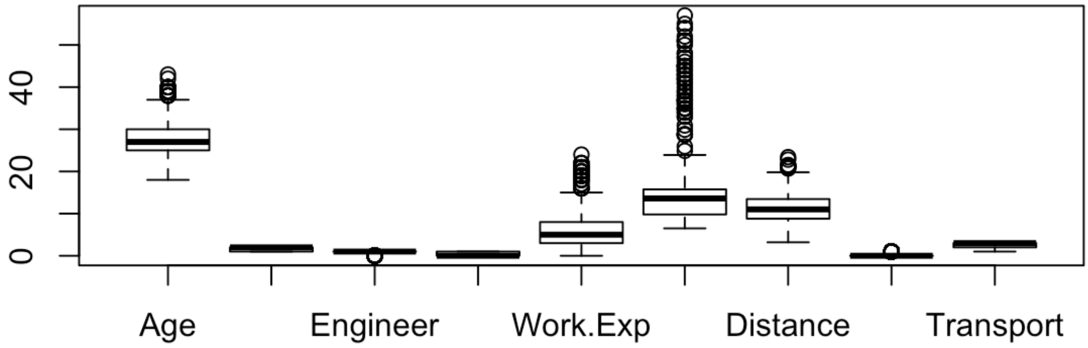
```
is.na(Transportation)
summary(is.na(Transportation))
Transportation=na.omit(Transportation)
```

Treating the class of variables

- The target variable has 3 classes. But as per the problem statement it needs only 2 values that shows “Car” or “Not car”.
- The target variable is changed to a new column named “Caruse” by converting the variable “transport” to a binary data.
- The variables Engineer, MBA, license and Caruse are converted to factors.

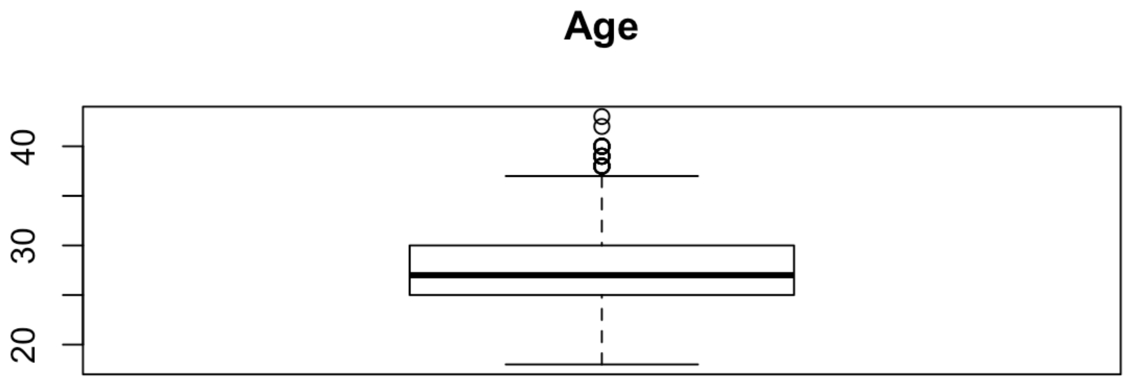
```
Caruse=ifelse(Transportation$Transport == "Car", 1, 0)
Caruse=as.factor(Caruse)
Transportation=cbind(Transportation,Caruse)
Engineer=as.factor(Engineer)
MBA=as.factor(MBA)
license=as.factor(license)
```

Checking for outliers

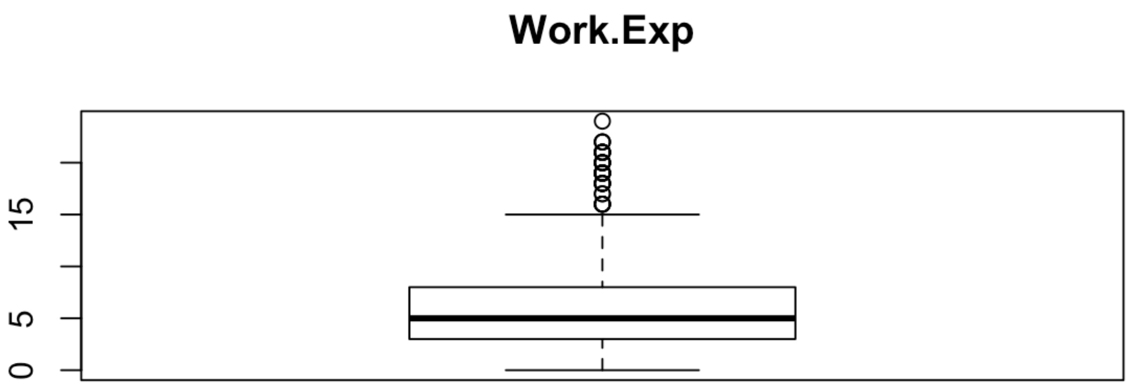


- The data is showing outliers in all the continuous variables

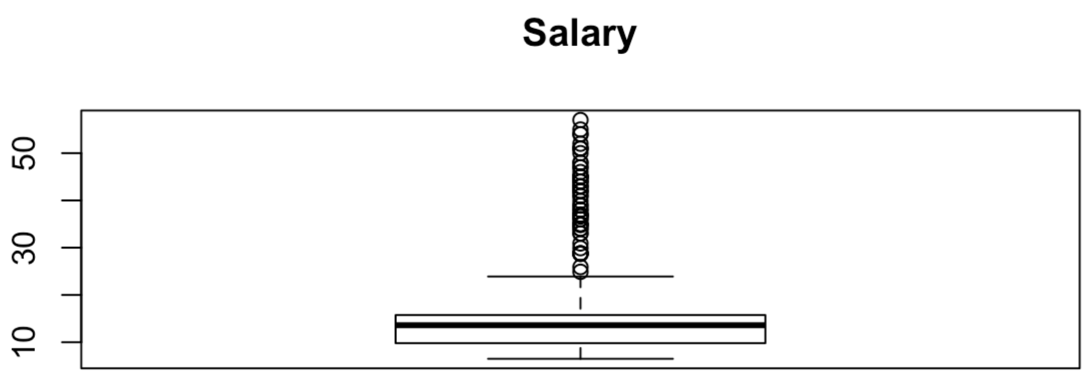
- The plot of “Age”shows 4 outliers. These are significant in the model.



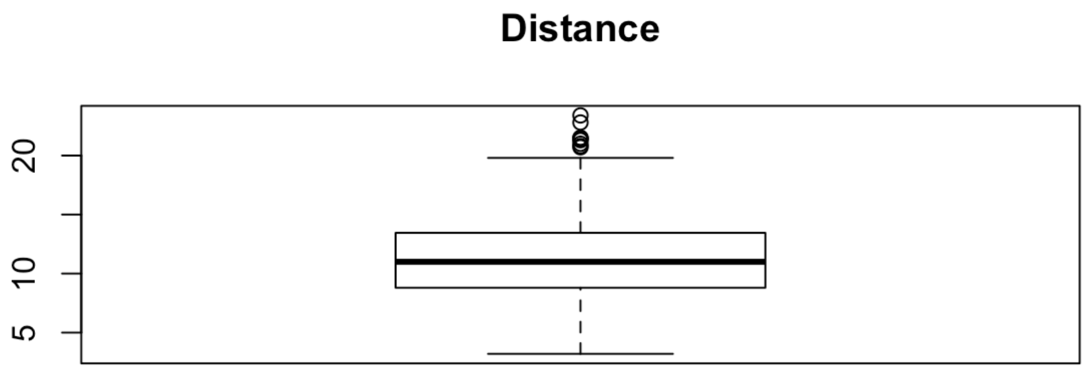
- The plot of “Work Exp”shows 8 outliers. These are significant in the model.



- The plot of “Work Exp”shows numerous outliers. These are significant in the model.



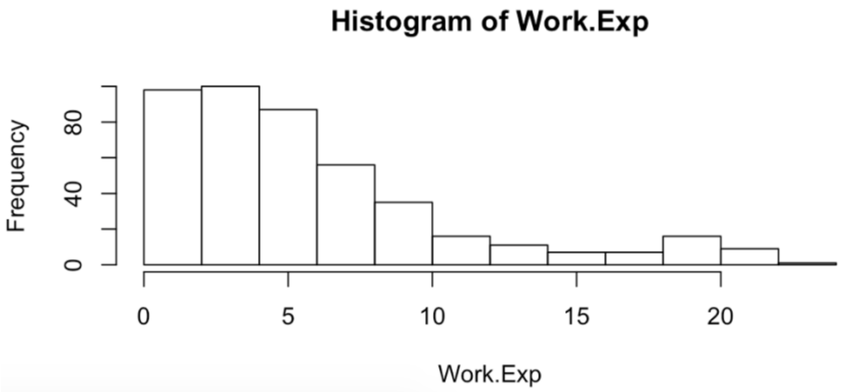
The plot of “Distance ”shows 9 outliers. These are significant in the model



Univariate Analysis

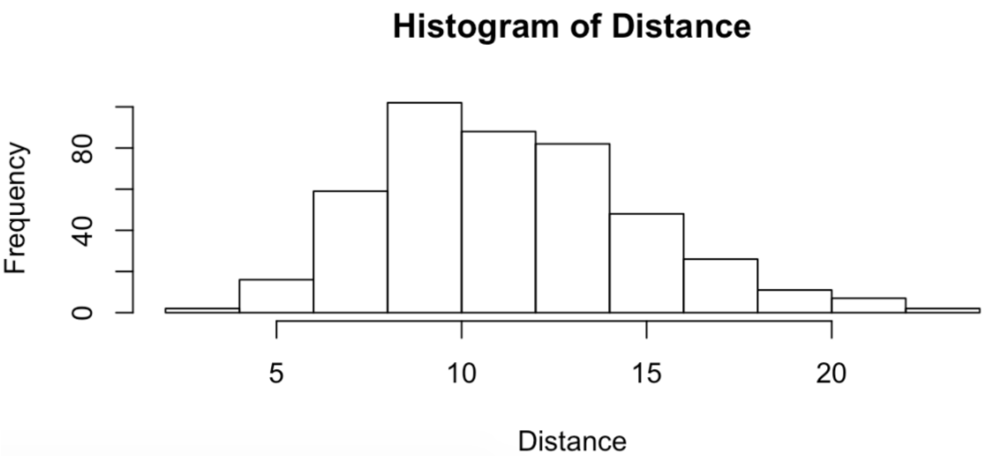
The histogram of Age shows that the variable is skewed towards left.

The histogram of Work.Exp shows that the variable is skewed towards left.

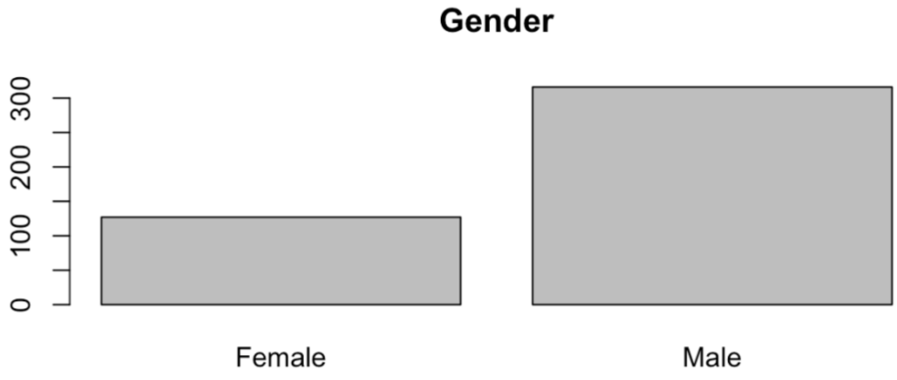


The histogram of Salary shows that the variable is skewed towards left.

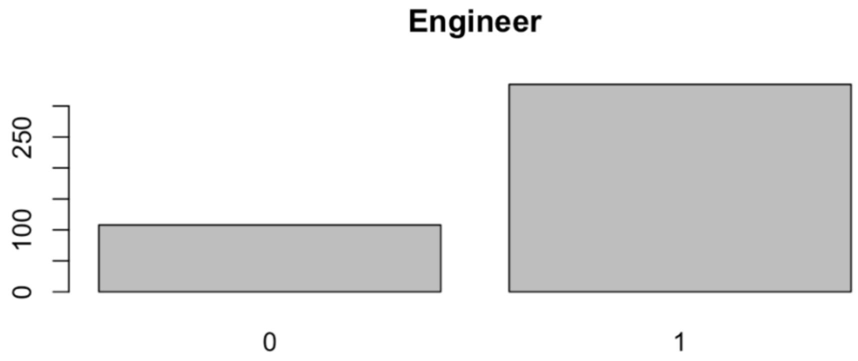
The histogram of Distance shows that the variable is skewed towards left.



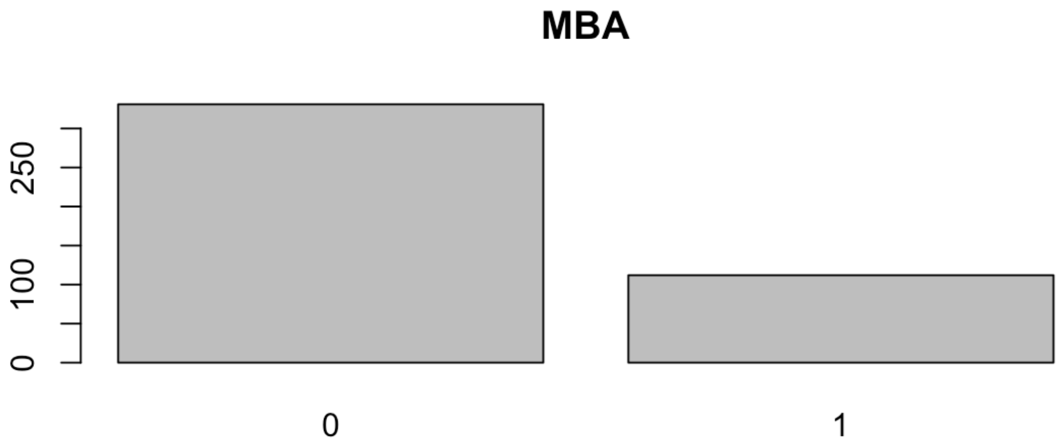
The plot of Gender shows that there are twice as many males as to females.



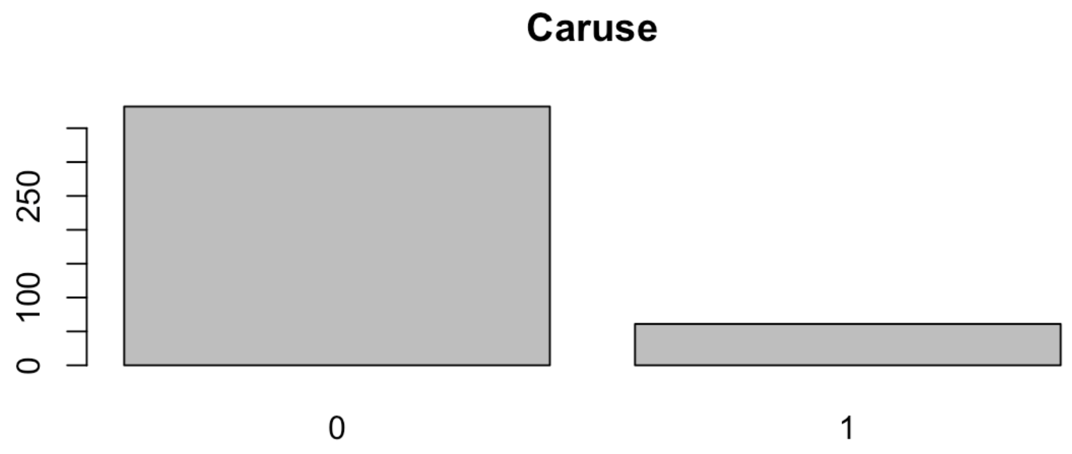
The plot of Engineer shows that there are thrice as many Engineers as to Non-Engineers.



The plot of Gender shows that there are one-third as many MBAs as to non-MBA’s.

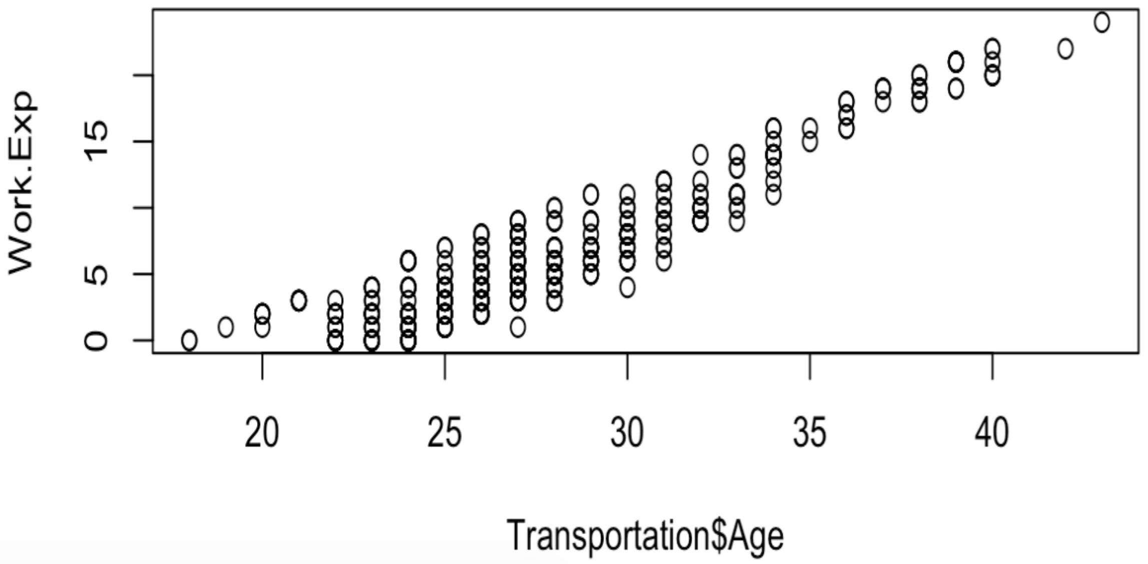


The plot of Caruse shows that there are one-fifth as many car users as to non-car users.

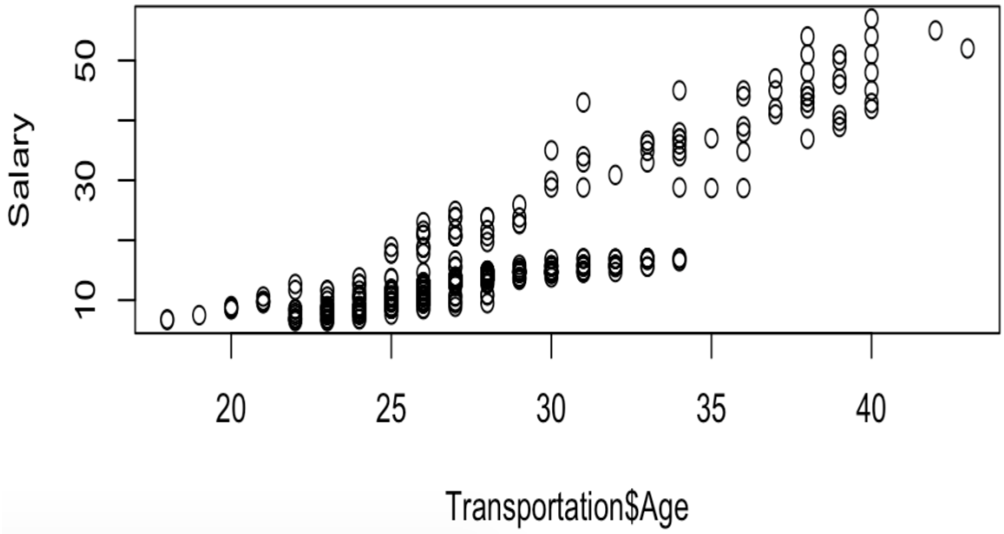


Bi-variate Analysis

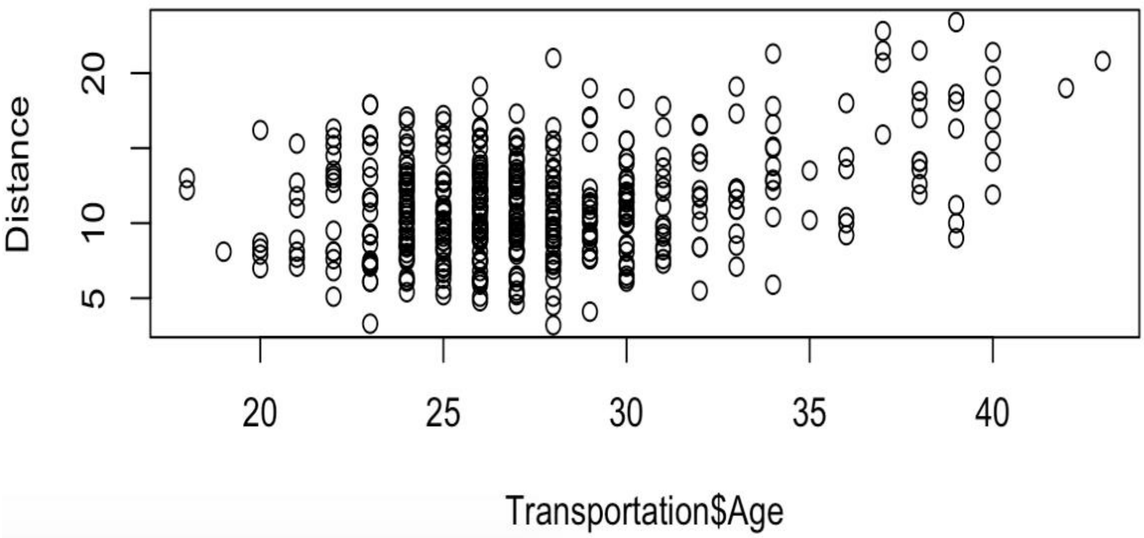
Age vs Work.Exp plot shows a high correlation. (0.932251)



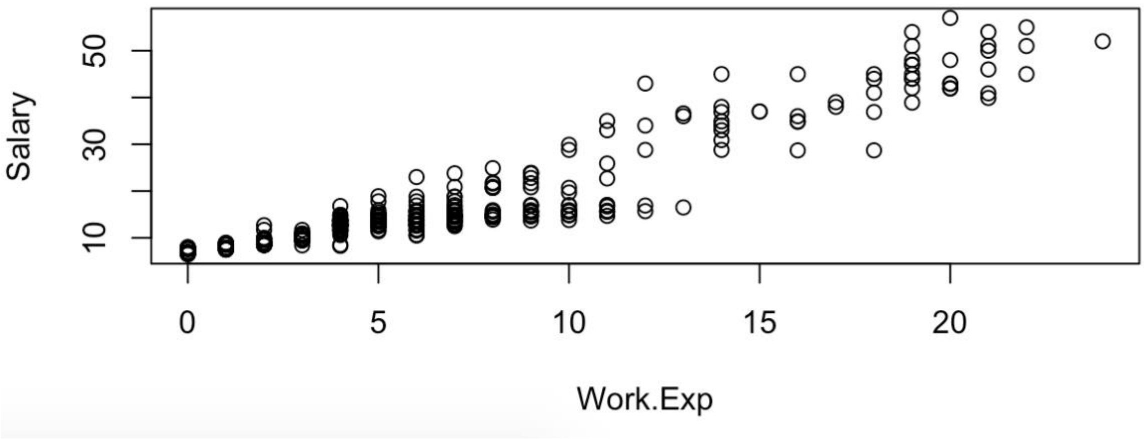
Age vs Salary plot shows a high correlation.(0.8607652



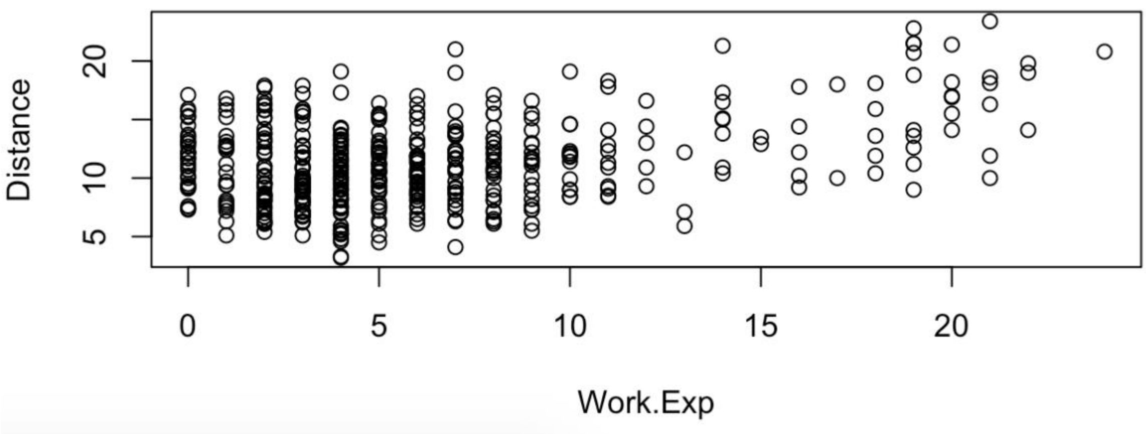
Age vs Distance plot shows a low correlation.(0.3530563)



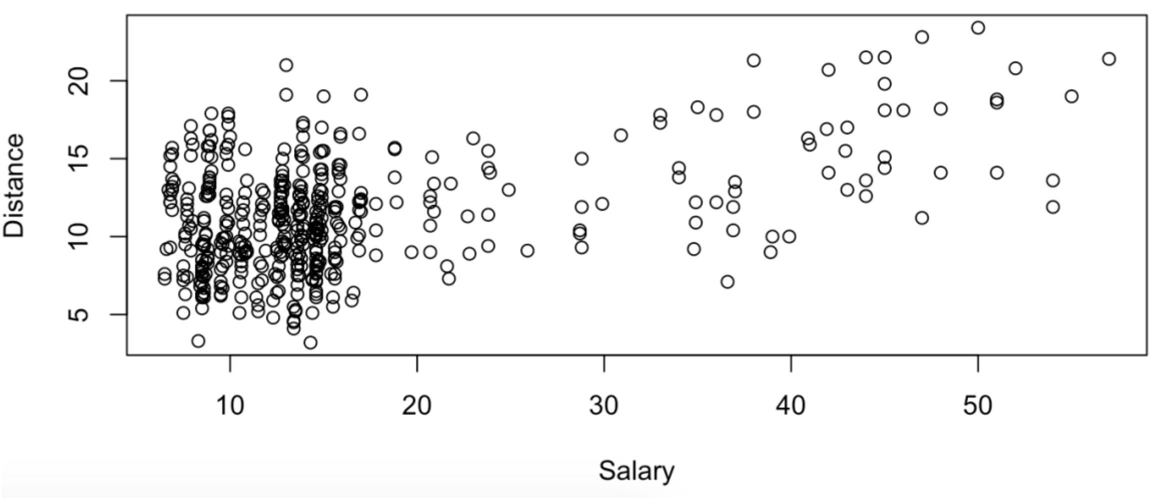
Work.Exp vs Salary plot shows a high correlation.(0.9320081)



Work.Exp vs Distance plot shows a low correlation.(0.3727857)



Work.Exp vs Distance plot shows a low correlation.(0.4422379)



Checking Multicollinearity and treating it

The below process is followed to check multicollinearity.

1. Create a logistic regression model with all variables.
2. Check the variable inflation factor.
3. Check the summary and identify variables showing considerable inflation.

```
> glm.trans.full=glm(Caruse~Age+Work.Exp+Salary+Distance+Gender+Engineer+MBA+license,data=Transportation,family="binomial")
> vif(glm.trans.full)
      Age  Work.Exp    Salary  Distance    Gender  Engineer       MBA   license
11.903723 16.950407  3.970501  1.714671  1.486422  1.112810  1.461142  1.844857
> summary(glm.trans.full)

Call:
glm(formula = Caruse ~ Age + Work.Exp + Salary + Distance + Gender +
    Engineer + MBA + license, family = "binomial", data = Transportation)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.99436  -0.04239  -0.00718  -0.00050   2.27142

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -71.0540    15.6669  -4.535 5.75e-06 ***
Age           2.2605     0.5264   4.294 1.75e-05 ***
Work.Exp     -1.1989     0.3617  -3.315 0.000917 ***
Salary        0.1852     0.0720   2.573 0.010086 *
Distance      0.4906     0.1409   3.482 0.000499 ***
GenderMale   -1.7066     0.8336  -2.047 0.040631 *
Engineer      0.8569     0.9138   0.938 0.348396
MBA          -1.9357     0.9094  -2.129 0.033285 *
license       2.7085     0.8635   3.137 0.001709 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 355.075  on 442  degrees of freedom
Residual deviance:  63.262  on 434  degrees of freedom
AIC: 81.262

Number of Fisher Scoring iterations: 10
```

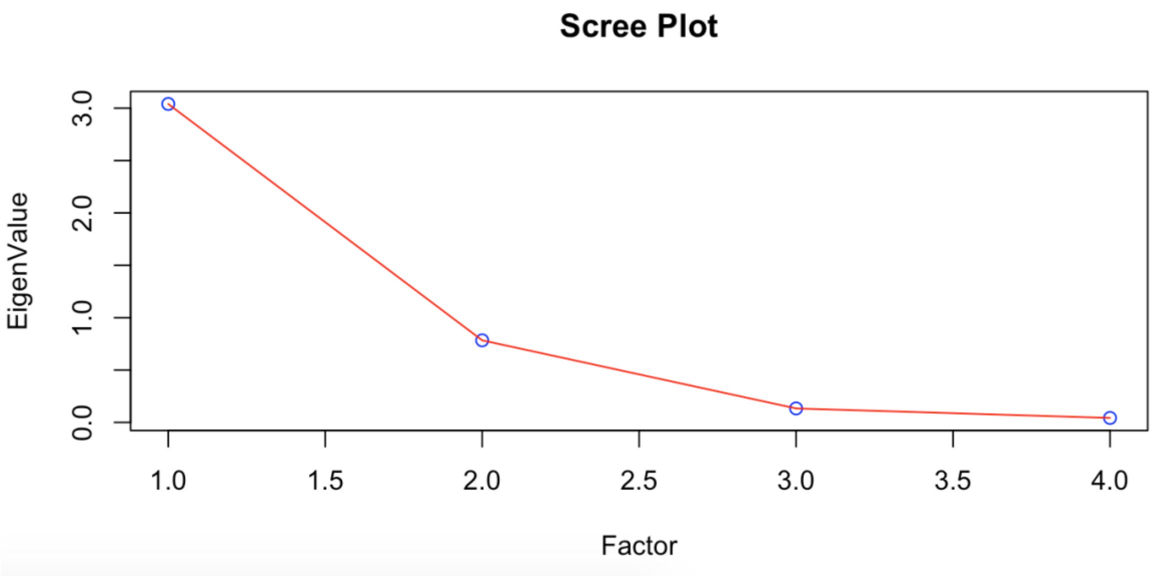
The following variables show high **contribution to multicollinearity.**
- Age
- Work Exp
- Salary

Principle Component Analysis is done on continuous variables to eliminate multi-collinearity

```
> cor(Transportation[,-c(2,3,4,8,9,10)])
               Age  Work.Exp    Salary  Distance
Age      1.0000000 0.9322510 0.8607652 0.3530563
Work.Exp 0.9322510 1.0000000 0.9320081 0.3727857
Salary   0.8607652 0.9320081 1.0000000 0.4422379
Distance 0.3530563 0.3727857 0.4422379 1.0000000
> ev = eigen(cor(Transportation[,-c(2,3,4,8,9,10)]))
> ev
eigen() decomposition
$values
[1] 3.04001463 0.78379167 0.13359073 0.04260296

$vectors
           [,1]        [,2]        [,3]        [,4]
[1,] -0.5398327  0.22950168  0.70211845 -0.40365748
[2,] -0.5550864  0.21330159 -0.03472008  0.80322854
[3,] -0.5494388  0.09873911 -0.70562278 -0.43642186
[4,] -0.3139694 -0.94450092  0.08899827  0.03768969

> EigenValue=ev$values
> EigenValue
[1] 3.04001463 0.78379167 0.13359073 0.04260296
> Factor=c(1,2,3,4)
> Scree=data.frame(Factor,EigenValue)
> plot(Scree,main="Scree Plot", col="Blue")
> lines(Scree,col="Red")
>
```



Chi-Square test is done to determine the multicollinearity of categoric variables.
There is no significant multicollinearity between categoric variables.

```
chisq.test(Caruse,Gender)
chisq.test(Caruse,Engineer)
chisq.test(Caruse,MBA)
chisq.test(Caruse,licence)
chisq.test(Gender,Engineer)
chisq.test(Gender,MBA)
chisq.test(Gender,licence)
chisq.test(Engineer,MBA)
chisq.test(Engineer,licence)
chisq.test(MBA,licence)
```

The variable "Salary" is eliminated due to multi-collinearity.

Treating the data for Analysis

```
table(Caruse)
set.seed(123)
spl = sample.split(Transportation$Caruse, SplitRatio = 0.7)
train = subset(Transportation, spl == T)
test = subset(Transportation, spl == F)

dim(train)
dim(test)
prop.table(table(train$Caruse))
prop.table(table(test$Car))
```

- A seed is set
- Data is split into Test and Train.
- Train has 310 observations
- Test has 133 Observations
- Proportion of the target variables in test and train are confirmed to be similar.


```
table(Caruse)
set.seed(123)
spl = sample.split(Transportation$Caruse, SplitRatio = 0.7)
train = subset(Transportation, spl == T)
test = subset(Transportation, spl == F)

dim(train)
dim(test)
prop.table(table(train$Caruse))
prop.table(table(test$Car))
```

Logistic Regression

One of the best models for predictions when the variable to be determined is binary in nature is the logistic regression model.
Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

A logistic regression model is run with all the significant variables that are available.

```
> LRmodel = glm(Caruse ~., data = train, family = binomial)
> summary(LRmodel)

Call:
glm(formula = Caruse ~ ., family = binomial, data = train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.13151  -0.06645  -0.01375  -0.00173   2.28930

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -55.7249    14.3015  -3.896 9.76e-05 ***
Age           1.7594     0.5017   3.507 0.000454 ***
GenderMale   -1.2414     0.9366  -1.325 0.185015
Engineer      0.7486     1.0403   0.720 0.471791
MBA          -1.8681     1.0482  -1.782 0.074706 .
Work.Exp     -0.6623     0.2802  -2.363 0.018109 *
Distance      0.3922     0.1406   2.790 0.005266 **
license       2.8417     0.9643   2.947 0.003209 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 249.621  on 309  degrees of freedom
Residual deviance:  49.052  on 302  degrees of freedom
AIC: 65.052

Number of Fisher Scoring iterations: 9

>
```

4 of the variables as per below are identified to be non-significant in the model.
- Gender
- Engineer
- MBA
- Salary

The mentioned variables are removed and the logistic regression model is run again.

```
> LRmodel.sub = glm(Caruse ~., data = train.sub, family = binomial)
> summary(LRmodel.sub)

Call:
glm(formula = Caruse ~ ., family = binomial, data = train.sub)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.14713  -0.12698  -0.04231  -0.01128   2.66213

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -39.6260    10.3305  -3.836 0.000125 ***
Age           1.3223     0.3945   3.352 0.000802 ***
Work.Exp     -0.4430     0.2492  -1.778 0.075484 .
license       2.1968     0.6884   3.191 0.001416 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 249.621  on 309  degrees of freedom
Residual deviance:  66.519  on 306  degrees of freedom
AIC: 74.519

Number of Fisher Scoring iterations: 8

>
```

Testing the performance

```
> predTest = predict(LRmodel.sub, newdata = test.sub, type = 'response')
> cmLR = table(test.sub$Caruse, predTest>0.1)
> sum(diag(cmLR))/sum(cmLR)
[1] 0.9323308
> cmLR

    FALSE TRUE
  0   107    8
  1     1   17
```

Measure	Value
Sensitivity	0.9444
Specificity	0.9304
Precision	0.6800

Accuracy	0.9323

SMOTE

- The given data is under sampled.
- Smote is performed to boost positive observations of target variable.

```
610 366
> balanced.data = SMOTE(Caruse ~.,perc.over = 500 , Transportation, k = 5, perc.under = 200)
> table(balanced.data$Caruse)

  0   1
610 366
```

- After Smote the positive occurrences are boosted to 366 numbers.
-The negative occurrences are boosted to 610 numbers .

Logistic Regression after Smote

- The data after SMOTE is used to perform Logistic Regression.

```
Call:
glm(formula = Caruse ~ ., family = binomial, data = train.bal)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.6592  -0.0130  -0.0008   0.0077   2.0150

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -76.4151    11.3576  -6.728 1.72e-11 ***
Age           2.4058     0.3750   6.416 1.40e-10 ***
GenderMale   -1.5476     0.6497  -2.382 0.017223 *
Engineer      0.9642     0.6810   1.416 0.156817
MBA          -1.8135     0.7061  -2.568 0.010214 *
Work.Exp     -0.7187     0.1946  -3.693 0.000221 ***
Distance      0.6337     0.1346   4.707 2.52e-06 ***
license       1.3068     0.6579   1.986 0.046983 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 903.57  on 682  degrees of freedom
Residual deviance: 108.22  on 675  degrees of freedom
AIC: 124.22

Number of Fisher Scoring iterations: 9
```

Measuring the model performance.

```
> pred.Test.smote = predict(LR.smote, newdata = test.bal, type = 'response')
> cm.smote = table(test.bal$Caruse, pred.Test.smote >0.1)
> cm.smote

    FALSE TRUE
  0   162   21
  1     0  110
```

Measure	Value
Sensitivity	0.8397
Specificity	1.0000
Precision	1.0000
Accuracy	0.9283

The results show that SMOTE caused below changes to the model
- Decreased Sensitivity
- Increased Specificity
- Increased Precision
- Decreased Accuracy

Naïve Bayes (Can be used in this model)

Naïve Bayes is a classification model which was initially established with a condition that all the variables are supposed to be categorical and not continuous.
But in practical situations such a condition is not possible as we would be always getting a mix of continuous and categorical variables.
We can try building a Naïve Bayes model data using the library (e1071) in R.

```
> NBmodel = naiveBayes(Caruse ~., data = train)
> NBpredTest = predict(NBmodel, newdata = test)
> tabNB = table(test$Caruse, NBpredTest)
> tabNB
   NBpredTest
      0   1
  0 113   2
  1   3  15
```

Model Performance

Measure	Value
Sensitivity	0.8824
Specificity	0.9741
Precision	0.8333
Accuracy	0.9624

Naïve Bayes after SMOTE

The Naïve Bayes model is run on the data that was amplified with SMOTE

```
> NBmodel.bal = naiveBayes(Caruse ~., data = train.bal)
> NBpredTest.bal = predict(NBmodel.bal, newdata = test.bal)
> tabNB.bal = table(test.bal$Caruse, NBpredTest.bal)
> tabNB.bal
   NBpredTest.bal
      0   1
  0 174   9
  1   7 103
> sum(diag(tabNB.bal))/sum(tabNB.bal)#Shows 94.5% sensitivity
[1] 0.9453925
```

Model Performance

Measure	Value
Sensitivity	0.9196
Specificity	0.9613
Precision	0.9364
Accuracy	0.9454

The results show that SMOTE caused below changes to the model

- Increased Sensitivity
- Increased Specificity
- Increased Precision
- Decreased Accuracy

K Nearest Neighbour (KNN)

k nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. This algorithms segregates unlabeled data points into well defined groups.

The kNN algorithm is applied to the training data set and the results are verified on the test data set.

KNN Model does not have a formula like regression models.
It just classifies
Multiple values of K was used and experimented with the model.
When the value of K was 3, it seemed to predict the values much closer to the initial % of the Car usage rate.

```
> train.num = train[,sapply(train, is.numeric)]
> test.num = test[,sapply(train, is.numeric)]
> names(train.num)
[1] "Age"      "Engineer" "MBA"      "Work.Exp" "Distance" "license"
> predKNNmodel = knn(train = train.num, test = test.num, cl = train[,3], k = 3)
> tabKNN = table(test$Caruse, predKNNmodel)
> tabKNN
   predKNNmodel
     0  1
  0 18 97
  1  0 18
> sum(diag(tabKNN))/sum(tabKNN)
[1] 0.2706767
```

Model Performance

Measure	Value
Sensitivity	0.1565
Specificity	1.0000
Precision	1.0000
Accuracy	0.2707

KNN with SMOTE Data

```
> train.num.bal = train.bal[,sapply(train.bal, is.numeric)]
> test.num.bal = test.bal[,sapply(train.bal, is.numeric)]
> names(train.num.bal)
[1] "Age"      "Engineer" "MBA"      "Work.Exp" "Distance" "license"
> predKNNmodel.bal = knn(train = train.num.bal, test = test.num.bal, cl = train.bal[,3], k = 3)
> tabKNN.bal = table(test.bal$Caruse, predKNNmodel.bal)
> tabKNN.bal
   predKNNmodel.bal
     0 6.30451831966639e-05 0.0870488593354821 0.0944892147090286 0.228598581394181
  0  38                    0                  0                  0                 0
  1   8                    0                  0                  0                 0
   predKNNmodel.bal
    0.255002213642001 0.43207043223083 0.507282199338078 0.542174537898973
  0                 0                0                 0                 0
  1                 0                0                 0                 0
   predKNNmodel.bal
    0.562992612132803 0.628368729492649 0.643847410799935 0.656610624166206
  0                 0                 0                 0                 0
  1                 0                 0                 0                 0
   predKNNmodel.bal
    0.67570475791581 0.792727085994557 0.795135331572965   1
  0                0                 0                 0 145
  1                0                 0                 0 102
> sum(diag(tabKNN))/sum(tabKNN)
[1] 0.2706767
>
> knn_fit
k-Nearest Neighbors

683 samples
  7 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 456, 455, 455
Resampling results across tuning parameters:

  k  Accuracy   Kappa
   5  0.9721707  0.9407022
   7  0.9721771  0.9407100
   9  0.9765631  0.9501279
  11  0.9721643  0.9407974
  13  0.9692338  0.9344332
  15  0.9721578  0.9406709
  17  0.9692338  0.9344332
  19  0.9677719  0.9312548
  21  0.9648414  0.9253263
  23  0.9662970  0.9285161

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 9.
>
> predKNN_fit = predict(knn_fit, newdata = test.bal[,-8], type = "raw")
> tabknnfit=table(test.bal$Caruse, predKNN_fit)
> tabknnfit
   predKNN_fit
      0   1
  0 180   3
  1   3 107
> sum(diag(tabknnfit))/sum(tabknnfit)
[1] 0.9795222
```

Measuring Performance

Measure	Value
Sensitivity	0.9727
Specificity	0.9836
Precision	0.9727
Accuracy	0.9795

Bagging

Bootstrap aggregating, also called bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting.

```
> Transportation.bagging <- bagging(Caruse ~.,
+                           data=train,
+                           control=rpart.control(maxdepth=5, minsplit=4))
>
> test$pred.caruse <- predict(Transportation.bagging, test)
>
> table(test$Caruse,test$pred.caruse)

      0   1
  0 115   0
  1   5  13
```

Bagging performance

Sensitivity	1.0000
Specificity	0.9583
Precision	0.7222
Accuracy	0.9624

Boosting

```
train.bal$Engineer =as.numeric(train.bal$Engineer)
train.bal$MBA = as.numeric(train.bal$MBA)
train.bal$license = as.numeric(train.bal$license)
train.bal$Caruse = as.numeric(train.bal$Caruse)
test.bal$Engineer =as.numeric(test.bal$Engineer)
test.bal$MBA = as.numeric(test.bal$MBA)
test.bal$license = as.numeric(test.bal$license)
test.bal$Caruse = as.numeric(test.bal$Caruse)
train.bal$Caruse[train.bal$Caruse == 1] = 0
train.bal$Caruse[train.bal$Caruse == 2] = 1
test.bal$Caruse[test.bal$Caruse == 1] = 0
test.bal$Caruse[test.bal$Caruse == 2] = 1

features_train = as.matrix(train.bal[,c(1,2,6,7)])
label_train = as.matrix(train.bal[,7])
features_test = as.matrix(test.bal[,c(1,2,6,7)])

XGBmodel = xgboost(
  data = features_train,
  label = label_train,
  eta = .001,
  max_depth = 5,
  min_child_weight = 3,
  nrounds = 10,
  nfold = 5,
  objective = "binary:logistic",
  verbose = 0,
  early_stopping_rounds = 10
)

XGBpredTest = predict(XGBmodel, features_test)
tabXGB = table(test.bal$Caruse, XGBpredTest>0.5)
tabXGB
sum(diag(tabXGB))/sum(tabXGB)
```

Actionable Insights

From the various analysis and modelling done on this project, Naïve Bayes with Smoting is identified to be the best model.
The model can predict the Car usage probability of a customer with about 97% accuracy.
It is quite clear from the models tried that 3 of the variables are very critical in deciding the Car Usage rate.

1. Age
2. Distance
3. License

This shows that if the company focuses to improve these 4 variables, they can significantly improve the prediction of Car usage pattern of employees.

SMOTE has been identified as a very good tool to boost the data in this case.
Bagging too has been found to be very fruitful. The results came just second behind Naïve Bayes with Smoting with about 96% Accuracy.

Both the methods are also helpful in increasing sensitivity and specificity.