Mini Project – Predicting mode of Transport

Project Report

Jyosmitha M

Contents

1. Project Objective

To predict whether an employee will use Car as a mode of transport. Also, which variables are a significant predictor behind this decision. This project requires you to understand what mode of transport employees prefers to commute to their office. Below are the items that will be

- Importing the dataset in R
- Understanding the structure of dataset and modifying the data into format
- Graphical exploration
- Descriptive statistics
- Insights from the dataset
- Check if the assumptions are met
- Check for multicollinearity
- Creating models using logistic regression, KNN and Naïve Bayes model
- Confusion Matrix
- Model Validation
- Bagging and boosting
- Actionable insights and Recommendations

2. Assumptions

Data is highly imbalanced and there maybe be possible outliers and missing values.

Data might not be following a normal distribution.

Logistic Regression:

- No outliers should be present
- No missing values should be present
- There should not be a multi-collinearity between independent variables. (only a little collinearity is allowed)
- There should be a linear relationship between the link function and independent variables in logit model.
- Dependent variables need not be normally distributed
- Dataset should be large.
- Errors need to be independent and not normally distributed.

KNN Model:

- No specific assumptions but data must be scaled before building model to avoid influence of variable in higher metric

Naïve Bayes:

- All the featured variables are independent and not correlated with each other. This is called as conditional independence

- Numerical variables should be normally distributed

<u>Bagging:</u>
Bagging is applied when the dataset is having low bias but high variance. Bagging reduces the high variance

<u>Boosting:</u>
Boosting might take more time to run if the dataset size is huge. Boosting focuses on reducing biasness in the dataset.

<u>XGBoost:</u>
XGBoost is used to reduce both bias and variance. Dataset passed into algorithm must be in the form of numerical matrix.
Hot encoding should be done for Categorical variables

3. Environment Set up and Data Import

3.1 Install necessary Packages and Invoke Libraries

- library(mice)
- library(ggplot2)
- library(gbm)
- library(Ckmeans.1d.dp)
- library(xgboost)
- library(dmm)
- library(xgboost)
- library(reshape2)
- library(DataExplorer)
- library(corrplot)
- library(ipred)
- library(rpart)
- library(DMwR)
- library(gridExtra)
- library(e1071)
- library(GGally)
- library(mice)
- library(ROCR)
- library(ineq)
- library(plyr)
- library(car)
- library(lmtest)
- library(pan)
- library(corrplot)
- library(ggplot2)

3.2 Functions used in R-code:

- md.pattern

- sapply
- subset
- cbind
- melt
- table
- round
- vif
- glm
- chisq.test
- sample.split
- predict
- shapiro.test
- bptest
- ineq
- scale,KNN
- naiveBayes
- gbm
- xgbm

3.3 Set up working Directory

Setting a working directory on starting of the R session makes importing and exporting data files and code files easier. Basically, working directory is the location/ folder on the PC where you have the data, codes etc. related to the project

```
> setwd("C:/Users/ammu/Desktop/Great Lakes/6. Machine Learning/Project")
> getwd()
[1] "C:/Users/ammu/Desktop/Great Lakes/6. Machine Learning/Project"
```

3.3 Import and Read the Dataset

Data in csv format is imported into R environment.

```
> Cars=read.csv("Cars.csv")
```

4. Meta Data:

Churn is the predictor/response categorical variable. ContractRenewal and DataPlan are categorical and rest are Continuous variables

Column Name	Description
Age	Age of the Employee in Years
Gender	Gender of the Employee
Engineer	For Engineer =1 , Non Engineer =0
MBA	For MBA =1 , Non MBA =0
Work Exp	Experience in years
Salary	Salary in Lakhs per Annum

Distance	Distance in Kms from Home to Office
license	If Employee has Driving Licence -1, If not, then 0
Transport	Mode of Transport
Age	Age of the Employee in Years
Gender	Gender of the Employee

5. Exploratory Data Analysis

5.1 Column names and number of observations:

Colum names and dimensions of dataset: There are 444 rows and 9 columns in the dataset

```
> dim(Cars)
[1] 444   9
> head(Cars)
  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
> tail(Cars)
    Age Gender Engineer MBA Work.Exp Salary Distance license Transport
439  34   Male        1   0       14     38     21.3       1       Car
440  40   Male        1   0       20     57     21.4       1       Car
441  38   Male        1   0       19     44     21.5       1       Car
442  37   Male        1   0       19     45     21.5       1       Car
443  37   Male        0   0       19     47     22.8       1       Car
444  39   Male        1   1       21     50     23.4       1       Car
> names(Cars)
[1] "Age"       "Gender"    "Engineer"  "MBA"       "Work.Exp"  "Salary"
[7] "Distance"  "license"   "Transport"
```

5.2 Missing Values:

There is one missing values in MBA column in the dataset.


```
> anyNA(Cars)
[1] TRUE
```

```
> md.pattern(Cars)
    Age Gender Engineer Work.Exp Salary Distance license Transport MBA
443   1      1        1        1      1        1       1         1   1 0
1     1      1        1        1      1        1       1         1   0 1
      0      0        0        0      0        0       0         0   1 1
> sum(is.na(Cars))
[1] 1
> colSums(is.na(Cars))
      Age     Gender   Engineer        MBA   Work.Exp     Salary   Distance    licen
se
        0          0          0          1          0          0          0
0
Transport
        0
> sum(rowSums(is.na(Cars)))
[1] 1
```

5.3 Structure of the dataset:

All the variables are of numeric datatype. We must change Churn,ContractRenewal,DataPlan columns into factors as they are categorical variables

```
> str(Cars)
'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 ...
```

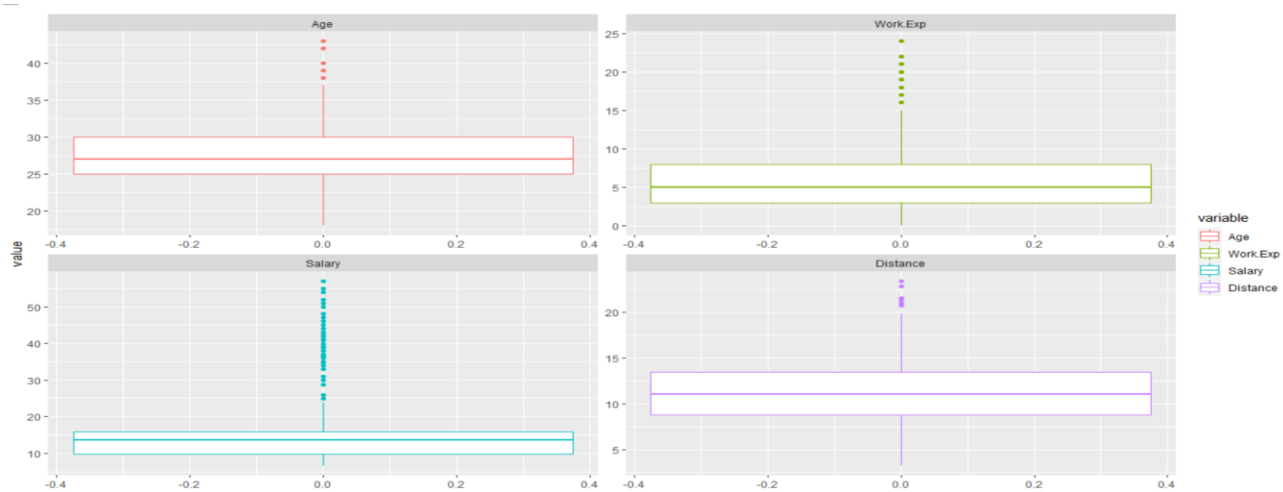
5.4 Five point Summary:

- **Churn, ContractRenewal, DataPlan** would be categorical variables
- **Possible outliers** in most of the variables except categorical variables.
- **Null values** are present in Family Members columns

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

5.5 Outliers:

- Outliers are present in all the continuous variables.
- This is expected phenomenon and outliers here are valid



5.6 Removing nulls:

```
> #remove nulls
> CarsData=CarsData[complete.cases(CarsData), ]
> anyNA(CarsData)
[1] FALSE
> sum(colSums(is.na(CarsData)))
[1] 0
> sum(rowSums(is.na(CarsData)))
[1] 0
```

5.7 Converting categorical variables into factors:

Engineer, MBA and license are converted into factors and datatypes for all variables are checked

```
> #converting variables into factors
> CarsData$Engineer=as.factor(CarsData$Engineer)
> CarsData$MBA=as.factor(CarsData$MBA)
> CarsData$license=as.factor(CarsData$license)
> str(CarsData)
'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 : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
 $ MBA      : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
 $ 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  : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
 $ Transport: Factor w/ 3 levels "2Wheeler","Car",..: 3 3 3 3 3 3 1 3 3 3 ...
```

5.8 Separating Continuous and Categorical variables:

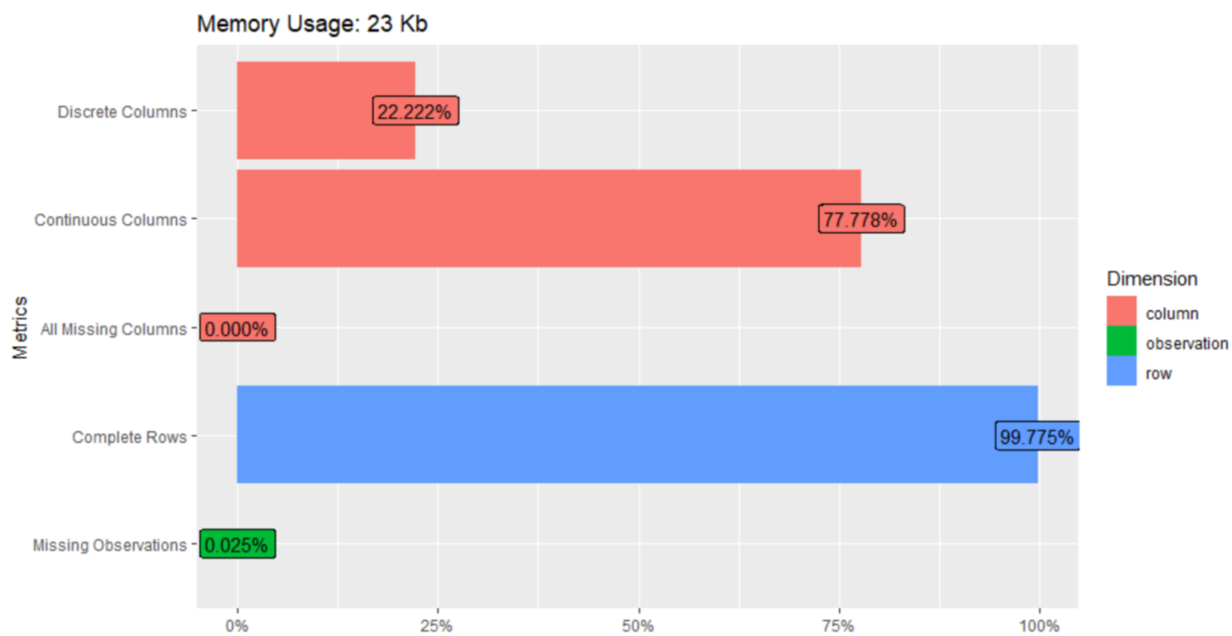
Gender, Engineer, MBA, License, Transport are categorical variables

Age, Work Experience, Salary, Distance are continuous variables

Transport if predictor/dependent variable and other variables are independent variables

```
> CarsContinous=CarsData[c("Age","Work.Exp","Salary","Distance")]
> CarsCategorical=CarsData[c("Gender","Engineer","MBA","license","Transport")
]
> names(CarsCategorical)
[1] "Gender"    "Engineer"  "MBA"       "license"   "Transport"
> names(CarsContinous)
[1] "Age"      "Work.Exp" "Salary"   "Distance"
```

5.9 Dataset Overview:



5.10 Scaling features for KNN Model:

```
> #scaling for KNN Model
> CarsContinousScaled=scale(CarsContinous)
> CarsContinousScaledTransport=cbind(CarsContinousScaled,Transport)
> CarsContinousScaledTransport=as.data.frame(CarsContinousScaledTransport)
> str(CarsContinousScaledTransport)
'data.frame':   444 obs. of  5 variables:
 $ Age      : num  0.0571 -1.075 0.2835 0.0571 -0.1693 ...
 $ Work.Exp : num  -0.45 -0.45 0.137 -0.254 -0.45 ...
 $ Salary   : num  -0.185 -0.759 -0.272 -0.272 -0.272 ...
 $ Distance : num  -2.25 -2.22 -2 -1.89 -1.86 ...
 $ Transport: num  3 3 3 3 3 3 1 3 3 3 ...
> names(CarsContinousScaledTransport)
[1] "Age"       "Work.Exp"  "Salary"    "Distance"  "Transport"
```

5.11 Hot code encoding the target variable:

```
#changing the target/predictor variable into numberical
CarsData$Transport=revalue(CarsData$Transport,
                  c("Public Transport"="0", "2Wheeler"="0","Car"="1"))
CarsData$Gender=revalue(CarsData$Gender,
                    c("Female"="1","Male"="0"))
str(CarsData)
```

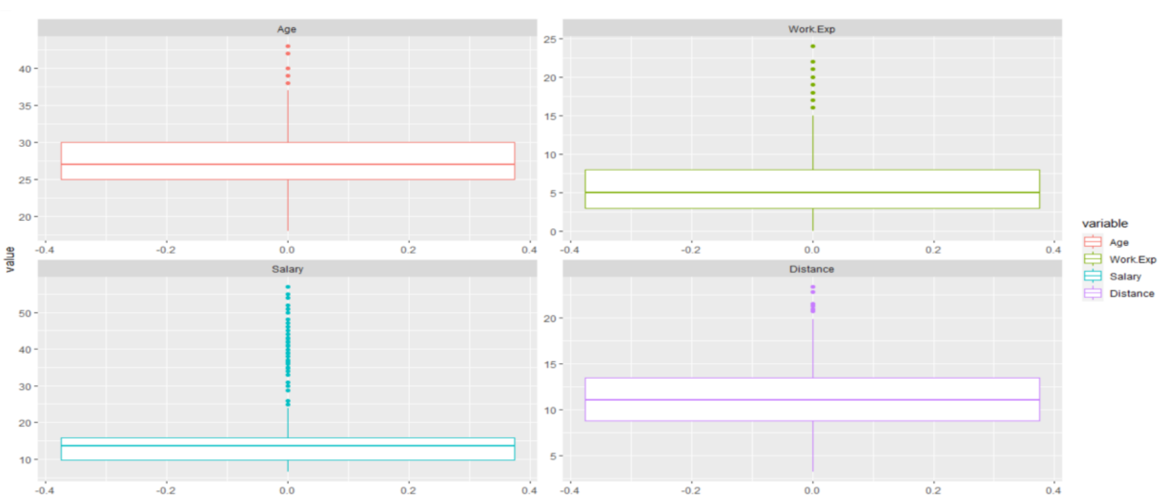
```
> str(CarsData)
'data.frame':   443 obs. of  9 variables:
 $ Age      : int  28 23 29 28 27 26 28 26 22 27 ...
 $ Gender   : Factor w/ 2 levels "1","0": 2 1 2 1 2 2 2 1 2 2 ...
 $ Engineer : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
 $ MBA      : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
 $ 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  : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
 $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

6. Univariate Analysis:

6.1 Continuous variables analysis:

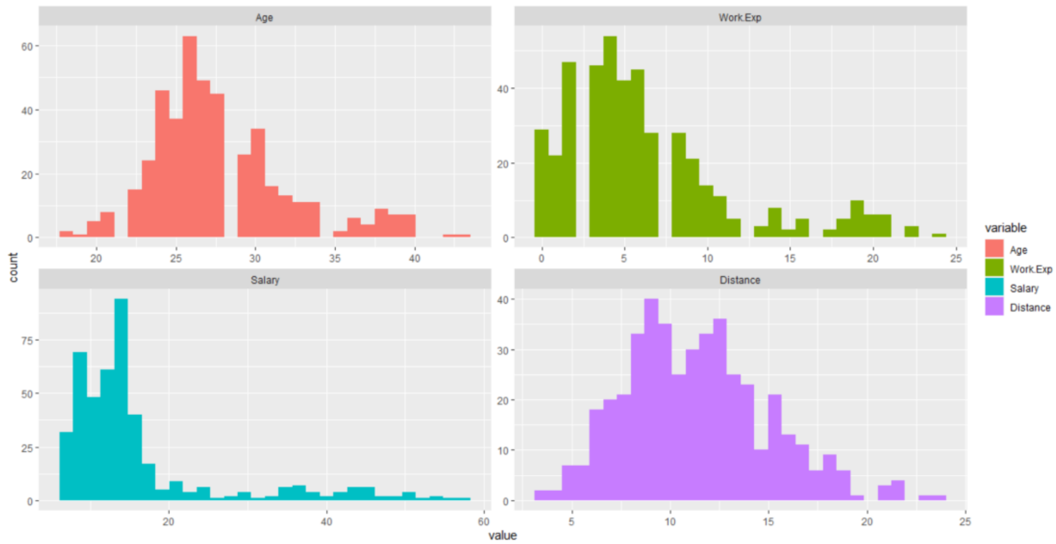
6.1.1 Boxplots:

- Majority of the people are falling in between 25 percentile-50 percentile for Salary.
- For Age and Work Exp, majority of the people are falling in between 50 percentile-75 percentile.
- Distribution of distance on even across from 25 percentile-75 percentile

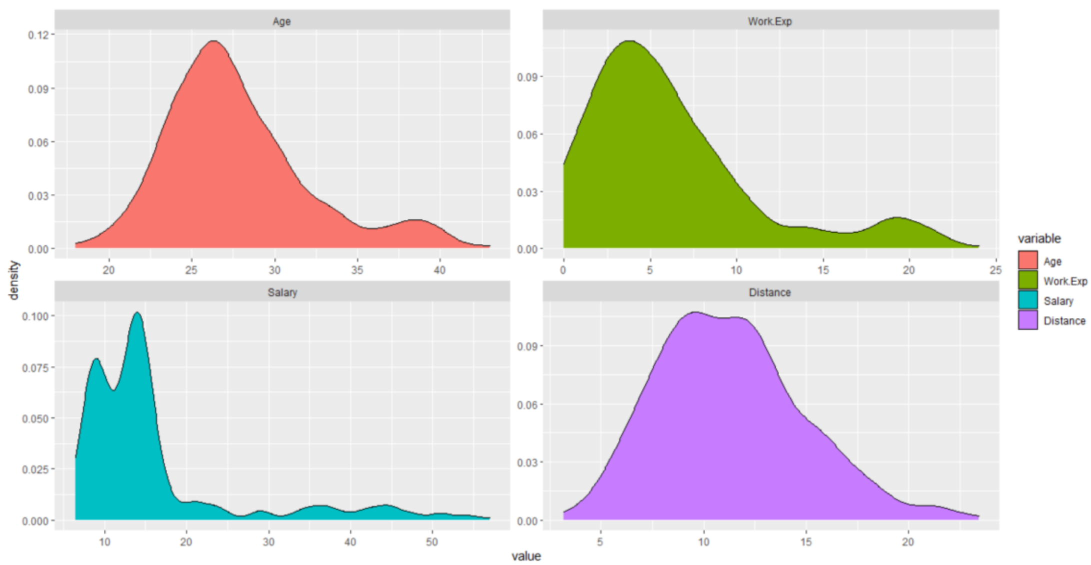


6.1.2 Histograms:

- All the continuous variables are skewed to the right.



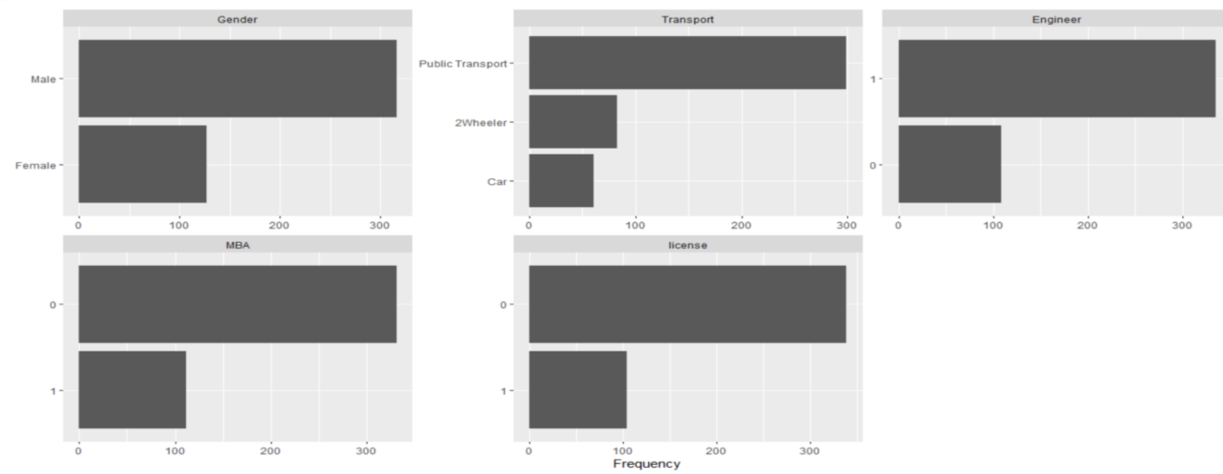
6.1.3 Density plots:



6.2 Categorical variables analysis:

6.2.1 Barplot:

- In Gender, Males are more than females.
- In Transport medium, public transport is widely used than 2 wheeler or car
- Most of the people are engineer graduates than MBA
- Most people don’t have a license.



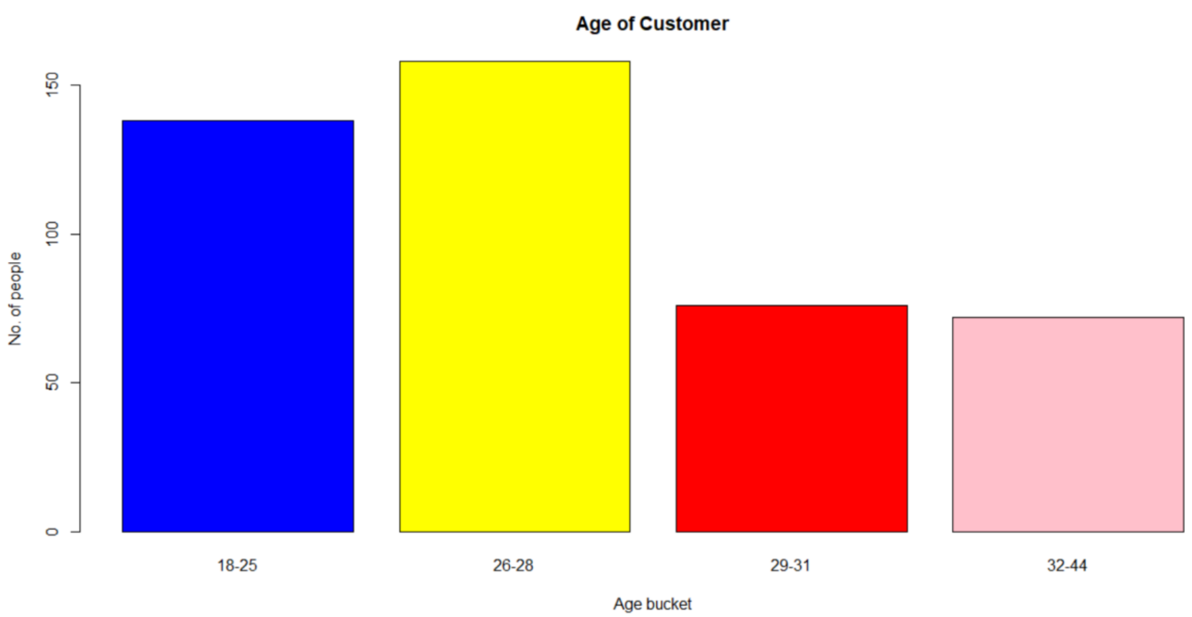
6.2.1 Contingency Table:

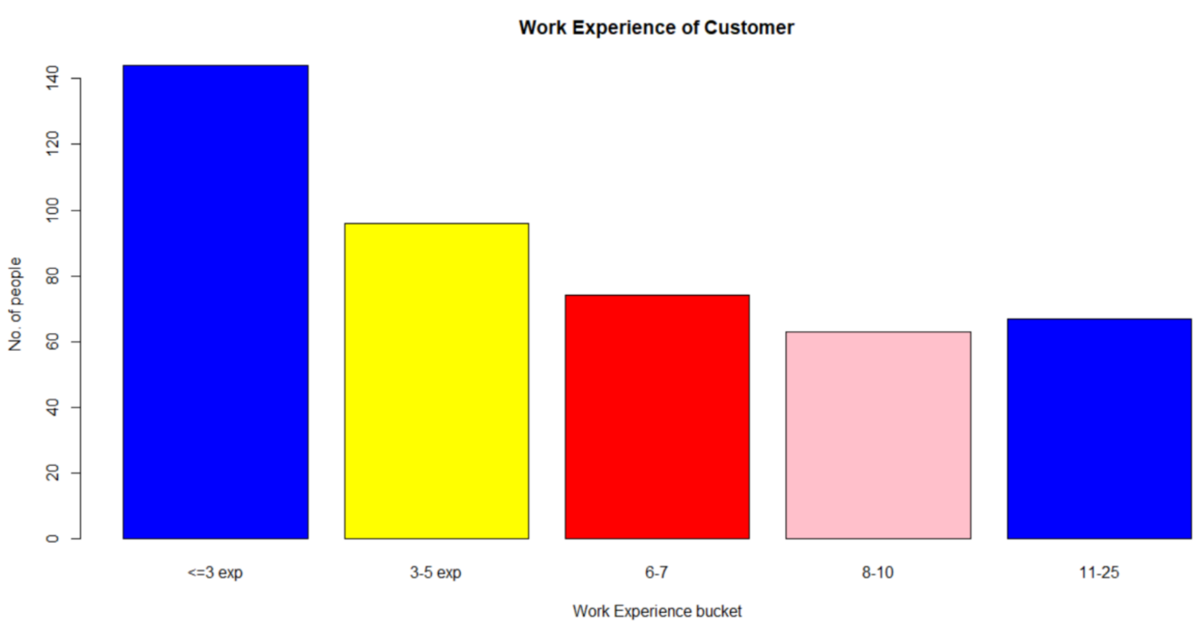
```
> #contingency tables for categorical variables
> variables=names(CarsCategorical)
> for (i in c(1:length(CarsCategorical)) )
+ {
+   print(variables[i])
+   print(table(CarsCategorical[i]))
+   print(round(prop.table(table(CarsCategorical[i])),3))
+ }
[1] "Gender"

Female   Male
   128    316

Female   Male
 0.288  0.712
[1] "Engineer"

  0   1
109 335

    0     1
0.245 0.755
[1] "MBA"

  0   1
331 112

    0     1
0.747 0.253
[1] "license"

  0   1
340 104

    0     1
0.766 0.234
```

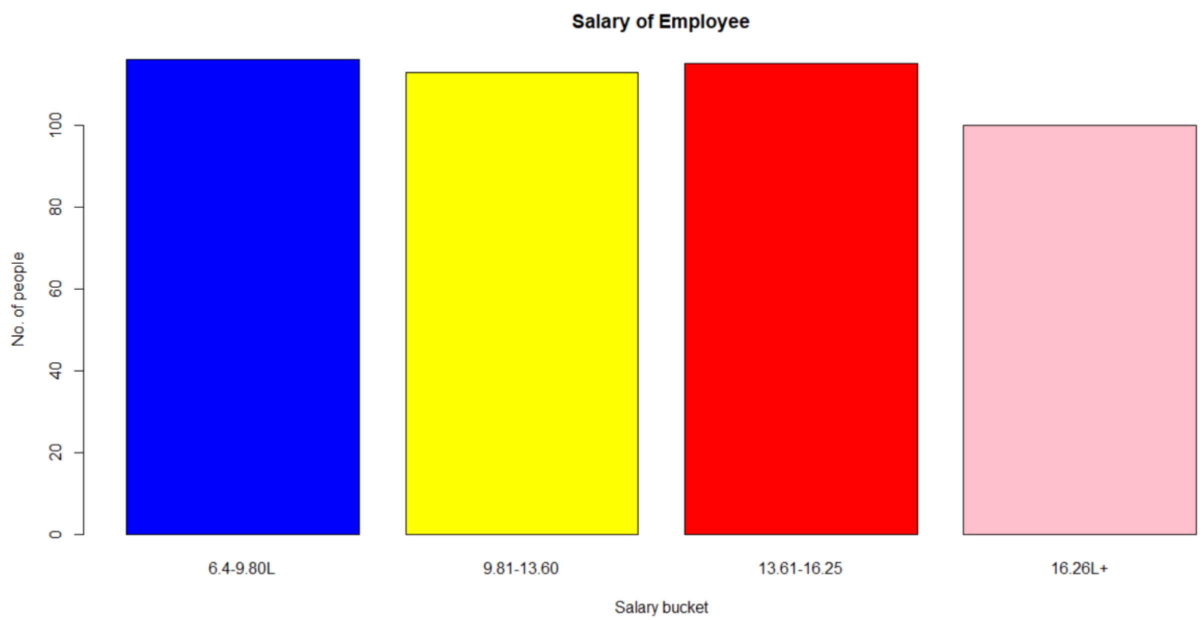
```
[1] "Transport"

2Wheeler      Car Public Transport
     83      61                300

2Wheeler    Car    Public Transport
 0.187      0.137             0.676
```

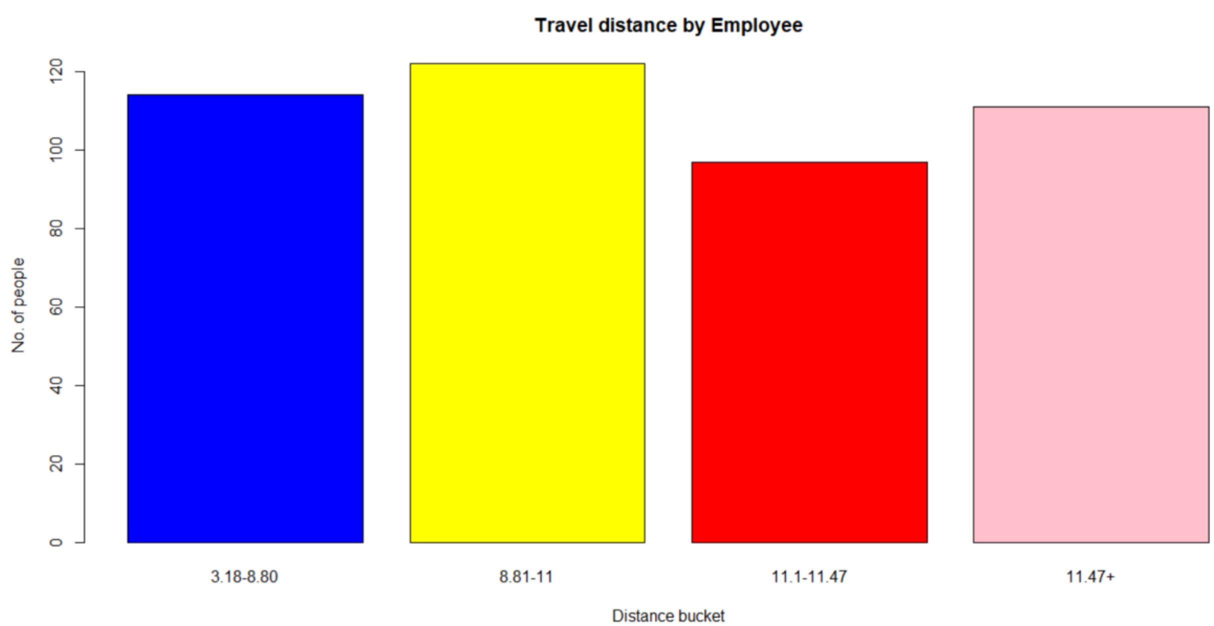
Continuous variables spread:

Continuous variables data spread across the quartile ranges






Salary of Employee
No. of people
0
20
40
60
80
100
6.4-9.80L
9.81-13.60
13.61-16.25
16.26L+
Salary bucket

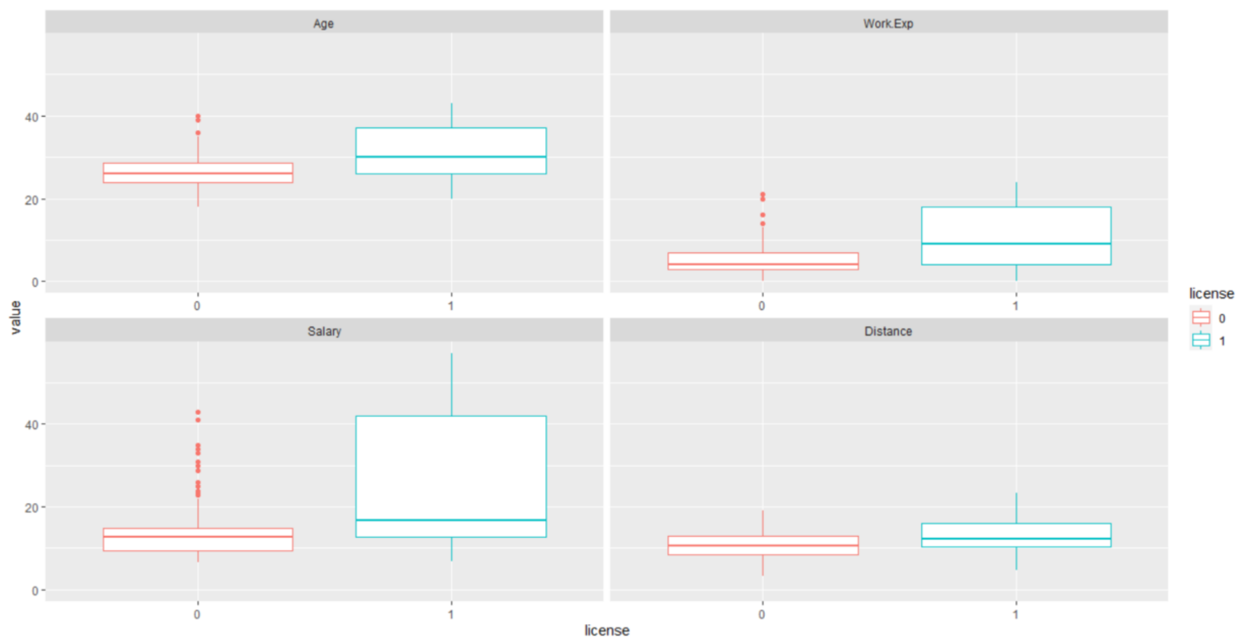

Travel distance by Employee
No. of people
0
20
40
60
80
100
120
3.18-8.80
8.81-11
11.1-11.47
11.47+
Distance bucket

7. Bivariate Analysis:

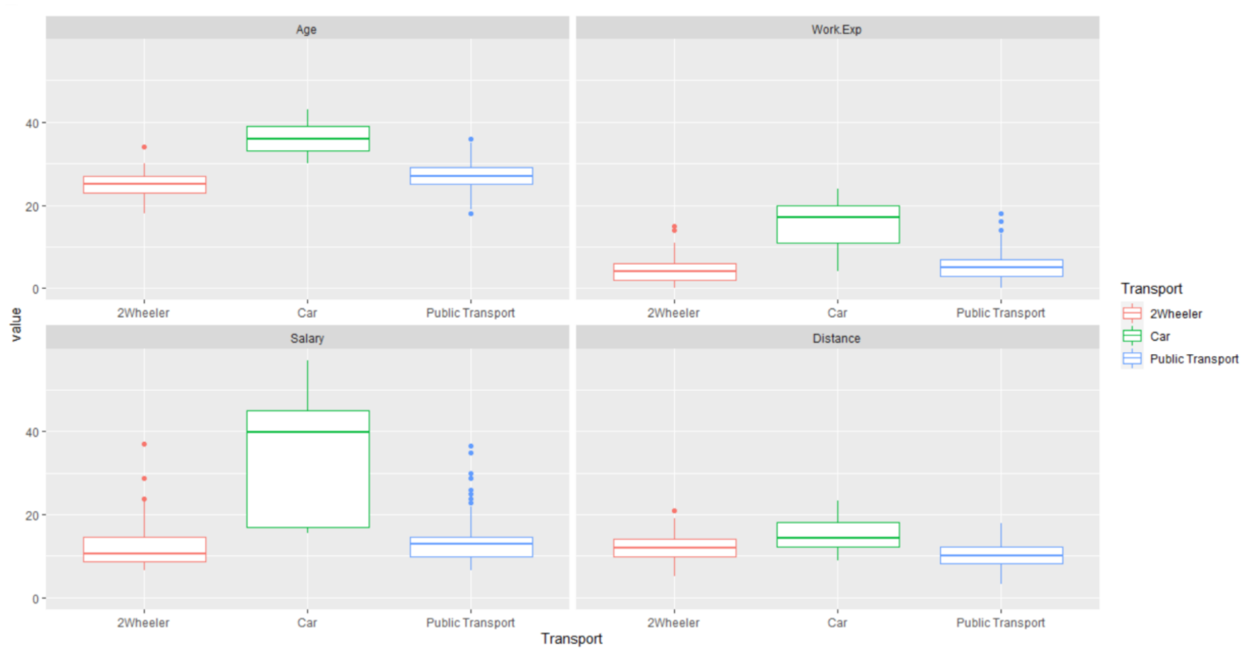
7.1 Continuous variables vs Categorical:

7.1.1 Box plots:

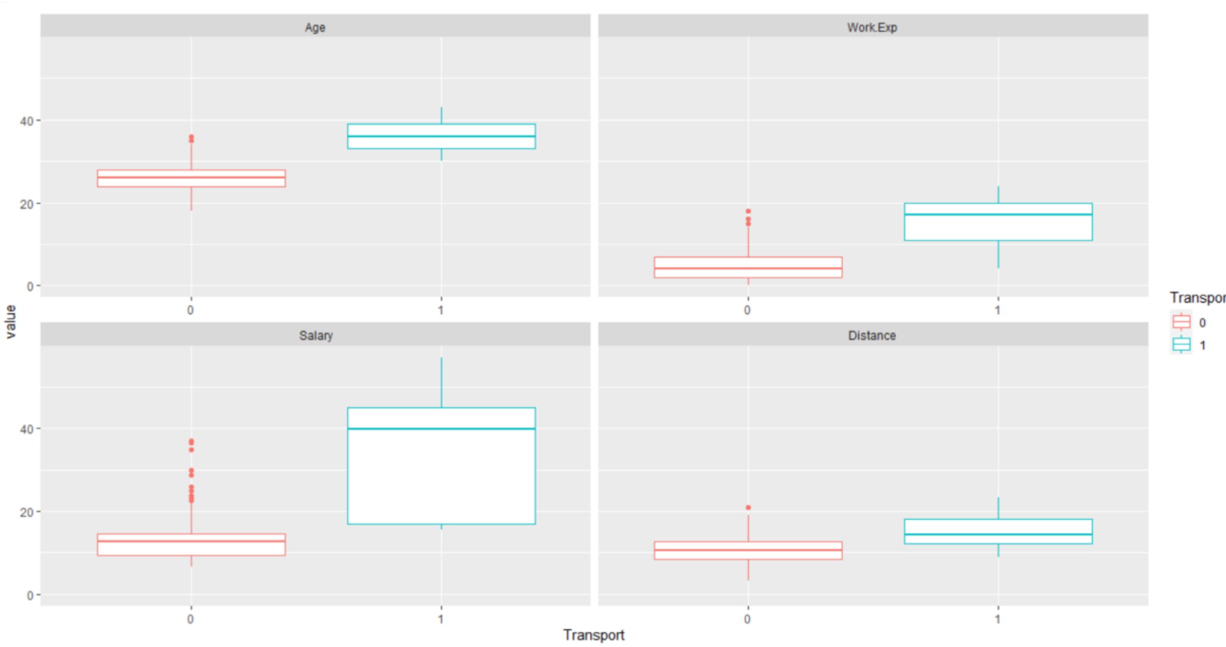
License vs Continuous:



Transport vs Continuous:

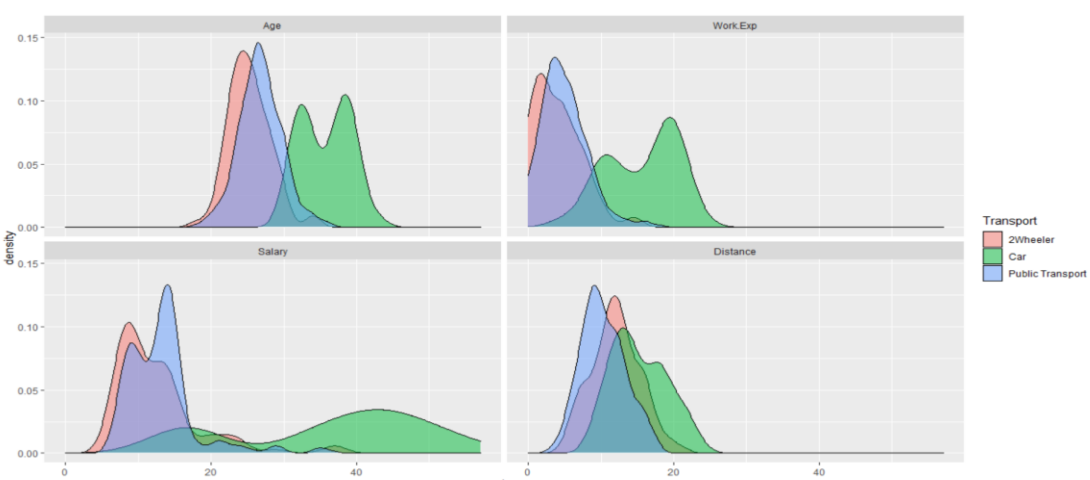


Transport=0 says employee does not use Car as medium of transport. And transport=1 says employee uses car as medium of transport.

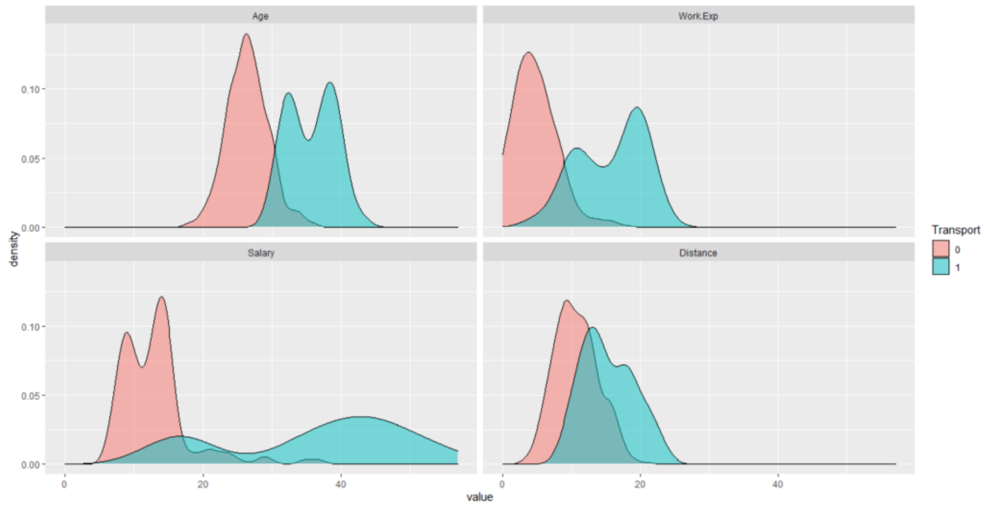


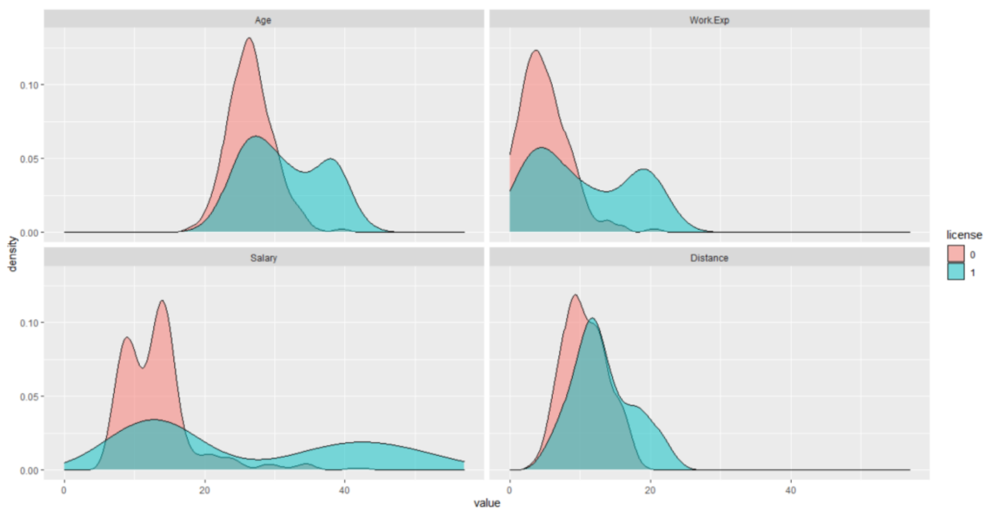
7.1.2 Density plots:

- License is there for the people who are earning more salary
- Cars are mostly used the by people who have age, Salary and work experience relatively more.
- Distance doesnt seem to play a major role, however, public transport was mostly used by people who have salary less than 20k

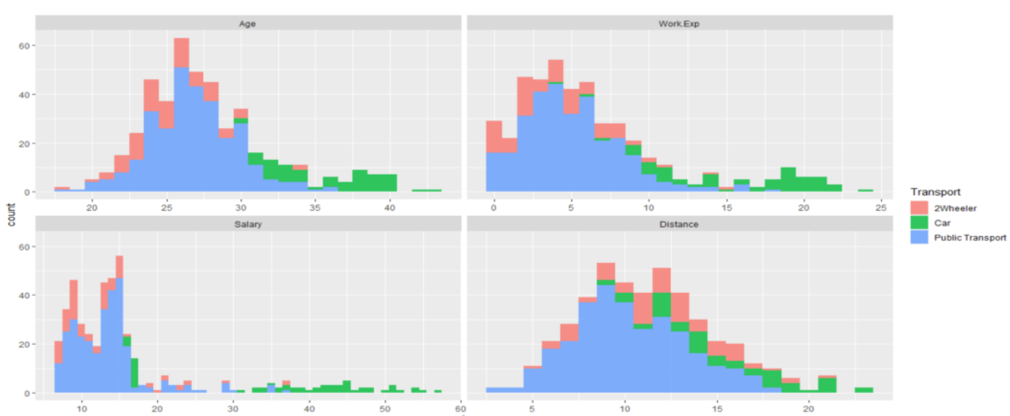


Transport=0 says employee does not use Car as medium of transport. And transport=1 says employee uses car as medium of transport.





7.1.3 Histogram:



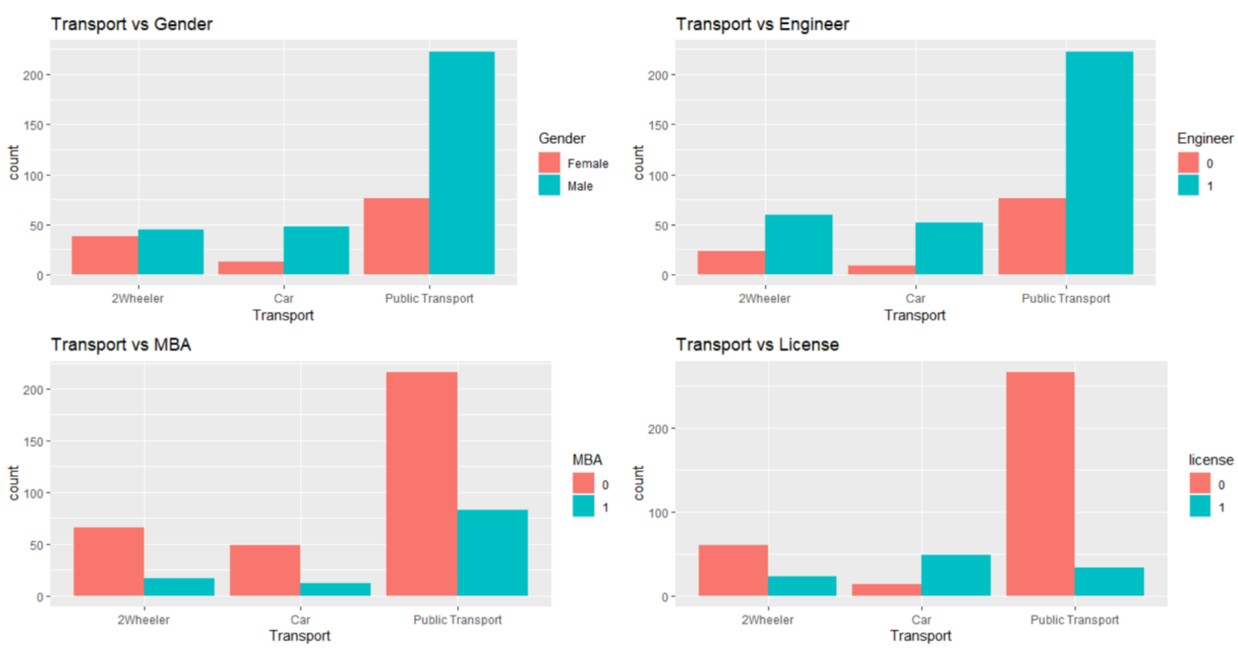
7.1.4 Correlation Plot:

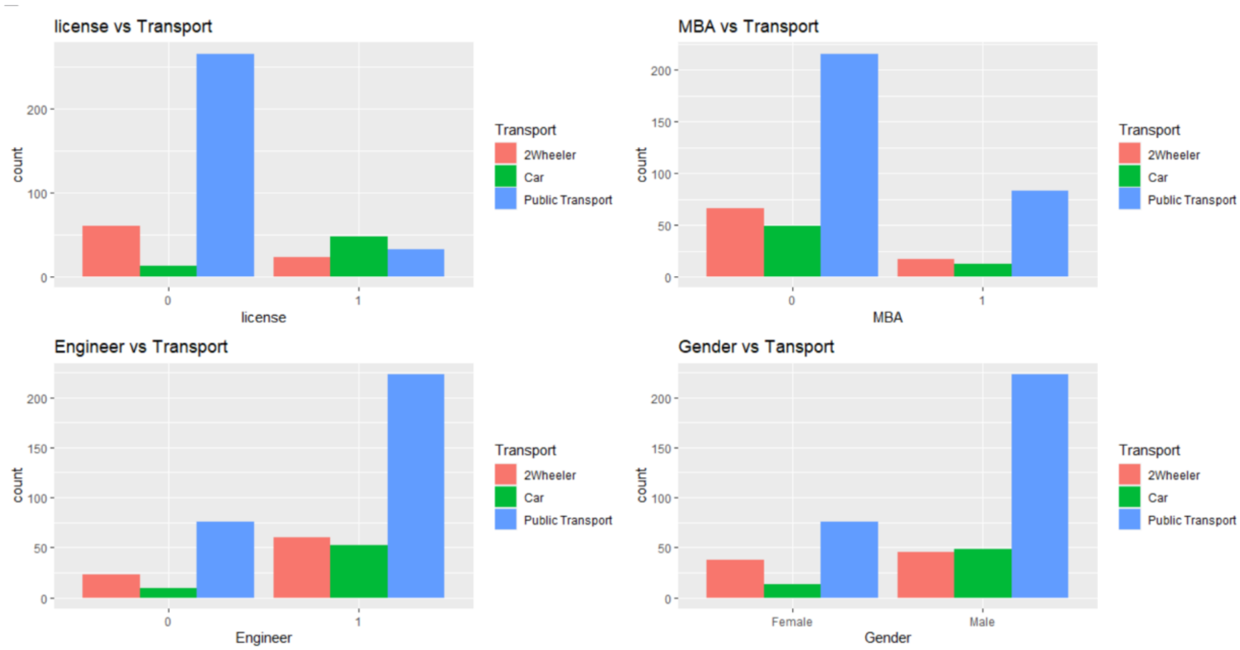
- Age is highly correlated to work experience and moderately correlated to salary
- Work Experience is highly correlated with Salary.
- There is less correlation between distance and Salary.

7.2 **<u>Categorical vs Categorical:</u>**

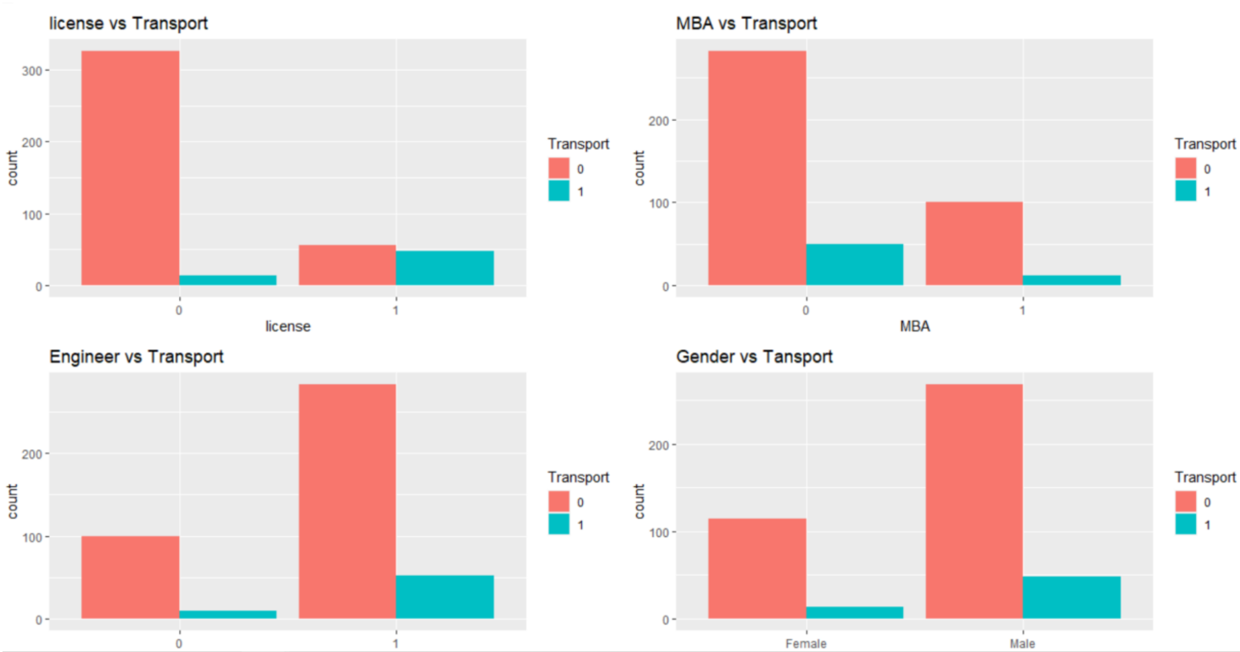
7.2.1 Barplots:

<u>Transport vs Categorical:</u>





Transport=0 says employee does not use Car as medium of transport. And transport=1 says employee uses car as medium of transport.



7.3 **Summary of EDA:**

- In Transport medium, **public transport is widely used** than 2 wheeler or car
- **Age** is highly correlated to **work experience** and moderately correlated to **salary**
- **Work Experience** is highly correlated with **Salary.** There is less correlation between distance and Salary.
- In most cases people who do not have license are the one who are using public transport.
- Cars are mostly used the by people who have more **age**, **License**, **Salary** and **work experience, distance** relatively more.
- Distribution of distance on even across from 25 percentile-75 percentile
- All the continuous variables are skewed to the right.

8. Multicollinearity:

Multi-collinearity can be checked with corrplot, scatterplot and VIF (variable inflation). If change in one variables is causing change in another variable (Directly proportional or inversely proportional), we can deduce that multicollinearity is existing among the variables. We will not be able to exactly narrow down which variable is responsible for predicting if multi-collinearity exists.

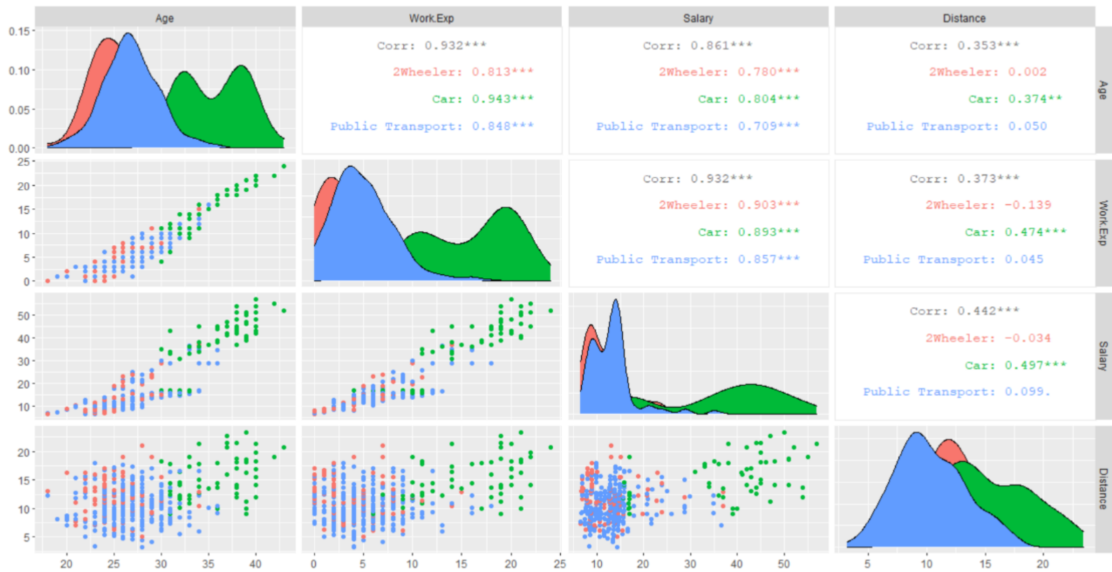
If we build model using all variables, it is showing that high multi-collinearity exists. Below are VIF correlation values:

- 1 = not correlated.
- Between 1 and 5 = moderately correlated.
- Greater than 5 = highly correlated.

```
> vif(glm(Transport~.,data = temp))
      Age  Work.Exp    Salary  Distance
 7.677817 15.312352  8.309499  1.263222
```

8.1 **Multicollinearity Graph:**

- Pairplot shows the density distribution, correlation coefficients for churned and non churned categories.
- There is a linear relationship between Age vs work exp and work exp vs salary.
- There is no correlation between age vs distance, work exp vs distance and



8.2 **Treating Multicollinearity:**

We can test multicollinearity using VIF (Variable Inflation Factor). IF there is multi collinearity, we must drop irrelevant variables and check the VIF again.

Work Experience has no relation with the choice of transport. Hence it can be removed for treating multi collinearity

Multi-collinear Variables	Correlation
Age & Work Experience	0.93
Work Experience and Salary	0.93
Age & Salary	0.86

```
> #removing "Work Experience"
> vif(glm(Transport~.,data = temp[,-2]))
     Age   Salary Distance
3.873928 4.215489 1.247685
```

Permutations of VIF with other continuous variables:

```
> vif(glm(Transport~.,data = temp))
      Age  Work.Exp    Salary  Distance
 7.677817 15.312352  8.309499  1.263222
> vif(glm(Transport~.,data = temp[,-1]))
Work.Exp   Salary Distance
7.726018 8.269677 1.261642
> vif(glm(Transport~.,data = temp[,-2]))
     Age   Salary Distance
3.873928 4.215489 1.247685
> vif(glm(Transport~.,data = temp[,-3]))
     Age Work.Exp Distance
```

```
7.641021 7.768105 1.161713
> vif(glm(Transport~.,data = temp[,-4]))
      Age  Work.Exp    Salary
 7.668210 15.124020  7.641771
```

9. Data Preparation for SMOTE:

The problem statement is to predict whether an employee will use Car as a mode of transport. The given dataset is highly imbalanced. Only 13.7% of the data points there in the minority set and 86.2% in the majority set. This data is highly imbalanced and biased.

As it is biased, our predictions will go wrong and most of the predictions might be biased towards majority set. In such cases we use SMOTE.

SMOTE stands for synthetic minority over-sampling technique.

The current split proportions before SMOTE:

```
> prop.table(table(CarsData$Transport))

        0         1
0.8623025 0.1376975
> prop.table(table(smoteTrain$Transport))

        0         1
0.8581081 0.1418919
> prop.table(table(smoteTest$Transport))

        0         1
0.8707483 0.1292517
```

```
set.seed(1234)
split=sample.split(CarsData,SplitRatio = 0.70)
smoteTrain=subset(CarsData,split==TRUE)
smoteTest=subset(CarsData,split==FALSE)
table(CarsData$Transport)
prop.table(table(CarsData$Transport))
prop.table(table(smoteTrain$Transport))
prop.table(table(smoteTest$Transport))

names(CarsData)
#removed work exp
smoteTrain=smoteTrain[ ,-5]
balancedTrainDataset=SMOTE(Transport~.,data=smoteTrain,
                      perc.over= 800, perc.under=270,k=5 )
table(balancedTrainDataset$Transport)
prop.table(table(balancedTrainDataset$Transport))
```

smoteTrain: Dataset without null values and work exp column

Perc.over= Percentage of oversampling the minority class

Perc.under=Percentage of under sampling the majority class

K= number indicating the number of nearest neighbours that are used to generate the new examples of the minority class.

After SMOTE proportions

```
> table(balancedTrainDataset$Transport)

  0   1 
907 378 
> prop.table(table(balancedTrainDataset$Transport))

        0         1 
0.7058366 0.2941634 
> dim(balancedTrainDataset)
[1] 1285    9
> balancedDataCont=balancedTrainDataset[,(c(1,5,6))]
> balancedDataCat=balancedTrainDataset[,(c(2,3,4,7,8))]
> names(balancedDataCont)
[1] "Age"      "Work.Exp" "Salary"  
> names(balancedDataCat)
[1] "Gender"   "Engineer" "MBA"      "Distance" "license" 
```

After SMOTE is done, the ratio of majority and minority is 70.5%-29.5%. Now the dataset is moderately balanced and moderately biased.

10.Logistic Regression model:

Logistic regression needs to be built basing on the columns which are significant. We use chisq test for checking significance for categorical variables and if they are correlated and univariate regression for continuous variables with our predictor variable churn

10.1 Checking Variables Significance:

Categorical Variables:

Variables	Significant?
Gender	No
Engineer	Yes
MBA	No
license	Yes

```
> for ( i in 1 :(ncol(balancedDataCat)-1)){
+   Statistic <- data.frame(
+     "Row" = colnames(balancedDataCat[5]),
+     "Column" = colnames(balancedDataCat[i]),
+     "Chi SQuare" = chisq.test(balancedDataCat[[5]], balancedDataCat[[i]])$s
tatistic,
+     "df"= chisq.test(balancedDataCat[[5]], balancedDataCat[[i]])$parameter,
```

```
+     "p.value" = chisq.test(balancedDataCat[[5]], balancedDataCat[[i]])$p.va
lue)
+   ChiSqStat <- rbind(ChiSqStat, Statistic)
+ }
> ChiSqStat <- data.table::data.table(ChiSqStat)
> ChiSqStat
         Row   Column Chi.SQuare df      p.value
1: Transport   Gender   1.006475  1 3.157488e-01
2: Transport Engineer   8.551762  1 3.451879e-03
3: Transport      MBA   1.473089  1 2.248589e-01
4: Transport  license 386.194216  1 5.577138e-86
```

From the above p-value output for categorical variables, at alpha 0.05, both the **Gender and MBA** variables seems to be insignificant while the others are significant.

Continuous Variables:

Variables	Significant?
Age	Yes
Salary	Yes
Distance	Yes

At alpha 0.05, all the continuous variables are significant as per the p value.

```
> model=glm(Transport~Age,data = balancedTrainDataset,family=binomial)
> summary(model)

Call:
glm(formula = Transport ~ Age, family = binomial, data = balancedTrainDataset
)

Deviance Residuals: 
     Min        1Q    Median        3Q       Max  
-2.71851  -0.11796  -0.04526   0.05351   2.09919  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) -30.86759    2.14119  -14.42   <2e-16 ***
Age           0.95938    0.06725   14.27   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1557.00  on 1284  degrees of freedom
Residual deviance:  348.66  on 1283  degrees of freedom
AIC: 352.66

Number of Fisher Scoring iterations: 8

> model=glm(Transport~Salary,data = balancedTrainDataset,family=binomial)
> summary(model)
```

```
Call:
glm(formula = Transport ~ Salary, family = binomial, data = balancedTrainData
set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3839  -0.3680  -0.2321   0.1215   2.1419

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.90822    0.28747  -20.55   <2e-16 ***
Salary       0.23549    0.01329   17.72   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1557.0  on 1284  degrees of freedom
Residual deviance:  571.1  on 1283  degrees of freedom
AIC: 575.1

Number of Fisher Scoring iterations: 6

> model=glm(Transport~Distance,data = balancedTrainDataset,family=binomial)
> summary(model)

Call:
glm(formula = Transport ~ Distance, family = binomial, data = balancedTrainDa
taset)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0268  -0.7109  -0.3851   0.6531   2.2795

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.47470    0.37469  -17.28   <2e-16 ***
Distance     0.43934    0.02786   15.77   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1557.0  on 1284  degrees of freedom
Residual deviance: 1145.8  on 1283  degrees of freedom
AIC: 1149.8

Number of Fisher Scoring iterations: 5
```

10.2 Building Logistic Regression Model and Interpretation:

Logistic regression mode is built with continuous and categorical variables in the dataset except below variables. They are removed due to the reasons as follows:

- **MBA was Gender removed** as it turned out to insignificant in chisquare test with transport variable
- **Work Experience** was removed to treat **multi-collinearity**.

Summary and VIF of the model 1:

The logistic regression equation for the model is as follows:

We observe there is a linear relation between the logarithmic probability values with a the variables given below.

Equation: log(y)= -46.30363+(1.20849)***Age**+(0.22641)***Engineer1**

+(-0.01240)***Salary**+(0.50136)***Distance**+(2.59334)***license1**

Where **y= p/(1-p)** and this is called odds ratio, where p is the probability of success

p-value: As per the P-values, at alpha=0.05, except Engineer and salary all are significant

As per the VIF, there is no multi collinearity exhibited in the model.

```
> LogModel=glm(Transport~.,data=LogisticTrain,family="binomial")
> summary(LogModel)

Call:
glm(formula = Transport ~ ., family = "binomial", data = LogisticTrain)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.12660  -0.03724  -0.00780   0.00607   2.29185

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -46.30363    4.76354  -9.720  < 2e-16 ***
Age           1.20849    0.13654   8.851  < 2e-16 ***
Engineer1     0.22641    0.52627   0.430    0.667
Salary       -0.01240    0.02799  -0.443    0.658
Distance      0.50136    0.08502   5.897 3.71e-09 ***
license1      2.59334    0.43239   5.998 2.00e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1557.00  on 1284  degrees of freedom
Residual deviance:  192.32  on 1279  degrees of freedom
AIC: 204.32

Number of Fisher Scoring iterations: 9

> vif(LogModel)
     Age Engineer   Salary Distance  license
2.140975 1.083216 1.700209 1.275159 1.307831
```

```
> logpred=predict(LogModel,LogisticTest,type="response")
> LogModelTable=table(LogisticTest$Transport,logpred>0.5)
> Accuracy = (TP+TN)/nrow(LogisticTest)
> Accuracy
[1] 0.9319728
> sensitivity = TP/(TP+FN)  #Recall
> sensitivity
[1] 0.7368421
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.9609375
> Precision = TP/(TP+FP)
> Precision
[1] 0.7368421
> F1 = 2*(Precision*sensitivity)/(Precision + sensitivity) #Harmonic Mean
> F1
[1] 0.7368421
```

Model 2:

Since **Engineer** and **Salary** are insignificant, another model is built, and predictions are made. We see there is a slight improvement in the model performance

log(y)= -45.30610+1.17945***Age**+(0.48939)***Distance**+(2.59310)***license1**

Where **y= p/(1-p)** and this is called odds ratio, where p is the probability of success

p-value: As per the P-values, at alpha=0.05, all are significant variables used in the below model

As per the VIF, there is no multi collinearity exhibited in the model.

```
> LogModel1=glm(Transport~.,data=LogisticTrain[,-c(2,3)],family="binomial")
> summary(LogModel1)

Call:
glm(formula = Transport ~ ., family = "binomial", data = LogisticTrain[,
    -c(2, 3)])

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.11281  -0.03985  -0.00826   0.00562   2.31993

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -45.30610    4.20416 -10.776  < 2e-16 ***
Age           1.17945    0.11250  10.484  < 2e-16 ***
Distance      0.48939    0.08179   5.984 2.18e-09 ***
license1      2.59310    0.42649   6.080 1.20e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1557.00  on 1284  degrees of freedom
Residual deviance:  192.65  on 1281  degrees of freedom
```

```
AIC: 200.65

Number of Fisher Scoring iterations: 9

> vif(LogModel1)
     Age Distance  license
1.447982 1.157902 1.272419
> logpred1=predict(LogModel1,LogisticTest,type="response")
> LogModelTable1=table(LogisticTest$Transport,logpred1>0.5)
> TP = LogModelTable1[2,2]
> FN = LogModelTable1[2,1]
> FP = LogModelTable1[1,2]
> TN = LogModelTable1[1,1]
> #confusion matrix
> Accuracy = (TP+TN)/nrow(LogisticTest)
> Accuracy
[1] 0.9319728
> sensitivity = TP/(TP+FN)  #Recall
> sensitivity
[1] 0.7368421
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.9609375
> Precision = TP/(TP+FP)
> Precision
[1] 0.7368421
> F1 = 2*(Precision*sensitivity)/(Precision + sensitivity) #Harmonic Mean
> F1
[1] 0.7368421
```

We are trying find out the who use car as mode of transport, i.e. correct prediction of 1 or True Negative rate/specificity. We got 96.03% which is a good measure for our model

Logistic	Prediction	
Actual	FALSE	TRUE
0	123	5
1	5	14

Logistic Regression	Accuracy	Sensitivity	Specificity	Precision	F1 Score	Important variables
Model 1	0.9319728	0.7368421	0.9609375	0.7368421	0.7368421	Age Distance License

11.KNN Model:

KNN refers to K Nearest Neighbor. We predict the response variable using the k-Value. The algorithm will classify the variable into a class for which maximum number is received with the given k-value.

Since we calculate the distance here, we must scale the data so that none of the variables gets influenced over the other.

Scaling the dataset for KNN, Train and Test proportions:

```
> KNNDatasetscale=scale(balancedDataCont)
> KNNTrain=cbind(KNNDatasetscale,balancedDataCat)
> View(KNNDataset)
> head(KNNTrain)
            Age     Salary    Distance Gender Engineer MBA license Transport
93  -0.83758603 -0.6456280 -0.98937001      0        1   0       0         0
243 -0.64496851 -0.5666347 -0.10572106      0        0   0       1         0
13  -1.03020355 -0.8905073 -1.79017686      0        1   0       0         0
133 -0.45235099 -0.5745340 -0.76845777      0        0   0       0         0
234 -0.06711594  0.2311979 -0.16094912      0        1   0       1         0
270  0.12550158  0.7999498  0.05996312      0        0   0       0         0
> KNNTest=smoteTest
> prop.table(table(KNNTrain$Transport))

        0         1 
0.7058366 0.2941634 
> prop.table(table(KNNTest$Transport))

        0         1 
0.8707483 0.1292517 
```

Building KNN model with various K-values:

Model	Accuracy	Sensitivity	Specificity
KNN Model	0.1293	NA	0.1293

K-Value	Accuracy	Sensitivity	Specificity
35	0.1293	NA	0.1293
37	0.1293	NA	0.1293
39	0.1293	NA	0.1293
33	0.1293	NA	0.1293

k=35	Prediction	
Actual	0	1
0	0	128
1	0	19

Positive class is taken as “0” by default. If employee use car as mode of transport, it is denoted as 1. As per the problem statement, we must predict. Hence, we can focus on achieving good True Negative/ Sensitivity.

At k-value =35:

k=35	Prediction	
Actual	0	1
0	0	128
1	0	19

```
> confusionMatrix(table(KNNTest$Transport,KNNModel))
Confusion Matrix and Statistics

   KNNModel
      0   1
  0   0 128
  1   0  19

               Accuracy : 0.1293
                 95% CI : (0.0796, 0.1945)
    No Information Rate : 1
    P-Value [Acc > NIR] : 1

                  Kappa : 0

 Mcnemar's Test P-Value : <2e-16

            Sensitivity :     NA
            Specificity : 0.1293
         Pos Pred Value :     NA
         Neg Pred Value :     NA
             Prevalence : 0.0000
         Detection Rate : 0.0000
   Detection Prevalence : 0.8707
      Balanced Accuracy :     NA

       'Positive' Class : 0
```

At k-Value=37:

k=37	Prediction	
Actual	0	1
0	0	128
1	0	19

```
> confusionMatrix(table(KNNTest$Transport,KNNModel))
Confusion Matrix and Statistics

   KNNModel
      0   1
  0   0 128
  1   0  19

               Accuracy : 0.1293
                 95% CI : (0.0796, 0.1945)
    No Information Rate : 1
    P-Value [Acc > NIR] : 1

                  Kappa : 0

 Mcnemar's Test P-Value : <2e-16

            Sensitivity :     NA
```

```
           Specificity : 0.1293
        Pos Pred Value :     NA
        Neg Pred Value :     NA
            Prevalence : 0.0000
        Detection Rate : 0.0000
  Detection Prevalence : 0.8707
     Balanced Accuracy :     NA

      'Positive' Class : 0
```

At k-Value=39

k=39	Prediction	
Actual	0	1
0	0	128
1	0	19

```
> confusionMatrix(table(KNNTest$Transport,KNNModel))
Confusion Matrix and Statistics

   KNNModel
      0   1
  0   0 128
  1   0  19

               Accuracy : 0.1293
                 95% CI : (0.0796, 0.1945)
    No Information Rate : 1
    P-Value [Acc > NIR] : 1

                  Kappa : 0

 Mcnemar's Test P-Value : <2e-16

            Sensitivity :     NA
            Specificity : 0.1293
         Pos Pred Value :     NA
         Neg Pred Value :     NA
             Prevalence : 0.0000
         Detection Rate : 0.0000
   Detection Prevalence : 0.8707
      Balanced Accuracy :     NA

       'Positive' Class : 0
```

12.NaiveBayes Model:

NaiveBayes cannot be directly build on the dataset.

The main assumption on NaiveBayes model is **conditional independence**. i.e the variables in the dataset are totally independent and not correlated to each other. But in our dataset we see there is a correlation between the variables as below. Hence we **drop the column "Work Experience"** and then build the NB model.

Multi-collinear Variables	Correlation
Age & Work Experience	0.93
Work Experience and Salary	0.93
Age & Salary	0.86

Train and test dataset proportions:

```
> NBTrain=balancedTrainDataset
> NBTest=smoteTest
> prop.table(table(NBTrain$Transport))

        0         1
0.7058366 0.2941634
> prop.table(table(NBTest$Transport))

        0         1
0.8707483 0.1292517
> set.seed(1234)
> dim(NBTrain)
[1] 1285    8
> dim(NBTest)
[1] 147   8
```

NaiveBayes Output:

```
> NBModel

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
        0         1
0.7058366 0.2941634

Conditional probabilities:
   Age
Y       [,1]     [,2]
  0 26.56560 2.942204
  1 36.02579 2.767304

   Gender
Y           1         0
  0 0.3020948 0.6979052
  1 0.2883598 0.7116402

   Engineer
```

```
Y           0         1
  0 0.2668137 0.7331863
  1 0.1455026 0.8544974

   MBA
Y           0         1
  0 0.7552370 0.2447630
  1 0.7169312 0.2830688

   Salary
Y        [,1]      [,2]
  0 13.04112  5.072831
  1 35.92661 10.644713

   Distance
Y        [,1]     [,2]
  0 10.61632 3.090111
  1 14.92187 2.932210

   license
Y           0         1
  0 0.8500551 0.1499449
  1 0.3015873 0.6984127
```

Confusion Matrix for NaiveBayes:

NaiveBayes	Prediction	
Actual	0	1
0	123	5
1	6	13

Model	Accuracy	Sensitivity	Specificity
Naïve Bayes	0.9252	0.9535	0.7222

```
> confusionMatrix(NBTable)
Confusion Matrix and Statistics

   NBPred
      0   1
  0 123   5
  1   6  13

               Accuracy : 0.9252
                 95% CI : (0.8701, 0.9621)
    No Information Rate : 0.8776
    P-Value [Acc > NIR] : 0.04438

                  Kappa : 0.6599

 Mcnemar's Test P-Value : 1.00000

            Sensitivity : 0.9535
            Specificity : 0.7222
```

```
         Pos Pred Value : 0.9609
         Neg Pred Value : 0.6842
             Prevalence : 0.8776
         Detection Rate : 0.8367
   Detection Prevalence : 0.8707
      Balanced Accuracy : 0.8379

       'Positive' Class : 0
```

13.Confusion Matrix and validation Exercise :

Confusion Matrix is one of the model performances measures to check how well our model is fitting the test or new data.

Important Measures of Confusion Matrix: Sensitivity, Specificity, Accuracy

Sensitivity: Also called as True positive rate or Recall. This is proportion of actual positive cases which are correctly identified. TP/(TP+FN)

Specificity: Also called as True Negative rate or False Positive rate. This is proportion of negatives that were correctly identified. TN/(TN+FP)

Accuracy: 1-error rate. This is how many correct predictions are done in both classes. Error rate: FP+FN/(TP+TN+FP+FN)

The confusion matrices which we made considered positive rate as '0'. From the cars data employee who use car as mode of transport are marked as '1'. Hence, we are looking at "**True Negativity**" or "**Specificity**" as our measure of interest.

Below are the metric comparisons for confusion matrices for various models. Out of all 3 models, logistic regression performed best, followed by NaiveBayes and then KNN.

Model Name	Accuracy	Sensitivity	Specificity
Logistic	0.931973	0.736842	0.960938
KNN Model	0.1293	NA	0.1293
Naïve Bayes	0.9252	0.9535	0.7222
Bagging	0.9524	0.9764	0.81
Gradient Boosting	0.959184	0.842105	0.976563
XGBoost	0.972789	0.894737	0.984375

KNN: KNN works on classifying the data point based on the nearest neighbour. Imbalanced set will have more of majority set and KNN is influenced by it. After performing SMOTE, though the proportion of train dataset is made 70-30, majority class is still influencing the classification output. On the other hand, we cannot make the dataset highly balanced, because model will be constructed on synthetic data and not on the original data. Hence it cannot be reliable. Hence all the confusion matrix metrics are quite low.

Logistic Regression: None of the continuous variables here are normally distributed. Once of the important assumption of logistic regression is that variables need not follow normal distribution. Hence this model performed better than others. Since the model is built on the significant variables, it is an added advantage.

NaiveBayes:

- **Conditional independence** is basic assumption of NaiveBayes. All the variables should be totally independent for this model to perform good.
- NaiveBayes can only predict only basing on the history data. If **incoming value is a new** one, this model **will fail to predict it right**
- Since the dataset is highly imbalanced, the specificity is low when compared to other models

Compared to KNN, Naïve Bayes performed better (next to logistic regression) using conditional independence to its advantage. 72% of the minority class and 95% of the majority class predictions are done right.

14.Bagging:

Bagging stands for bootstrap aggregation. This is one of the ensemble methods. Bagging mostly reduces the high variance and retains a bit of bias in the data. This model works in parallel and multiple trees are created with various bootstrap samples. The final prediction is made basing on the mode of the output value by tree in case of classification method. Since there is a striking balance between bias-variance, we notice a good performance of the model from the confusion matrix.

Bagging Train and Test dataset proportions:

```
> baggingData=balancedTrainDataset
> baggingTrain=balancedTrainDataset
> baggingTrest=smoteTest
> baggingTest=smoteTest
> dim(baggingTrain)
[1] 1285    8
> dim(baggingTest)
[1] 147   8
> prop.table(table(baggingTrain$Transport))

        0         1 
0.7058366 0.2941634 
> prop.table(table(baggingTest$Transport))

        0         1 
0.8707483 0.1292517 
```

Bagging model building:

```
> baggingModel <- bagging(Transport ~.,
+                          data=baggingTrain,
+                          control=rpart.control(maxdepth=10, minsplit=4))
> baggingModel

Bagging classification trees with 25 bootstrap replications

Call: bagging.data.frame(formula = Transport ~ ., data = baggingTrain,
    control = rpart.control(maxdepth = 10, minsplit = 4))
```

Prediction and confusion Matrix:

```
> baggingPred <- predict(baggingModel,baggingTest)
> baggingTable=table(baggingTest$Transport,baggingPred)
> confusionMatrix(baggingTable)
```

Bagging	Prediction	
Actual	0	1
0	124	4
1	3	16

Confusion Matrix for bagging:

```
Confusion Matrix and Statistics

   baggingPred
      0   1
  0 124   4
  1   3  16

               Accuracy : 0.9524
                 95% CI : (0.9043, 0.9806)
    No Information Rate : 0.8639
    P-Value [Acc > NIR] : 0.0004008

                  Kappa : 0.7931

 Mcnemar's Test P-Value : 1.0000000

            Sensitivity : 0.9764
            Specificity : 0.8000
         Pos Pred Value : 0.9688
         Neg Pred Value : 0.8421
             Prevalence : 0.8639
         Detection Rate : 0.8435
   Detection Prevalence : 0.8707
      Balanced Accuracy : 0.8882

       'Positive' Class : 0
```

15.Boosting:

Boosting works in sequential mode. The first algorithm is trained on the entire dataset and the subsequent algorithms are built by fitting the residuals of the first algorithm, thus giving higher weight to those observations that were poorly predicted by the previous model.

Boosting mostly treats bias and causes over fitting. Hence tunings is important here.

Boosting Train and test Dataset Proportions:

```
> boostingTrain=balancedTrainDataset
> boostingTest=smoteTest
> prop.table(table(boostingTrain$Transport))

        0         1 
0.7058366 0.2941634 
> prop.table(table(boostingTest$Transport))

        0         1 
0.8707483 0.1292517 
```

Building Boosting Model:

Distribution=”bernouli” since we are predicting a classification variable with 2 outputs

n.trees= number of trees to grow

Interaction depth=integer specifying the maximum depth of each tree (i.e., the highest level of variable interactions allowed). A value of 1 implies an additive model, a value of 2 implies a model with up to 2-way interactions, etc. Default is 1.

shrinkage parameter applied to each tree in the expansion. Also known as the learning rate or step-size reduction; 0.001 to 0.1 usually work, but a smaller learning rate typically requires more trees. Default is 0.1.

cv folds: Number of cross-validation folds to perform. If cv.folds>1 then gbm, in addition to the usual fit, will perform a cross-validation, calculate an estimate of generalization error returned in cv.error.

Verbose: Logical indicating whether or not to print out progress and performance indicators (TRUE). If this option is left unspecified for gbm.more, then it uses verbose from object. Default is FALSE.

n.cores: The number of CPU cores to use.

```
> boostingTrainfactor=boostingTrain
> boostingTrainfactor$Transport=unfactor(boostingTrainfactor$Transport)
> View(boostingTrainfactor)
> boostingModel<- gbm(
+   formula = Transport ~ .,
+  distribution = "bernoulli",
+  data = boostingTrainfactor,
```

```
+  n.trees = 1000,
+  interaction.depth = 1,
+  shrinkage = 0.01,
+  cv.folds = 5,
+  n.cores = NULL, # will use all cores by default
+  verbose = FALSE
+ )
```

Prediction and confusion Matrix:

Gradient Boosting	Prediction	
Actual	0	1
0	125	3
1	3	16

```
> boostingPred <- predict(boostingModel, boostingTest, type = "response")
Using 1000 trees...
> table(boostingTest$Transport,boostingPred>0.5)

    FALSE TRUE
  0   125    3
  1     3   16
```

Metrics for confusion Matrix:

Model	Accuracy	Sensitivity	Specificity
Gradient Boosting	0.959184	0.842105	0.976563

```
> #confusion matrix
> Accuracy = (TP+TN)/nrow(boostingTest)
> Accuracy
[1] 0.9591837
> sensitivity = TP/(TP+FN)  #Recall
> sensitivity
[1] 0.8421053
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.9765625
> Precision = TP/(TP+FP)
> Precision
[1] 0.8421053
> F1 = 2*(Precision*sensitivity)/(Precision + sensitivity) #Harmonic Mean
> F1
[1] 0.8421053
```

Extreme Gradient Boosting:

To implement XGboost, the data should be inputed into algorithm in terms of a matrix. It works on numerical data, hence matrix conversion is mandatory. If there are any character categorical variables, hot encoding should be done to change them into numeric. This also works similar to

GBM with an additional improved performance. It takes the 1st derivative of the error produced during the computation.

Converting Train and test Data numerical matrix for boosting:

```
> features_train<-as.matrix(boostingTrain[,1:8])
> label_train<-as.matrix(boostingTrain[,8])
> features_test<-as.matrix(boostingTest[,1:8])
> features_train=apply(features_train, 2, as.numeric)
> label_train=apply(label_train, 2, as.numeric)
> features_test=apply(features_test, 2, as.numeric)
```

Building model and choosing best number for iterations:

```
> #model building
> xgb.fit <- xgboost(
+   data = features_train,
+   label = label_train,
+   eta = 0.7,
+   max_depth = 5,
+   nrounds = 20,
+   nfold = 5,
+   objective = "binary:logistic",  # for regression models
+   verbose = 1,               # silent,
+   early_stopping_rounds = 20 # stop if no improvement for 10 consecutive tr
ees
+ )
[15:49:14] WARNING: amalgamation/../src/learner.cc:480:
Parameters: { nfold } might not be used.

  This may not be accurate due to some parameters are only used in language b
indings but
  passed down to XGBoost core.  Or some parameters are not used but slip thro
ugh this
  verification. Please open an issue if you find above cases.


[1]     train-error:0.008560
Will train until train_error hasn't improved in 20 rounds.

[2]     train-error:0.003113
[3]     train-error:0.001556
[4]     train-error:0.000778
[5]     train-error:0.000778
[6]     train-error:0.000778
[7]     train-error:0.000778
[8]     train-error:0.000000
[9]     train-error:0.000000
[10]    train-error:0.000000
[11]    train-error:0.000000
[12]    train-error:0.000000
[13]    train-error:0.000000
[14]    train-error:0.000000
[15]    train-error:0.000000
```

We see the error stopped reducing after 8 iterations.

Prediction and confusion matrix:

XGBoost	Prediction	
Actual	0	1
0	126	2
1	2	17

```
> predXgb <- predict(xgb.fit, features_test)
> xgbTable=table(boostingTest$Transport,predXgb>=0.5)
> xgbTable

    FALSE TRUE
  0   126    2
  1     2   17
```

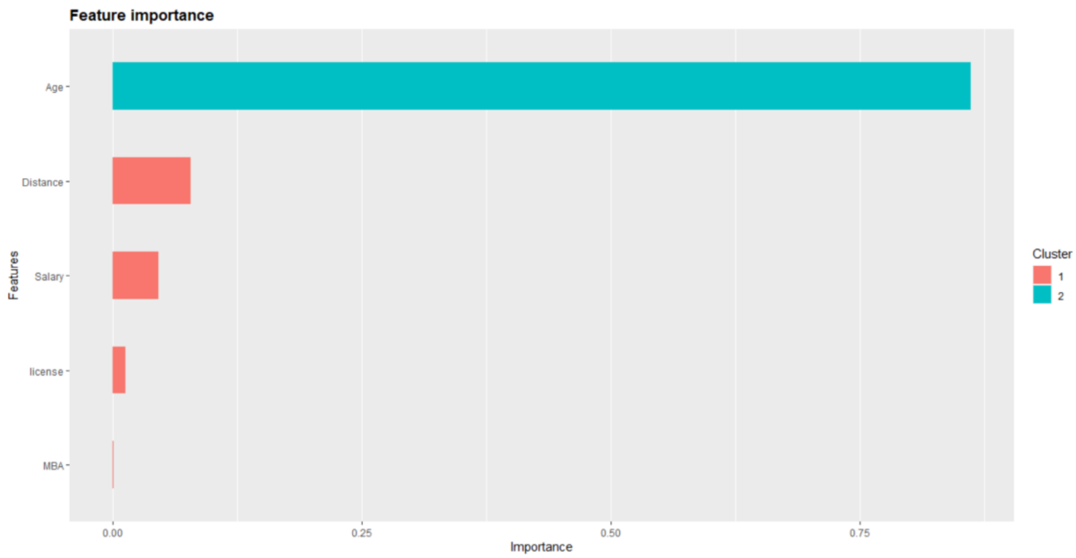
Confusion Matrix metrics:

Model	Accuracy	Sensitivity	Specificity
XGBoost	0.972789	0.894737	0.984375

```
> #confusion matrix
> Accuracy = (TP+TN)/nrow(features_test)
> Accuracy
[1] 0.9727891
> sensitivity = TP/(TP+FN)  #Recall
> sensitivity
[1] 0.8947368
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.984375
> Precision = TP/(TP+FP)
> Precision
[1] 0.8947368
> F1 = 2*(Precision*sensitivity)/(Precision + sensitivity) #Harmonic Mean
> F1
[1] 0.8947368
```

Variable Importance:

```
> impMatrix=xgb.importance(model=xgb.fit)
> impMatrix
    Feature        Gain       Cover   Frequency
1:      Age 0.861495519 0.371699009 0.235849057
2: Distance 0.078357953 0.264487826 0.339622642
3:   Salary 0.045910993 0.300655283 0.311320755
4:  license 0.013151310 0.058733663 0.103773585
5:      MBA 0.001084225 0.004424219 0.009433962
> xgb.ggplot.importance(importance_matrix = impMatrix,
+                       model = xgb.fit)
```



Age is the driving factor for an employee to choose car as mode of transport. This is followed by **distance, Salary and License**. MBA is contributing very little amount for prediction and other variables are negligible.

16.Actionable Insights and Recommendations:

Employees who have more experience tends to have more age and earn more salary. Cars are affordable and used as mode of transport by employee who are earning more **salary**.

Employees whose **salary** is more than 17L approx. are opting for Car as mode of transport. This could be since they are the group of audience who can afford a Car

License is another driving variable for prediction of employee who are using car as mode of transport. We see 76% of the employees who does not have license are opting for public transport

While **distance** plays a little role, people travelling more than 20 KMs of distance are choosing car as mode of transport as it provides comfort.

*******************************THE END***********************************