Project - Cars Case Study

Cars Case Study

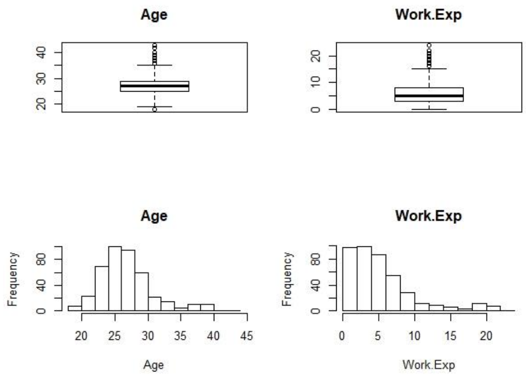
This project requires you to understand what mode of transport employees prefers to commute to their office. The dataset "Cars-dataset" includes employee information about their mode of transport as well as their personal and professional details like age, salary, work exp. We need to predict whether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

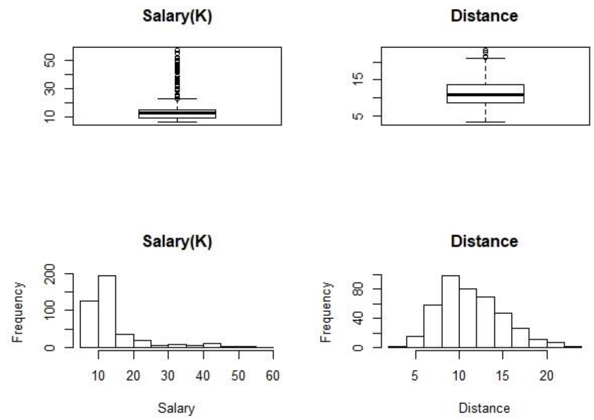
Cars Dataset Data Dictionary

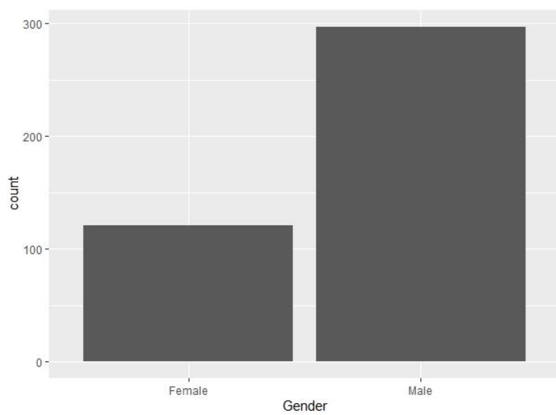
AGE	Age of the employee
GENDER	Gender of employee
ENGINEER	Does employee have Engineering Degree. 1 indicates employee has engineering degree 0 indicates employee doesn’t
MBA	Does employee have MBA Degree. 1 indicates employee has MBA degree 0 indicates employee doesn’t
WORK EXP	Work experience in years
SALARY	Annual Salary of employee (in thousand)
DISTANCE	Distance from office (in KM)
LISCENSE	Does employee have license
TRANSPORT	Modes of transport chosen by employee

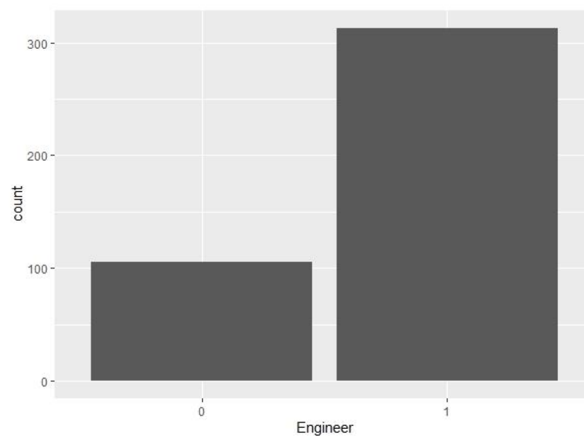
Table of Contents

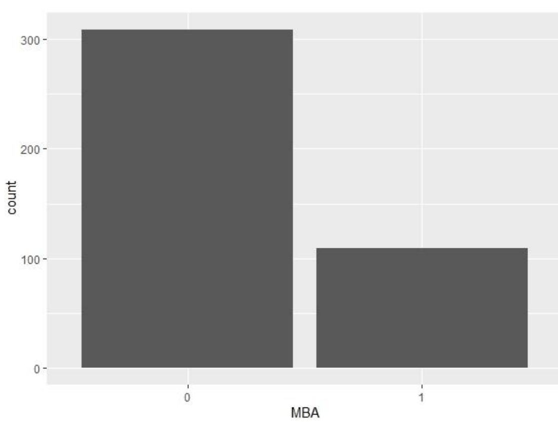
Perform an EDA on the data

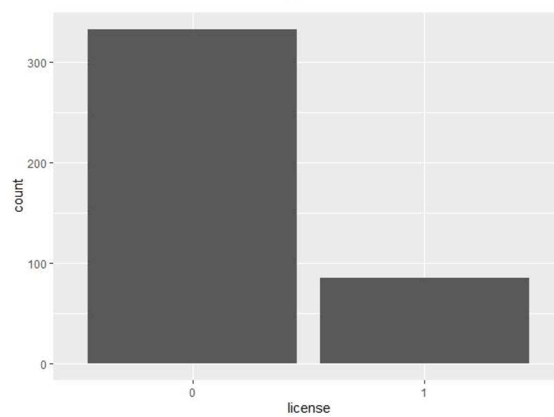


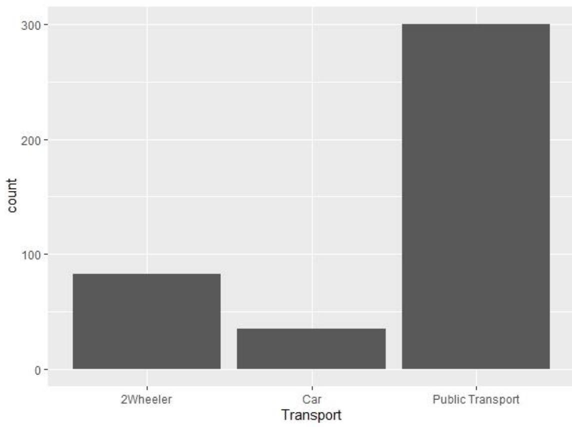


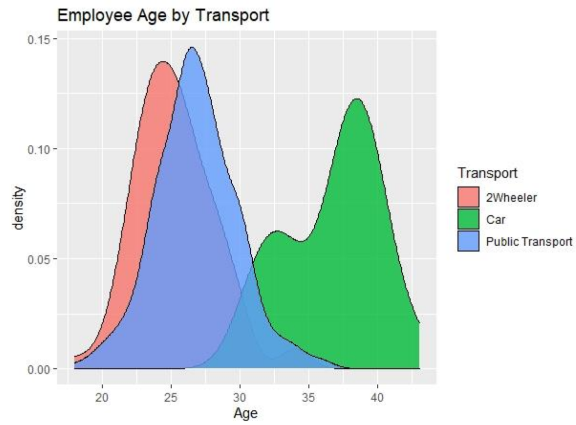


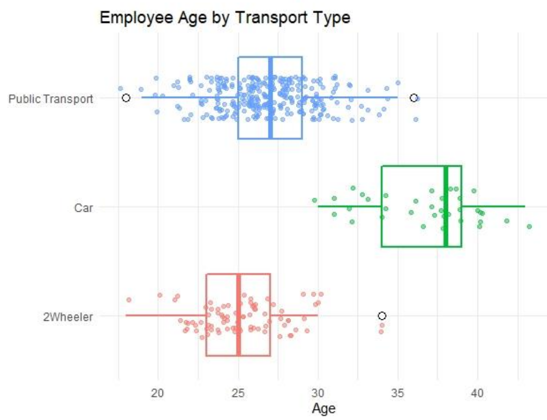


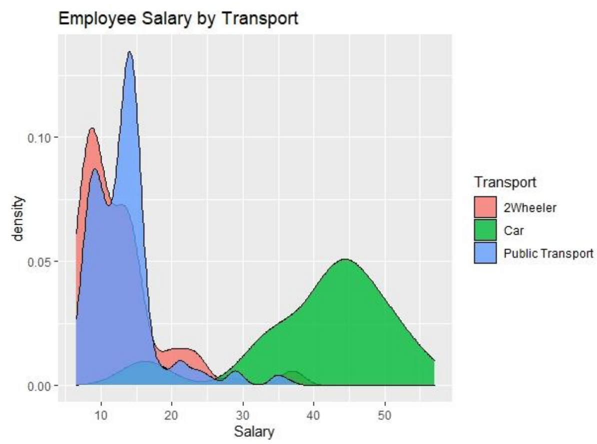


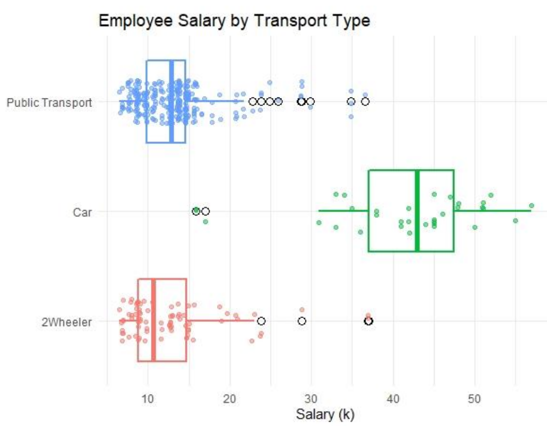


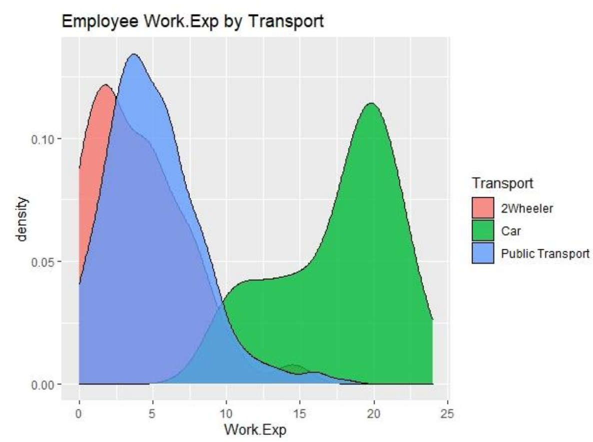

count
Transport
2Wheeler
Car
Public Transport

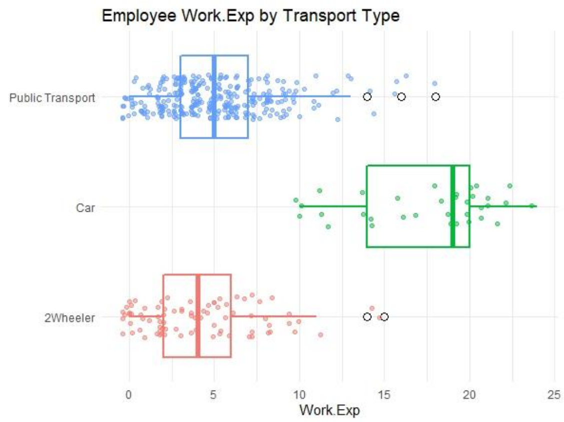

Employee Age by Transport
density
Age
Transport
2Wheeler
Car
Public Transport

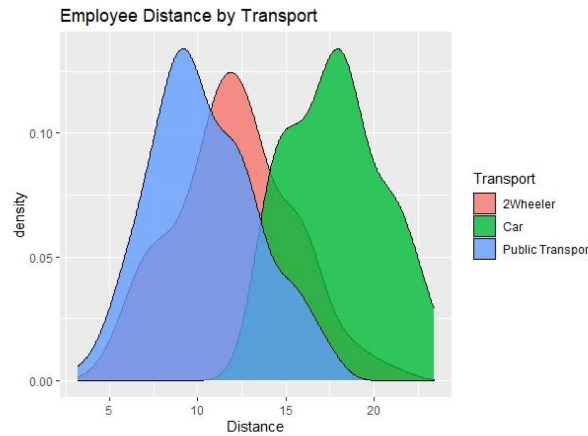

Employee Age by Transport Type
Public Transport
Car
2Wheeler
Age

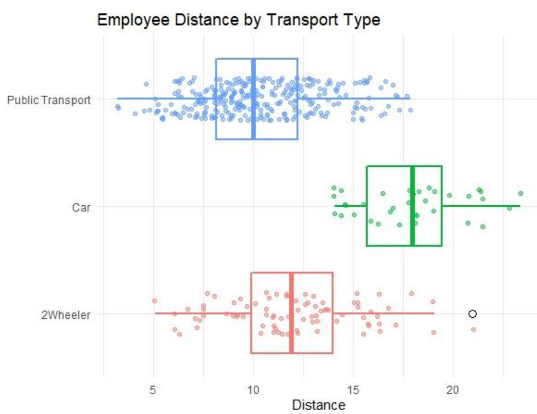

Employee Salary by Transport
density
Salary
Transport
2Wheeler
Car
Public Transport

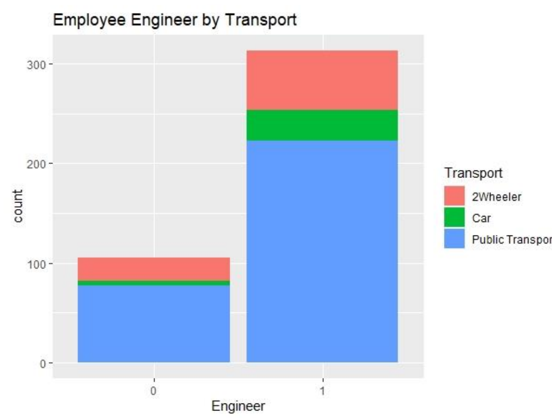

Employee Salary by Transport Type
Public Transport
Car
2Wheeler
Salary (k)


Employee Work.Exp by Transport
density
Work.Exp
Transport
2Wheeler
Car
Public Transport










```
> #Display the internal structure of an dataset.
> 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 : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 2 1 2 2 ...
 $ MBA      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
 $ 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  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
 $ Transport: Factor w/ 3 levels "2Wheeler","Car",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
> summary(Cars)
      Age            Gender     Engineer MBA         Work.Exp          Salary
 Min.   :18.00   Female:121   0:105    0:308    Min.   : 0.000   Min.   : 6.50
0
 1st Qu.:25.00   Male  :297   1:313    1:110    1st Qu.: 3.000   1st Qu.: 9.62
5
 Median :27.00                                  Median : 5.000   Median :13.00
0
 Mean   :27.33                                  Mean   : 5.873   Mean   :15.41
8
 3rd Qu.:29.00                                  3rd Qu.: 8.000   3rd Qu.:14.90
0
 Max.   :43.00                                  Max.   :24.000   Max.   :57.00
0
    Distance        license            Transport
 Min.   : 3.20   0:333   2Wheeler        : 83
 1st Qu.: 8.60   1: 85   Car             : 35
 Median :10.90           Public Transport:300
 Mean   :11.29
 3rd Qu.:13.57
 Max.   :23.40
```

Illustrate the insights based on EDA

- There is a lot of outlier i n the dataset
- Most of the continues data not normal distribution
- Most of the dataset are males.
- Most of them are engineer.
- Most of them don't have MBA.
- Most of them don't have License.
- Most of them use public transportation.
- Most of employee who's use car are older than others.
- Most of employee who's use car has good salary compare to others.
- Most of employee who's use car has more work.exp than others.
- Most of engineers uses public transportation.
- Most of MBA uses public transportation.
- Most of employee don't has license.

What is the most challenging aspect of this problem? What method will you use to deal with this? Comment

The most challenge is that the Variables Engineer, MBA and License came as int variable while its indicators to identify is the employee Engineer or not, and so on.

We are going to use ***as.factor()*** to convert this variable to be as factor instead of int.

Prepare the data for analysis

- We are going to use ***as.factor()*** to convert some variables to be as factor instead of int.
- Null value treatment in MBA variable.

```
- # SEPERATE DATE TO BE TOW PARS ONE FOR TRAIN AND OTHER FOR TEST
> set.seed(300)
> spl = sample.split(Cars$BiTransport, SplitRatio=0.75)
> train = subset(Cars, spl ==T)
> test = subset(Cars, spl==F)
```

```
- # we are convering dependent varible to 1 and 0 where 1 indicate Cars
and 0 indicates others
> Cars$BiTransport = ifelse(Transport == "Car",1,0)
> Cars$BiTransport = as.factor(Cars$BiTransport)
> summary(Cars)
      Age            Gender      Engineer MBA        Work.Exp          Sal
ary
 Min.   :18.00   Female:121   0:105    0:308   Min.   : 0.000   Min.
: 6.500
 1st Qu.:25.00   Male  :297   1:313    1:110   1st Qu.: 3.000   1st Qu.
: 9.625
 Median :27.00                                 Median : 5.000   Median
:13.000
 Mean   :27.33                                 Mean   : 5.873   Mean
:15.418
 3rd Qu.:29.00                                 3rd Qu.: 8.000   3rd Qu.
:14.900
 Max.   :43.00                                 Max.   :24.000   Max.
:57.000
    Distance     license           Transport    BiTransport
 Min.   : 3.20   0:333   2Wheeler        : 83   0:383
 1st Qu.: 8.60   1: 85   Car             : 35   1: 35
 Median :10.90           Public Transport:300
 Mean   :11.29
 3rd Qu.:13.57
 Max.   :23.40
```

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

```
> #KNN
> set.seed(1)
> knnmod <- caret::train(Transport ~ .,
+                         method     = "knn",
+                         tuneGrid   = expand.grid(k = 2:51),
+                         metric     = "Accuracy",
+                         preProcess = c("scale"),
+                         data       = train)
> knnmod
k-Nearest Neighbors

313 samples
  9 predictor
  3 classes: '2Wheeler', 'Car', 'Public Transport'

Pre-processing: scaled (9)
```

```
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 313, 313, 313, 313, 313, 313, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   2  0.7762964  0.4678880
   3  0.7748510  0.4465575
   4  0.7683782  0.4279847
   5  0.7764867  0.4360279
   6  0.7810937  0.4408172
   7  0.7914532  0.4538881
   8  0.7789962  0.4186331
   9  0.7806633  0.4176431
  10  0.7869559  0.4239309
  11  0.7928878  0.4321724
  12  0.7949008  0.4350748
  13  0.7964895  0.4323157
  14  0.7961462  0.4280275
  15  0.7944398  0.4169366
  16  0.7937002  0.4117211
  17  0.7962035  0.4144081
  18  0.7955360  0.4107836
  19  0.7941330  0.4008977
  20  0.7958262  0.4041582
  21  0.8007897  0.4143353
  22  0.7986755  0.4063707
  23  0.7977001  0.3991758
  24  0.7981589  0.3965351
  25  0.7999622  0.4010637
  26  0.7988916  0.3964317
  27  0.8003182  0.4003303
  28  0.7995258  0.3949627
  29  0.7988300  0.3908381
  30  0.7974384  0.3863873
  31  0.7992030  0.3905546
  32  0.7978083  0.3847628
  33  0.7974939  0.3815151
  34  0.7975353  0.3801359
  35  0.7978614  0.3823755
  36  0.7971545  0.3784883
  37  0.7965127  0.3755230
  38  0.7964904  0.3739130
  39  0.7957782  0.3710881
  40  0.7944253  0.3655013
  41  0.7919896  0.3559043
  42  0.7909797  0.3503390
  43  0.7878573  0.3369283
  44  0.7854366  0.3255156
  45  0.7861842  0.3275326
  46  0.7834071  0.3147580
  47  0.7775007  0.2888213
  48  0.7760464  0.2824309
  49  0.7749871  0.2778716
  50  0.7745671  0.2760775
  51  0.7731758  0.2696837

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 21.
>
> na.omit(train)
    Age Gender Engineer MBA Work.Exp Salary Distance license        Transport
BiTransport
2    24   Male        1   0        6   10.6      6.1       0         2Wheeler
0
```

```
4    25   Male      0  0     1   7.6    6.3    0       2Wheeler
0
5    25 Female      0  0     3   9.6    6.7    0       2Wheeler
0
6    21   Male      0  0     3   9.5    7.1    0       2Wheeler
0
7    23   Male      1  1     3  11.7    7.2    0       2Wheeler
0
8    23   Male      0  0     0   6.5    7.3    0       2Wheeler
0
9    24   Male      1  0     4   8.5    7.5    0       2Wheeler
0
12   21   Male      0  1     3  10.6    7.7    0       2Wheeler
0
14   24   Male      1  0     6  12.7    8.7    0       2Wheeler
0
15   27   Male      0  1     8  15.6    9.0    0       2Wheeler
0
18   29 Female      0  0     7  14.6    9.2    0       2Wheeler
0
19   29   Male      1  0     9  23.8    9.4    0       2Wheeler
0
20   22 Female      1  1     2   8.5    9.5    0       2Wheeler
0
22   25 Female      1  0     6  11.6   10.1    0       2Wheeler
0
23   34   Male      1  1    14  36.9   10.4    1       2Wheeler
0
24   28   Male      1  0     5  14.7   10.5    1       2Wheeler
0
25   26 Female      1  0     2   9.8   10.7    0       2Wheeler
0
26   23 Female      0  0     4  11.6   10.7    0       2Wheeler
0
27   25   Male      1  1     7  13.6   10.7    0       2Wheeler
0
29   21 Female      0  0     3   9.8   11.0    0       2Wheeler
0
30   26 Female      1  0     4  12.6   11.0    0       2Wheeler
0
31   25 Female      1  0     2   8.6   11.0    0       2Wheeler
0
32   24   Male      1  0     0   8.0   11.0    1       2Wheeler
0
34   25 Female      1  1     1   8.6   11.2    0       2Wheeler
0
35   29   Male      1  0    11  22.7   11.3    1       2Wheeler
0
36   30 Female      1  0     8  14.7   11.4    1       2Wheeler
0
37   23   Male      1  0     4  10.6   11.4    0       2Wheeler
0
38   23   Male      1  0     0   6.9   11.7    0       2Wheeler
0
39   24   Male      1  0     4  12.7   11.7    0       2Wheeler
0
40   23   Male      1  0     0   7.7   11.7    0       2Wheeler
0
41   27 Female      1  0     5  12.8   11.8    0       2Wheeler
0
42   30   Male      1  1    10  28.8   11.9    1       2Wheeler
0
43   28   Male      1  0     5  13.9   12.2    1       2Wheeler
0
```

45	26	Female	1	0	2	9.8	12.2	0	2Wheeler	0
47	28	Male	0	0	5	14.9	12.5	1	2Wheeler	0
48	24	Female	1	1	1	8.8	12.6	1	2Wheeler	0
49	24	Female	1	1	2	8.7	12.6	0	2Wheeler	0
50	25	Male	0	0	5	13.7	12.7	1	2Wheeler	0
51	34	Male	1	1	15	37.0	12.9	1	2Wheeler	0
53	18	Male	0	0	0	6.7	13.0	0	2Wheeler	0
56	26	Female	0	0	5	12.8	13.2	0	2Wheeler	0
57	22	Male	1	0	0	6.9	13.2	0	2Wheeler	0
60	26	Female	1	0	4	12.8	13.6	1	2Wheeler	0
61	23	Male	0	0	0	6.9	13.7	0	2Wheeler	0
62	24	Female	1	0	2	8.9	13.8	0	2Wheeler	0
63	24	Female	0	0	2	9.0	14.2	0	2Wheeler	0
64	27	Female	1	0	7	23.8	14.4	0	2Wheeler	0
65	24	Female	1	0	2	9.0	15.1	0	2Wheeler	0
66	22	Male	0	0	0	6.8	15.2	1	2Wheeler	0
67	25	Female	1	0	2	8.8	15.2	0	2Wheeler	0
68	24	Male	0	0	0	6.9	15.3	0	2Wheeler	0
71	26	Female	0	0	7	18.8	15.7	0	2Wheeler	0
73	23	Male	1	0	0	8.0	15.9	0	2Wheeler	0
74	20	Female	1	0	2	9.0	16.2	0	2Wheeler	0
75	22	Male	0	0	1	7.9	16.3	1	2Wheeler	0
76	26	Female	1	0	6	23.0	16.3	0	2Wheeler	0
77	26	Male	1	0	2	10.0	16.4	1	2Wheeler	0
79	24	Male	1	0	0	7.9	17.1	0	2Wheeler	0
80	23	Female	1	1	2	9.0	17.9	0	2Wheeler	0
82	26	Male	1	0	4	13.0	19.1	1	2Wheeler	0
83	28	Female	1	1	7	13.0	21.0	1	2Wheeler	0
84	38	Male	1	0	19	48.0	14.1	1	Car	1
85	38	Male	1	1	20	42.0	14.1	1	Car	1
86	40	Male	1	0	22	51.0	14.1	1	Car	1
91	34	Male	1	0	14	45.0	15.1	1	Car	1

93	37	Male	1	1	18	41.0	15.9	1	Car	1
94	39	Male	1	0	21	40.9	16.3	0	Car	1
95	32	Female	1	0	14	30.9	16.5	0	Car	1
96	40	Male	1	1	20	41.9	16.9	1	Car	1
97	38	Female	1	0	20	43.0	17.0	1	Car	1
98	33	Male	1	0	14	33.0	17.3	0	Car	1
100	31	Male	0	0	11	33.0	17.8	1	Car	1
102	39	Male	1	0	21	46.0	18.1	1	Car	1
103	38	Male	1	0	18	45.0	18.1	1	Car	1
104	40	Male	1	0	20	48.0	18.2	1	Car	1
105	30	Male	1	1	11	35.0	18.3	1	Car	1
106	39	Male	0	0	21	51.0	18.6	1	Car	1
108	42	Male	1	0	22	55.0	19.0	1	Car	1
109	33	Male	1	1	10	17.0	19.1	0	Car	1
110	40	Male	1	0	22	45.0	19.8	1	Car	1
111	37	Male	0	0	19	42.0	20.7	1	Car	1
112	43	Male	1	1	24	52.0	20.8	1	Car	1
113	34	Male	1	0	14	38.0	21.3	1	Car	1
114	40	Male	1	0	20	57.0	21.4	1	Car	1
115	38	Male	1	0	19	44.0	21.5	1	Car	1
116	37	Male	1	0	19	45.0	21.5	1	Car	1
118	39	Male	1	1	21	50.0	23.4	1	Car	1
120	23	Female	1	0	4	8.3	3.3	0	Public Transport	0
121	29	Male	1	0	7	13.4	4.1	0	Public Transport	0
122	28	Female	1	1	5	13.4	4.5	0	Public Transport	0
123	27	Male	1	0	4	13.4	4.6	0	Public Transport	0
125	26	Female	1	0	3	10.5	5.1	0	Public Transport	0
127	27	Male	1	0	4	13.5	5.2	0	Public Transport	0
129	27	Male	1	0	4	13.5	5.3	1	Public Transport	0
130	24	Male	1	0	2	8.5	5.4	0	Public Transport	0
131	27	Male	1	0	4	13.4	5.5	1	Public Transport	0
132	32	Male	1	0	9	15.5	5.5	0	Public Transport	0

```
133  25   Male        1   1       4   11.5      5.6       0 Public Transport
0
134  34   Male        1   0      13   16.5      5.9       0 Public Transport
0
135  26 Female       1   0       4   12.3      5.9       0 Public Transport
0
 [ reached 'max' / getOption("max.print") -- omitted 213 rows ]
>
> model_knn=knn(train[,c(3,4,8)],test[,c(3,4,8)],train$Transport,k=19)
>
> caret::confusionMatrix(test$Transport,model_knn,positive="Car")
Confusion Matrix and Statistics

                  Reference
Prediction         2Wheeler Car Public Transport
  2Wheeler                0   4               18
  Car                     0   4                5
  Public Transport        0   5               69

Overall Statistics

               Accuracy : 0.6952
                 95% CI : (0.5978, 0.7813)
    No Information Rate : 0.8762
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1805

 Mcnemar's Test P-Value : 6.523e-05

Statistics by Class:

                     Class: 2Wheeler Class: Car Class: Public Transport
Sensitivity                       NA    0.30769                  0.7500
Specificity                   0.7905    0.94565                  0.6154
Pos Pred Value                    NA    0.44444                  0.9324
Neg Pred Value                    NA    0.90625                  0.2581
Prevalence                    0.0000    0.12381                  0.8762
Detection Rate                0.0000    0.03810                  0.6571
Detection Prevalence          0.2095    0.08571                  0.7048
Balanced Accuracy                 NA    0.62667                  0.6827
>
> lrmod <- caret::train(BiTransport ~ Engineer+MBA+license,
+                        method     = "glm",
+                        metric     = "Sensitivity",
+                        data       = train)
Warning message:
In train.default(x, y, weights = w, ...) :
  The metric "Sensitivity" was not in the result set. Accuracy will be used i
nstead.
>
> lrpred<-predict(lrmod,newdata=test)
> lrpred
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
 [46] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
 [91] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Levels: 0 1
>
> Name = c("Naive_Bayes", "KNN", "Logistic_Regression")
> Accuracy = c(0.93,0.94,0.92)
> Sensitivity=c(0.98,1,0.87)
> Specificity=c(0.90,0.91,0.97)
```

```
> output = data.frame(Name,Accuracy,Sensitivity,Specificity)
> output
                 Name Accuracy Sensitivity Specificity
1         Naive_Bayes     0.93        0.98        0.90
2                 KNN     0.94        1.00        0.91
3 Logistic_Regression     0.92        0.87        0.97
```

```
> #naiveBayes
> model<-naiveBayes(BiTransport~.,data=train)
> model

Naive Bayes Classifier for Discrete Predictors

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

A-priori probabilities:
Y
         0          1 
0.91693291 0.08306709 

Conditional probabilities:
   Age
Y       [,1]     [,2]
  0 26.42857 3.037232
  1 37.26923 3.377072

   Gender
Y       Female       Male
  0 0.31010453 0.68989547
  1 0.07692308 0.92307692

   Engineer
Y           0         1
  0 0.2543554 0.7456446
  1 0.1153846 0.8846154

   MBA
Y           0         1
  0 0.7421603 0.2578397
  1 0.7307692 0.2692308

   Work.Exp
Y         [,1]     [,2]
  0  4.682927 3.151283
  1 18.230769 3.839872

   Salary
Y       [,1]     [,2]
  0 12.73937 4.700009
  1 43.06538 8.478016

   Distance
Y       [,1]     [,2]
  0 10.76620 3.181471
  1 18.26538 2.543217

   license
Y           0         1
  0 0.8606272 0.1393728
  1 0.1538462 0.8461538

   Transport
```

```
Y    2Wheeler       Car Public Transport
  0 0.2125436 0.0000000        0.7874564
  1 0.0000000 1.0000000        0.0000000

>
> # generating the probabilities in prediction
> ypred<-predict(model, newdata = test, type="raw")
> plot(test$BiTransport,ypred[,2])
>
> # generating the class in prediction
> pred<-predict(model,newdata=test)
>
>
>
> p_test<-prediction(ypred[,2], test$BiTransport)
> perf<-performance(p_test,"tpr", "fpr")
> plot(perf,colorize = TRUE)
>
> cutoffs <-
+     data.frame(
+         cut = perf@alpha.values[[1]],
+         fpr = perf@x.values[[1]],
+         tpr = perf@y.values[[1]]
+     )
>
> head(cutoffs)
        cut        fpr       tpr
1       Inf 0.00000000 0.0000000
2 1.0000000 0.00000000 0.4444444
3 1.0000000 0.00000000 0.5555556
4 1.0000000 0.00000000 0.6666667
5 0.9999999 0.00000000 0.7777778
6 0.9971898 0.01041667 0.7777778
>
> cutoffs <- cutoffs[order(cutoffs$tpr, decreasing=TRUE),]
> head(subset(cutoffs, fpr < 0.1))
            cut        fpr tpr
10 3.289848e-03 0.03125000   1
11 7.123813e-06 0.04166667   1
12 7.023308e-06 0.05208333   1
13 1.599182e-07 0.06250000   1
14 1.522523e-07 0.07291667   1
15 1.394685e-07 0.08333333   1
>
> class_prediction_with_new_cutoff = ifelse(ypred[, 2] >= 0.0129, 1, 0)
> new_confusion_matrix = table(test$BiTransport, class_prediction_with_new_cu
toff)
>
>
> new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
> new_accuracy
[1] 0.9619048
>
> new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ]
)
> new_sensitivity
[1] 0.8888889
>
> new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ]
)
> new_specificity
[1] 0.96875
>
> AUC_NB=performance(p_test,"auc")@y.values
```

```
> AUC_NB
[[1]]
[1] 0.9930556

>
> ks_nb = max(attr(perf,'y.values')[[1]] - attr(perf,'x.values')[[1]])
> ks_nb
[1] 0.96875
>
> GINI_NB=2*AUC_NB[[1]]-1
> GINI_NB
[1] 0.9861111
```

```
> # Logistic Regression
> ## Check split consistency
> prop.table(table(train$BiTransport))

         0          1
0.91693291 0.08306709
> prop.table(table(test$BiTransport))

         0          1
0.91428571 0.08571429
> prop.table(table(Cars$BiTransport))

         0          1
0.91626794 0.08373206
> LRmodel = glm(BiTransport~ ., data = train, family= binomial)
Warning message:
glm.fit: algorithm did not converge
> summary(LRmodel)

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

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-2.409e-06  -2.409e-06  -2.409e-06  -2.409e-06   2.409e-06

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)               -2.657e+01  3.062e+05       0        1
Age                        1.553e-13  1.346e+04       0        1
GenderMale                -3.291e-13  4.676e+04       0        1
Engineer1                  3.654e-13  4.785e+04       0        1
MBA1                       3.055e-12  4.698e+04       0        1
Work.Exp                   2.019e-12  1.628e+04       0        1
Salary                    -2.009e-12  7.646e+03       0        1
Distance                   1.227e-13  6.720e+03       0        1
license1                  -3.078e-12  6.321e+04       0        1
TransportCar               5.313e+01  1.523e+05       0        1
TransportPublic Transport -2.132e-12  5.783e+04       0        1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1.7916e+02  on 312  degrees of freedom
Residual deviance: 1.8159e-09  on 302  degrees of freedom
AIC: 22

Number of Fisher Scoring iterations: 25

>
```

```
> log_model = stepAIC(LRmodel, direction = "both",k=5) #loosely speaking K=5, represents (P < 0.02)
Start:  AIC=55
BiTransport ~ Age + Gender + Engineer + MBA + Work.Exp + Salary + 
    Distance + license + Transport

            Df   Deviance AIC
- Transport  2 3.4191e-08  45
- Age        1 1.8160e-09  50
- Gender     1 1.8160e-09  50
- Engineer   1 1.8160e-09  50
- MBA        1 1.8160e-09  50
- Work.Exp   1 1.8160e-09  50
- Salary     1 1.8160e-09  50
- Distance   1 1.8160e-09  50
- license    1 1.8160e-09  50
<none>         1.8160e-09  55

Step:  AIC=45
BiTransport ~ Age + Gender + Engineer + MBA + Work.Exp + Salary + 
    Distance + license

            Df Deviance    AIC
- Age        1    0.000 40.000
- Work.Exp   1    0.000 40.000
- MBA        1    0.000 40.000
- Engineer   1    0.000 40.000
- Gender     1    0.000 40.000
- license    1    0.000 40.000
- Salary     1    0.000 40.000
<none>            0.000 45.000
+ Transport  2    0.000 55.000
- Distance   1   26.264 66.264

Step:  AIC=40
BiTransport ~ Gender + Engineer + MBA + Work.Exp + Salary + Distance + 
    license

            Df Deviance    AIC
- Work.Exp   1    0.000 35.000
- MBA        1    0.000 35.000
- Engineer   1    0.000 35.000
- Gender     1    0.000 35.000
- license    1    0.000 35.000
- Salary     1    0.000 35.000
<none>            0.000 40.000
+ Age        1    0.000 45.000
+ Transport  2    0.000 50.000
- Distance   1   26.899 61.899

Step:  AIC=35
BiTransport ~ Gender + Engineer + MBA + Salary + Distance + license

            Df Deviance    AIC
- MBA        1    0.000 30.000
- Engineer   1    0.000 30.000
- Gender     1    0.000 30.000
- license    1    0.000 30.000
<none>            0.000 35.000
+ Work.Exp   1    0.000 40.000
+ Age        1    0.000 40.000
+ Transport  2    0.000 45.000
- Distance   1   27.063 57.063
- Salary     1   53.548 83.548
```

```
Step:  AIC=30
BiTransport ~ Gender + Engineer + Salary + Distance + license

            Df Deviance    AIC
- Engineer   1    0.000 25.000
- Gender     1    0.000 25.000
- license    1    0.000 25.000
<none>            0.000 30.000
+ MBA        1    0.000 35.000
+ Age        1    0.000 35.000
+ Work.Exp   1    0.000 35.000
+ Transport  2    0.000 40.000
- Distance   1   28.723 53.723
- Salary     1   54.166 79.166

Step:  AIC=25
BiTransport ~ Gender + Salary + Distance + license

            Df Deviance    AIC
- Gender     1    0.000 20.000
- license    1    0.000 20.000
<none>            0.000 25.000
+ Engineer   1    0.000 30.000
+ Age        1    0.000 30.000
+ Work.Exp   1    0.000 30.000
+ MBA        1    0.000 30.000
+ Transport  2    0.000 35.000
- Distance   1   28.802 48.802
- Salary     1   58.307 78.307

Step:  AIC=20
BiTransport ~ Salary + Distance + license

            Df Deviance    AIC
<none>            0.000 20.000
- license    1    5.567 20.567
+ Gender     1    0.000 25.000
+ MBA        1    0.000 25.000
+ Age        1    0.000 25.000
+ Engineer   1    0.000 25.000
+ Work.Exp   1    0.000 25.000
+ Transport  2    0.000 30.000
- Distance   1   28.879 43.879
- Salary     1   59.536 74.536
There were 50 or more warnings (use warnings() to see the first 50)
> summary(log_model)

Call:
glm(formula = BiTransport ~ Salary + Distance + license, family = binomial, 
    data = train)

Deviance Residuals: 
       Min          1Q      Median          3Q         Max  
-2.512e-04  -2.100e-08  -2.100e-08  -2.100e-08   2.143e-04  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -891.22  107362.28  -0.008    0.993
Salary          10.37    1247.97   0.008    0.993
Distance        38.35    4658.31   0.008    0.993
license1       -67.05    9508.29  -0.007    0.994

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 1.7916e+02  on 312  degrees of freedom
Residual deviance: 1.6590e-07  on 309  degrees of freedom
AIC: 8

Number of Fisher Scoring iterations: 25

> varImp(log_model)
             Overall
Salary   0.008310284
Distance 0.008232789
license1 0.007052167
> # convert to data frame
> l = data.frame(varImp(log_model))
> l <- cbind(newColName = rownames(l), l)
> rownames(l) <- 1:nrow(l)
>
> # soritng the imprtance of varaible
> l[with(l, order(-Overall)), ]
  newColName     Overall
1     Salary 0.008310284
2   Distance 0.008232789
3   license1 0.007052167
>
> exp(0.0296)
[1] 1.030042
> P = 0.49
>
> # prediction on test dataset
> predTrain = predict(log_model, newdata= train, type="response")
> tb = table(predTrain>0.50,train$BiTransport)
> tb

          0   1
  FALSE 287   0
  TRUE    0  26
> print('accuracy is ')
[1] "accuracy is "
> sum(diag(tb))/sum(tb)
[1] 1
>
> # prediction on test dataset
> predTest = predict(log_model, newdata= test, type="response")
> tb = table(predTest >0.50,test$BiTransport)
> tb

         0  1
  FALSE 96  3
  TRUE   0  6
> print('accuracy is ')
[1] "accuracy is "
> sum(diag(tb))/sum(tb)
[1] 0.9714286
>
> #par(mfrow=c(1,2))
> p0 <- prediction(predTrain,train$BiTransport)
> p1 <- performance(p0, "tpr", "fpr")
> plot(p1, main = "ROC Curve" ,colorize = TRUE) ## logistic regression model
> AUC  <- as.numeric(performance(p0, "auc")@y.values) ## AUC  = 0.9083176
> gini <- 2*AUC - 1                                   ## gini = 0.8166352
> KS   <- max(p1@y.values[[1]] - p1@x.values[[1]])    ## KS   = 0.6511416
> print('AUC')
[1] "AUC"
> AUC
```

```
[1] 1
> print('KS')
[1] "KS"
> KS
[1] 1
> p0 = prediction(predTrain,train$BiTransport)
> p1 = performance(p0,"tpr","fpr")
> plot(p1, main = "ROC Curve" ,colorize = TRUE)
> str(p1)
Formal class 'performance' [package "ROCR"] with 6 slots
  ..@ x.name      : chr "False positive rate"
  ..@ y.name      : chr "True positive rate"
  ..@ alpha.name  : chr "Cutoff"
  ..@ x.values    :List of 1
  .. ..$ : num [1:8] 0 0 0 0 0.00348 ...
  ..@ y.values    :List of 1
  .. ..$ : num [1:8] 0 0.923 0.962 1 1 ...
  ..@ alpha.values:List of 1
  .. ..$ : num [1:8] Inf 1.00 1.00 1.00 3.15e-08 ...
>
> cutoffs <-
+     data.frame(
+         cut = p1@alpha.values[[1]],
+         fpr = p1@x.values[[1]],
+         tpr = p1@y.values[[1]]
+     )
>
> head(cutoffs)
           cut         fpr       tpr
1          Inf 0.000000000 0.0000000
2 1.000000e+00 0.000000000 0.9230769
3 1.000000e+00 0.000000000 0.9615385
4 1.000000e+00 0.000000000 1.0000000
5 3.154956e-08 0.003484321 1.0000000
6 1.402683e-08 0.006968641 1.0000000
> View(cutoffs)
> cutoffs <- cutoffs[order(cutoffs$tpr, decreasing=TRUE),]
> head(subset(cutoffs, fpr < 0.2))
           cut         fpr       tpr
4 1.000000e+00 0.000000000 1.0000000
5 3.154956e-08 0.003484321 1.0000000
6 1.402683e-08 0.006968641 1.0000000
7 1.065691e-12 0.010452962 1.0000000
3 1.000000e+00 0.000000000 0.9615385
2 1.000000e+00 0.000000000 0.9230769
> class_prediction_with_new_cutoff = ifelse(predTrain>= 0.24, 1, 0)
> new_confusion_matrix = table(train$BiTransport,class_prediction_with_new_cu
toff )
> new_confusion_matrix
   class_prediction_with_new_cutoff
      0   1
  0 287   0
  1   0  26
> new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
> new_accuracy
[1] 1
> new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ]
)
> new_sensitivity
[1] 1
> new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ]
)
> new_specificity
[1] 1
```

```
> class_prediction_with_new_cutoff = ifelse(predTest>= 0.24, 1, 0)
> new_confusion_matrix = table(test$BiTransport ,class_prediction_with_new_cu
toff)
> new_confusion_matrix
   class_prediction_with_new_cutoff
     0  1
  0 96  0
  1  3  6
> new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
> new_accuracy
[1] 0.9714286
> new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ]
)
> new_sensitivity
[1] 0.6666667
> new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ]
)
> new_specificity
[1] 1
```

Apply both bagging and boosting modeling procedures to create 2 models and compare its accuracy with the best model of the above step

```
>
> #logistic regression
> german_logistic <- glm(Transport~., data=train, family=binomial(link="logit
"))
> test$log.pred<-predict(german_logistic, test, type="response")
> table(test$Transport,test$log.pred>0.5)

                    FALSE TRUE
  2Wheeler              5   17
  Car                   0    9
  Public Transport      8   66
>
> #knn
> #knn compare
> knn_fit<- knn(train = train[,c(3,4,8)], test = test[,c(3,4,8)], cl= train[,
8],k = 3,prob=TRUE)
> table(test[,9],knn_fit)
                  knn_fit
                    0  1
  2Wheeler         16  6
  Car               2  7
  Public Transport 64 10
>
> #naive bayes
> nb_gd<-naiveBayes(x=train[,c(3,4,8)], y=as.factor(train[,9]))
> pred_nb<-predict(nb_gd,newdata = test[,c(3,4,8)])
> table(test[,9],pred_nb)
                  pred_nb
                   2Wheeler Car Public Transport
  2Wheeler                0   5               17
  Car                     1   6                2
  Public Transport        2   8               64
>
> ## Bagging
> Cars.bagging <- bagging(Transport ~.,
```

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

                   2Wheeler Car Public Transport
  2Wheeler                4   0               18
  Car                     0   9                0
  Public Transport        1   0               73
>
> #Boosting
> gbm.fit <- gbm(
+     formula = Transport ~ .,
+     data = train,
+     n.trees = 10000, #these are the number of stumps
+     interaction.depth = 1,#number of splits it has to perform on a tree (st
arting from a single node)
+     shrinkage = 0.001,#shrinkage is used for reducing, or shrinking the imp
act of each additional fitted base-learner(tree)
+     cv.folds = 5,#cross validation folds
+     n.cores = NULL, # will use all cores by default
+     verbose = FALSE#after every tree/stump it is going to show the error an
d how it is changing
+ )
Distribution not specified, assuming multinomial ...
Warning message:
Setting `distribution = "multinomial"` is ill-advised as it is currently brok
en. It exists only for backwards compatibility. Use at your own risk.
> test$pred.Transport <- predict(gbm.fit, test,type="response" )
Using 5324 trees...

> #we have to put type="response" just like in logistic regression else we wi
ll have log odds
> table(test$Transport,head(test$pred.Transport,105))

                   0.00176928594465705 0.00181555556933653 0.0021050252753647
8
  2Wheeler                           0                   0
0
  Car                                1                   1
1
  Public Transport                   0                   0
0

                   0.00213662488994355 0.00236008596309604 0.0028104069341935
5
  2Wheeler                           0                   0
0
  Car                                1                   1
1
  Public Transport                   0                   0
0

                   0.00340461791469289 0.00544113719131183 0.0063307883552711
5
  2Wheeler                           0                   0
0
  Car                                1                   1
1
  Public Transport                   0                   0
0
```

```
                    0.0228962276781549 0.0299655195057322 0.0305624718013885 0
.0333921824285581
  2Wheeler                           0                  0                  1
0
  Car                                0                  0                  0
0
  Public Transport                   1                  1                  0
1

                    0.0337318839628448 0.0344116116405112 0.0356777862098044 0
.0361203874762107
  2Wheeler                           0                  0                  0
0
  Car                                0                  0                  0
0
  Public Transport                   1                  1                  1
1

                    0.0374378867605702 0.0387246159933002 0.04438694154863 0.0
475072252229793
  2Wheeler                           0                  0                0
0
  Car                                0                  0                0
0
  Public Transport                   1                  1                1
1

                    0.0554345358776495 0.0555357176835204 0.0627128328899716 0
.0645434647517116
  2Wheeler                           0                  0                  1
0
  Car                                0                  0                  0
0
  Public Transport                   1                  1                  0
1

                    0.066265701796915 0.0733694189946565 0.0744999120471867 0.
0774021874838983
  2Wheeler                          0                  0                  0
1
  Car                               0                  0                  0
0
  Public Transport                  1                  1                  1
0

                    0.0790652120028973 0.0791022090347707 0.0815224886231135 0
.082291342515923
  2Wheeler                           0                  0                  0
0
  Car                                0                  0                  0
0
  Public Transport                   1                  1                  1
1

                    0.0829659902567295 0.0840917417391661 0.0856158314922602 0
.0856294920128647
  2Wheeler                           0                  0                  0
0
  Car                                0                  0                  0
0
  Public Transport                   1                  1                  1
1
```

```
                  0.0866596450907235 0.0881854270641302 0.0883357857232036 0
.0883541051707774
  2Wheeler                         0                  0                  1
0
  Car                              0                  0                  0
0
  Public Transport                 1                  1                  0
1

                  0.0896325928445594 0.092082429232548 0.0921142160385165 0.
0966805338908322
  2Wheeler                         0                 0                  0
0
  Car                              0                 0                  0
0
  Public Transport                 1                 1                  1
1

                  0.0968988664397376 0.0985404558731847 0.100350254955396 0.
103233026493284
  2Wheeler                         0                  0                 0
0
  Car                              0                  0                 0
0
  Public Transport                 1                  1                 1
1

                  0.10592148714438 0.108671531372673 0.1118496735716 0.11264
0688713877
  2Wheeler                       1                 1               1
0
  Car                            0                 0               0
0
  Public Transport               0                 0               0
1

                  0.113208660513838 0.117788656747037 0.120368520305303 0.12
6393556116369
  2Wheeler                        0                 0                 0
0
  Car                             0                 0                 0
0
  Public Transport                1                 1                 1
1

                  0.128011441811165 0.12896436017338 0.135767320305236 0.137
021865294057
  2Wheeler                        0                0                 0
0
  Car                             0                0                 0
0
  Public Transport                1                1                 1
1

                  0.147063086394311 0.148767874357105 0.149184245028799 0.15
4154447646541
  2Wheeler                        0                 0                 0
0
  Car                             0                 0                 0
0
  Public Transport                1                 1                 1
1
```

```
                  0.158160585108155 0.15974603975533 0.160562128225509 0.181
276691867355
  2Wheeler                        0                0                 0
1
  Car                             0                0                 0
0
  Public Transport                1                1                 1
0

                  0.202879329016927 0.210490872639391 0.213532004071484 0.21
8400679933066
  2Wheeler                        0                 0                 0
0
  Car                             0                 0                 0
0
  Public Transport                1                 1                 1
1

                  0.23094463923439 0.237078989161193 0.249668423298066 0.258
384529747216
  2Wheeler                       2                 0                 0
0
  Car                            0                 0                 0
0
  Public Transport               0                 1                 1
1

                  0.261298283275964 0.283239983453924 0.286054437095495 0.30
1998075809966
  2Wheeler                        1                 0                 1
1
  Car                             0                 0                 0
0
  Public Transport                0                 1                 0
0

                  0.313636811594753 0.315234546958144 0.327490847573369 0.33
5084162420767
  2Wheeler                        0                 0                 0
0
  Car                             0                 0                 0
0
  Public Transport                1                 1                 1
1

                  0.341523205879136 0.363131028203453 0.399766259226481 0.41
2454074809064
  2Wheeler                        0                 1                 1
0
  Car                             0                 0                 0
0
  Public Transport                1                 0                 0
1

                  0.431965937050087 0.434576174821102 0.441408932112767 0.45
7540781067235
  2Wheeler                        1                 0                 0
0
  Car                             0                 0                 0
0
  Public Transport                0                 1                 1
1
```

```
                  0.457558020284486 0.476916516318756 0.481478290926092 0.49
7685709229536
  2Wheeler                        1                 0                 0
1
  Car                             0                 0                 0
0
  Public Transport                0                 1                 1
0

                  0.511119964957866 0.515482043834696 0.516843948119909 0.54
6126041966845
  2Wheeler                        1                 1                 1
1
  Car                             0                 0                 0
0
  Public Transport                0                 0                 0
0

                  0.564018561701784 0.589944160123038 0.696807137792704
  2Wheeler                        0                 0                 0
  Car                             0                 0                 0
  Public Transport                1                 1                 1
```

Appendix A – Source Code

```r
#==============================================================================
#
# Project 4
#
#==============================================================================
#calling all libraries that we are going to use
library(readr)
library(ggplot2)

#setting up working directory
setwd("C:/Users/ahmasiri/Desktop/PGP DSBA/Data/Project 4 - Cars Case Study")
#reading data from csv file to Cars variable and view it
Cars <- read.csv("Cars-dataset.csv")
attach(Cars)



#Prepare the data for analysis
Cars$Engineer = as.factor(Engineer)
Cars$MBA = as.factor(MBA)
Cars$license = as.factor(license)

#dealing with NA values in MBA variable
Cars[is.na(Cars)] <- 1


#check if ther is any NA value in dataset
anyNA(Cars)


# EDA

#Retrieve the dimension of an object.
dim(Cars)

#Get the names of an object.
names(Cars)

#Display the internal structure of an dataset.
str(Cars)

#Returns the first 10 rows of the dataset.
head(Cars, 10)

#Returns the last 10 rows of the dataset.
tail(Cars, 10)

#Return a summary of the dataset variables.
summary(Cars)
```

```
#graph for all variable variables
# Quantitative
par(mfrow=c(2,2))
boxplot(Age, main = "Age")
boxplot(Work.Exp, main = "Work.Exp")
hist(Age, main = "Age")
hist(Work.Exp, main = "Work.Exp")

par(mfrow=c(2,2))
boxplot(Salary,main = "Salary(K)")
boxplot(Distance, main = "Distance")
hist(Salary, main = "Salary(K)")
hist(Distance, main = "Distance")

# catagorical
par(mfrow=c(2,2))
ggplot(Cars) + geom_bar(aes(x = Gender))
ggplot(Cars) + geom_bar(aes(x = Engineer))
ggplot(Cars) + geom_bar(aes(x = MBA))
ggplot(Cars) + geom_bar(aes(x = license))
ggplot(Cars) + geom_bar(aes(x = Transport))


#Bi-Variate Analysis
#kernel density plots
ggplot(Cars,
       aes(x = Age, #quantitative variable
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  geom_density(alpha = 0.8) + #setting transparency of graph to keep overlaps
visible
  labs(fill = "Transport", # setting title of legend
       x = "Age",
       title = "Employee Age by Transport")
#jitter and box plots
ggplot(Cars,
       aes(x = factor(Transport, #defining x axis a categorical
                      labels = c("2Wheeler", "Car", "Public Transport")),
           y = Age,
           color = Transport)) + #specifying that coloring is to be based on
drive type
  geom_boxplot(size=1, #makes the lines thicker
               outlier.shape = 1, #specifies circles for outliers
               outlier.color = "black", #makes outliers black
               outlier.size = 3) + #increases the size of the outlier symbol
  geom_jitter(alpha = 0.5, #setting transparency of graph
              width=.2) + #decreases the amount of jitter (.4 is the default)
  labs(title = "Employee Age by Transport Type",
       x = "",
       y = "Age") +
  theme_minimal() + #setting minimal theme (no background color)
  theme(legend.position = "none") + #hiding legend
  coord_flip() #x and y axes are reversed
```

```
#kernel density plots
ggplot(Cars,
       aes(x = Salary, #quantitative variable
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  geom_density(alpha = .8) + #setting transparency of graph to keep overlaps
visible
  labs(fill = "Transport", # setting title of legend
       x = "Salary",
       title = "Employee Salary by Transport")

#jitter and box plots
ggplot(Cars,
       aes(x = factor(Transport, #defining x axis a categorical
                      labels = c("2Wheeler", "Car", "Public Transport")),
           y = Salary,
           color = Transport)) + #specifying that coloring is to be based on
drive type
  geom_boxplot(size=1, #makes the lines thicker
               outlier.shape = 1, #specifies circles for outliers
               outlier.color = "black", #makes outliers black
               outlier.size = 3) + #increases the size of the outlier symbol
  geom_jitter(alpha = 0.5, #setting transparency of graph
              width=.2) + #decreases the amount of jitter (.4 is the default)
  labs(title = "Employee Salary by Transport Type",
       x = "",
       y = "Salary (k)") +
  theme_minimal() + #setting minimal theme (no background color)
  theme(legend.position = "none") + #hiding legend
  coord_flip() #x and y axes are reversed


#kernel density plots
ggplot(Cars,
       aes(x = Work.Exp, #quantitative variable
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  geom_density(alpha = .8) + #setting transparency of graph to keep overlaps
visible
  labs(fill = "Transport", # setting title of legend
       x = "Work.Exp",
       title = "Employee Work.Exp by Transport")
#jitter and box plots
ggplot(Cars,
       aes(x = factor(Transport, #defining x axis a categorical
                      labels = c("2Wheeler", "Car", "Public Transport")),
           y = Work.Exp,
           color = Transport)) + #specifying that coloring is to be based on
drive type
  geom_boxplot(size=1, #makes the lines thicker
               outlier.shape = 1, #specifies circles for outliers
```

```
             outlier.color = "black", #makes outliers black
             outlier.size = 3) + #increases the size of the outlier symbol
  geom_jitter(alpha = 0.5, #setting transparency of graph
            width=.2) + #decreases the amount of jitter (.4 is the default)
  labs(title = "Employee Work.Exp by Transport Type",
       x = "",
       y = "Work.Exp") +
  theme_minimal() + #setting minimal theme (no background color)
  theme(legend.position = "none") + #hiding legend
  coord_flip() #x and y axes are reversed


#kernel density plots
ggplot(Cars,
       aes(x = Distance, #quantitative variable
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  geom_density(alpha = .8) + #setting transparency of graph to keep overlaps
visible
  labs(fill = "Transport", # setting title of legend
       x = "Distance",
       title = "Employee Distance by Transport")
#jitter and box plots
ggplot(Cars,
       aes(x = factor(Transport, #defining x axis a categorical
                      labels = c("2Wheeler", "Car", "Public Transport")),
           y = Distance,
           color = Transport)) + #specifying that coloring is to be based on
drive type
  geom_boxplot(size=1, #makes the lines thicker
             outlier.shape = 1, #specifies circles for outliers
             outlier.color = "black", #makes outliers black
             outlier.size = 3) + #increases the size of the outlier symbol
  geom_jitter(alpha = 0.5, #setting transparency of graph
            width=.2) + #decreases the amount of jitter (.4 is the default)
  labs(title = "Employee Distance by Transport Type",
       x = "",
       y = "Distance") +
  theme_minimal() + #setting minimal theme (no background color)
  theme(legend.position = "none") + #hiding legend
  coord_flip() #x and y axes are reversed



# stacked bar chart
ggplot(Cars,
       aes(x = Engineer,
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  labs(fill = "Transport", # setting title of legend
       x = "Engineer",
```

```r
       title = "Employee Engineer by Transport") +
  geom_bar(position = "stack") #specifying the type of bar chart as stacked

# stacked bar chart
ggplot(Cars,
       aes(x = MBA,
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  labs(fill = "Transport", # setting title of legend
       x = "MBA",
       title = "Employee MBA by Transport") +
  geom_bar(position = "stack") #specifying the type of bar chart as stacked


# stacked bar chart
ggplot(Cars,
       aes(x = license,
           fill = factor(Transport, #defining x axis a categorical
                         levels = c("2Wheeler", "Car", "Public Transport"),
                         labels = c("2Wheeler", "Car", "Public Transport"))))
+
  labs(fill = "Transport", # setting title of legend
       x = "license",
       title = "Employee license by Transport") +
  geom_bar(position = "stack") #specifying the type of bar chart as stacked





# we are convering dependent varible to 1 and 0 where 1 indicate Cars and 0
indicates others
Cars$BiTransport = ifelse(Transport == "Car",1,0)
Cars$BiTransport = as.factor(Cars$BiTransport)
summary(Cars)

library(caTools)

# SEPERATE DATE TO BE TOW PARS ONE FOR TRAIN AND OTHER FOR TEST
set.seed(300)
spl = sample.split(Cars$BiTransport, SplitRatio=0.75)
train = subset(Cars, spl ==T)
test = subset(Cars, spl==F)



library(class)

#KNN
set.seed(1)
knnmod <- caret::train(Transport ~ .,
                       method     = "knn",
                       tuneGrid   = expand.grid(k = 2:51),
```

```r
                              metric      = "Accuracy",
                              preProcess = c("scale"),
                              data        = train)
knnmod

na.omit(train)

model_knn=knn(train[,c(3,4,8)],test[,c(3,4,8)],train$Transport,k=19)

caret::confusionMatrix(test$Transport,model_knn,positive="Car")

lrmod <- caret::train(BiTransport ~ Engineer+MBA+license,
                             method      = "glm",
                             metric      = "Sensitivity",
                             data        = train)

lrpred<-predict(lrmod,newdata=test)
lrpred

Name = c("Naive_Bayes", "KNN", "Logistic_Regression")
Accuracy = c(0.93,0.94,0.92)
Sensitivity=c(0.98,1,0.87)
Specificity=c(0.90,0.91,0.97)
output = data.frame(Name,Accuracy,Sensitivity,Specificity)
output




library(e1071) # to build a naive bayes model
library(ROCR)

#naiveBayes
model<-naiveBayes(BiTransport~.,data=train)
model

# generating the probabilities in prediction
ypred<-predict(model, newdata = test, type="raw")
plot(test$BiTransport,ypred[,2])

# generating the class in prediction
pred<-predict(model,newdata=test)



p_test<-prediction(ypred[,2], test$BiTransport)
perf<-performance(p_test,"tpr", "fpr")
plot(perf,colorize = TRUE)

cutoffs <-
  data.frame(
    cut = perf@alpha.values[[1]],
    fpr = perf@x.values[[1]],
    tpr = perf@y.values[[1]]
  )
```

```
head(cutoffs)

cutoffs <- cutoffs[order(cutoffs$tpr, decreasing=TRUE),]
head(subset(cutoffs, fpr < 0.1))

class_prediction_with_new_cutoff = ifelse(ypred[, 2] >= 0.0129, 1, 0)
new_confusion_matrix = table(test$BiTransport,
class_prediction_with_new_cutoff)


new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
new_accuracy

new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ])
new_sensitivity

new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ])
new_specificity

AUC_NB=performance(p_test,"auc")@y.values
AUC_NB

ks_nb = max(attr(perf,'y.values')[[1]] - attr(perf,'x.values')[[1]])
ks_nb

GINI_NB=2*AUC_NB[[1]]-1
GINI_NB




library(MASS)
library(caret)

# Logistic Regression
## Check split consistency
prop.table(table(train$BiTransport))
prop.table(table(test$BiTransport))
prop.table(table(Cars$BiTransport))
LRmodel = glm(BiTransport~ ., data = train, family= binomial)
summary(LRmodel)

log_model = stepAIC(LRmodel, direction = "both",k=5) #loosely speaking K=5,
represents (P < 0.02)
summary(log_model)
varImp(log_model)
# convert to data frame
l = data.frame(varImp(log_model))
l <- cbind(newColName = rownames(l), l)
rownames(l) <- 1:nrow(l)

# soritng the imprtance of varaible
l[with(l, order(-Overall)), ]

exp(0.0296)
P = 0.49
```

```
# prediction on test dataset
predTrain = predict(log_model, newdata= train, type="response")
tb = table(predTrain>0.50,train$BiTransport)
tb
print('accuracy is ')
sum(diag(tb))/sum(tb)

# prediction on test dataset
predTest = predict(log_model, newdata= test, type="response")
tb = table(predTest >0.50,test$BiTransport)
tb
print('accuracy is ')
sum(diag(tb))/sum(tb)

#par(mfrow=c(1,2))
p0 <- prediction(predTrain,train$BiTransport)
p1 <- performance(p0, "tpr", "fpr")
plot(p1, main = "ROC Curve" ,colorize = TRUE) ## logistic regression model
AUC  <- as.numeric(performance(p0, "auc")@y.values) ## AUC  = 0.9083176
gini <- 2*AUC - 1                                    ## gini = 0.8166352
KS   <- max(p1@y.values[[1]] - p1@x.values[[1]])     ## KS   = 0.6511416
print('AUC')
AUC
print('KS')
KS
p0 = prediction(predTrain,train$BiTransport)
p1 = performance(p0,"tpr","fpr")
plot(p1, main = "ROC Curve" ,colorize = TRUE)
str(p1)

cutoffs <-
  data.frame(
    cut = p1@alpha.values[[1]],
    fpr = p1@x.values[[1]],
    tpr = p1@y.values[[1]]
  )

head(cutoffs)
View(cutoffs)
cutoffs <- cutoffs[order(cutoffs$tpr, decreasing=TRUE),]
head(subset(cutoffs, fpr < 0.2))
class_prediction_with_new_cutoff = ifelse(predTrain>= 0.24, 1, 0)
new_confusion_matrix =
table(train$BiTransport,class_prediction_with_new_cutoff )
new_confusion_matrix
new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
new_accuracy
new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ])
new_sensitivity
new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ])
new_specificity
class_prediction_with_new_cutoff = ifelse(predTest>= 0.24, 1, 0)
new_confusion_matrix = table(test$BiTransport
,class_prediction_with_new_cutoff)
new_confusion_matrix
```

```
new_accuracy = sum(diag(new_confusion_matrix)) / sum(new_confusion_matrix)
new_accuracy
new_sensitivity = new_confusion_matrix[2,2] / sum(new_confusion_matrix[2, ])
new_sensitivity
new_specificity = new_confusion_matrix[1,1] / sum(new_confusion_matrix[1, ])
new_specificity



library(class)
library(e1071)
library(gbm)           # basic implementation using AdaBoost
library(xgboost)       # a faster implementation of a gbm#loading a few
libraries
library(caret)         # an aggregator package for performing many machine
learning models
library(ipred)
library(rpart)
library(gbm)

attach(Cars)

#logistic regression
german_logistic <- glm(Transport~., data=train,
family=binomial(link="logit"))
test$log.pred<-predict(german_logistic, test, type="response")
table(test$Transport,test$log.pred>0.5)

#knn
#knn compare
knn_fit<- knn(train = train[,c(3,4,8)], test = test[,c(3,4,8)], cl=
train[,8],k = 3,prob=TRUE)
table(test[,9],knn_fit)

#naive bayes
nb_gd<-naiveBayes(x=train[,c(3,4,8)], y=as.factor(train[,9]))
pred_nb<-predict(nb_gd,newdata = test[,c(3,4,8)])
table(test[,9],pred_nb)

## Bagging
Cars.bagging <- bagging(Transport ~.,
                        data=train,
                        control=rpart.control(maxdepth=5, minsplit=4))

test$pred.Transport <- predict(Cars.bagging, test)
table(test$Transport,test$pred.Transport)

#Boosting
gbm.fit <- gbm(
  formula = Transport ~ .,
  data = train,
  n.trees = 10000, #these are the number of stumps
  interaction.depth = 1,#number of splits it has to perform on a tree
(starting from a single node)
  shrinkage = 0.001,#shrinkage is used for reducing, or shrinking the impact
of each additional fitted base-learner(tree)
```

```
  cv.folds = 5,#cross validation folds
  n.cores = NULL, # will use all cores by default
  verbose = FALSE#after every tree/stump it is going to show the error and
how it is changing
)
test$pred.Transport <- predict(gbm.fit, test,type="response" )
#we have to put type="response" just like in logistic regression else we will
have log odds
table(test$Transport,head(test$pred.Transport,105))
```