Mini Project – Machine Learning

Table of Contents

1. Project Objective 3
2. Step by step approach 4
   2.1. Environment Set up and Data Import 4
   2.2. Perform Exploratory Data Analysis(EDA) 4
   2.3. Build Logistic Regression, KNN and Naïve Models 5
   2.4. Build Bagging, Boosting and SMOTE Models 6
   2.5. Compare the Models 7
   2.6. Conclusion 7
3. Appendix A – Source Code 8

1. Project Objective

The objective of the report is to explore **Cars** (employee information about their mode of transport as well as their personal and professional details like age, salary, work exp) dataset in R and prepare a Managerial Report by explaining the following points. This Managerial report consists of the following:

- Business Objective Statement
- Perform Exploratory data analysis(EDA)
- Illustrate the insights based on EDA
- Prepare the data for Analysis
- Model Performance using Logistic Regression, KNN and Naïve Models
- Model Performance using Bagging, Boosting and SMOTE models
- Summarize the accuracy of the model

The sample Cars dataset is:

1	Age	Gender	Engineer	MBA	Work Exp	Salary	Distance	license	Transport
2	28	Male	1	0	5	14.4	5.1	0	2Wheeler
3	24	Male	1	0	6	10.6	6.1	0	2Wheeler
4	27	Female	1	0	9	15.5	6.1	0	2Wheeler
5	25	Male	0	0	1	7.6	6.3	0	2Wheeler
6	25	Female	0	0	3	9.6	6.7	0	2Wheeler
7	21	Male	0	0	3	9.5	7.1	0	2Wheeler
8	23	Male	1	1	3	11.7	7.2	0	2Wheeler
9	23	Male	0	0	0	6.5	7.3	0	2Wheeler
10	24	Male	1	0	4	8.5	7.5	0	2Wheeler
11	28	Male	1	0	6	13.7	7.5	1	2Wheeler
12	26	Male	0	0	4	12.6	7.5	0	2Wheeler
13	21	Male	0	1	3	10.6	7.7	0	2Wheeler
14	22	Female	1	0	2	8.5	8.1	0	2Wheeler
15	24	Male	1	0	6	12.7	8.7	0	2Wheeler
16	27	Male	0	1	8	15.6	9	0	2Wheeler
17	28	Female	0	0	10	19.7	9	0	2Wheeler
18	23	Male	1	0	2	8.8	9.2	1	2Wheeler
19	29	Female	0	0	7	14.6	9.2	0	2Wheeler
20	29	Male	1	0	9	23.8	9.4	0	2Wheeler
21	22	Female	1	1	2	8.5	9.5	0	2Wheeler
22	27	Female	1	0	5	12.8	9.7	0	2Wheeler
23	25	Female	1	0	6	11.6	10.1	0	2Wheeler
24	34	Male	1	1	14	36.9	10.4	1	2Wheeler
25	28	Male	1	0	5	14.7	10.5	1	2Wheeler

Note: The provided data set has 419 rows and 9 columns of data.

Business Objective: To build a model that helps to predict whether or not an employee will use Car as a mode of transport.

2. Step by step approach

We shall follow a step by step approach to arrive to the final conclusion as follows:

1. Environment set up and Data import
2. Perform Exploratory data analysis(EDA)
3. Build Logistic Regression, KNN and Naïve Models
4. Build Bagging, Boosting and SMOTE models
5. Compare the models
6. Conclusion

2.1. Environment Set up and Data Import

Please refer Appendix A for Source Code.

2.2. Perform Exploratory Data Analysis(EDA)

Read the Cars Data and explore the variables

carsData <- read.csv("Cars.csv", header=T)

As the data is having Gender and Transport as categorical variables, convert these two variables into factor variables.

Transport is the predictor variable as we are going to predict whether employee is going to use Car as the transport. Hence, Convert car type to 1 and remaining two types of transport to 0.

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

The proportion of Transport data is shown below.

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

         0          1
0.91626794 0.08373206
>
```

Majority class is the Other type of Transportation comprising 91.6% of data.

Minority class is ‘Car’ transport mode comprising 0.83% of data.

Divide the Data set into Train and Test Data in 70:30 ratio to build various models.

The most challenging aspect of this problem is build a model based on the minority class of the predictor variable.

We overcome this challenge by building various models using Logistic Regression, KNN, Naïve, Bagging, boosting and SMOTE models and by comparing these models using their model performance metrics.

2.3. Build Logistic Regression, KNN and Naïve Models

Build Logistic Regression model using the Train and Test Data sets.

```
> #logistic regression
> LogRmodel  = glm(Transport ~., data = cars_train, family = binomial)
> LogRpredTest = predict(LogRmodel, newdata = cars_test, type = "response")
> tabLogR = table(cars_test$Transport, LogRpredTest > 0.5)
> tabLogR

    FALSE TRUE
  0   115    0
  1     1   10
> sum(diag(tabLogR))/sum(tabLogR)
[1] 0.9920635
>
```

Result shows that Logistic Regression model predicts with 99.2% accuracy.

Now build KNN model with Train and Test Data sets.

```
> #KNN
> library(class)
> predKNNmodel = knn(train = cars_train[,1:8], test = cars_test[,1:8], cl = cars_train[,9
], k = 3, prob = T)
> tabKNN = table(cars_test$Transport, predKNNmodel)
> tabKNN
   predKNNmodel
      0   1
  0 114   1
  1   0  11
> sum(diag(tabKNN))/sum(tabKNN)
[1] 0.9920635
>
```

Result shows that KNN model predicts with 99.2% accuracy.

Build the Naïve model with Train and Test data sets.

```
> NBmodel = naiveBayes(Transport ~., data = cars_train)
> NBpredTest = predict(NBmodel, newdata = cars_test)
> tabNB = table(cars_test$Transport, NBpredTest)
> tabNB
   NBpredTest
      0   1
  0 113   2
  1   0  11
> sum(diag(tabNB))/sum(tabNB)
[1] 0.984127
```

Result shows that Naive model predicts with 98.4% accuracy.

Please refer Appendix A for Source Code.

2.4. Build Bagging, Boosting and SMOTE Models

Build Bagging model with Train and Test Data.

```
> BAGmodel <- bagging(as.numeric(Transport) ~.,data=cars_train, control=rpart.control(max
depth=10, minsplit=50))
> BAGpredTest = predict(BAGmodel, cars_test)
> tabBAG = table(cars_test$Transport,BAGpredTest > 0.5)
> tabBAGcars_train

> tabBAG = table(cars_test$Transport,BAGpredTest > 0.5)
> tabBAG

   TRUE
  0  115
  1   11
> sum(diag(tabBAG))/sum(tabBAG)
[1] 0.9126984
```

Bagging model predicts with 91.3% accuracy which is almost equal to the results Of Logistic Regression, KNN and Naïve Models.

Build Boosting Model with Train and Test Data.

```
> XGBpredTest = predict(XGBmodel, features_test)
> tabXGB = table(cars_test$Transport, XGBpredTest>0.5)
> tabXGB

   FALSE TRUE
  0   114    1
  1     1   10
> sum(diag(tabXGB))/sum(tabXGB)
[1] 0.984127
>
```

Boosting model predicts with 98.4% accuracy which is almost equal to the results Of Logistic Regression, KNN and Naïve Models.

Now, perform SMOTE Test to predict the best fit model.

```
> smote_features_test<-as.matrix(smote.test[,1:8])
> smote.test$smote.pred.class <- predict(smote.xgb.fit, smote_features_test)
> tabSmt = table(smote.test$Transport,smote.test$smote.pred.class>=0.5)
> tabSmt

   FALSE TRUE
  0   114    1
  1     0   11
> sum(diag(tabSmt))/sum(tabSmt)
[1] 0.9920635
>
```

SMOTE Test shows that the model is best fit with 99.2% accuracy.

Please refer Appendix A for Source Code.

2.5. Compare the Models

Model Name	Performance Metric
Logistic Regression	99.2%
KNN	99.2%
Naive	98.4%
Bagging	91.3%
Boosting	98.4%
SMOTE Test	99.2%

As shown in the above table, the performance metrics of all the models match each other. Hence, the model built is a best fit to predict whether or not an employee will use Car as a mode of transport.

2.6. Conclusion

As the performance metrics of all the models match closely with each other, we conclude that the model best fits in predicting whether an employee will use Car as mode of Transport.

3. Appendix A – Source Code

```
#============================================================================
# Data Analysis - Cars
#============================================================================
#Environment Set up and Data Import
#Set up working Directory
rm(list=ls())
set.seed(248)
setwd("C:/Users/Radhika/Desktop/R Programming/Project_MachineLearning")
getwd()
carsData <- read.csv("Cars.csv", header=T)
#convert the categorical variable Gender into numeric and factor data
carsData$Gender = as.factor(carsData$Gender)
carsData$Gender = as.numeric(carsData$Gender)
#There are three types of transport data in the given data set
#As we are going to predict whether employee is going to use Car as the transport
#Convert car type to 1 and remaining two types of transport to 0
#convert the categorical variable Transport into numeric and factor data
carsData$Transport = as.numeric(carsData$Transport)
carsData$Transport[carsData$Transport == 1] = 0
carsData$Transport[carsData$Transport == 2] = 1
carsData$Transport[carsData$Transport == 3] = 0
carsData$Transport = as.factor(carsData$Transport)
str(carsData)
#Check the proportion of Transport data
prop.table(table(carsData$Transport))
# In the above data, car is the minority class and other transport are the
# Majority class
#Split the data into Train and Test data sets
library(caTools)
split = sample.split(carsData$Transport, SplitRatio = 0.7)
cars_train = subset(carsData, split == T)
cars_test = subset(carsData, split == F)
#Check the proportion of Transport data in train and test
prop.table(table(cars_train$Transport))
prop.table(table(cars_test$Transport))
#In Training data set, the majority class is 91.7% and minority is 0.82%
#In Test data set, the majority class is 91.2% and minority is 0.87%
```

```r
#In Test data set, the majority class is 91.2% and minority is 0.87%
#logistic regression
LogRmodel  = glm(Transport ~., data = cars_train, family = binomial)
LogRpredTest = predict(LogRmodel, newdata = cars_test, type = "response")
tabLogR = table(cars_test$Transport, LogRpredTest > 0.5)
tabLogR
sum(diag(tabLogR))/sum(tabLogR)
# By applying logistic regression, this model is 99.2% accurate
#KNN
library(class)
predKNNmodel = knn(train = cars_train[,1:8], test = cars_test[,1:8], cl = cars_train[,9], k = 3, prob = T)
tabKNN = table(cars_test$Transport, predKNNmodel)
tabKNN
sum(diag(tabKNN))/sum(tabKNN)
# KNN also shows 99.2% accuracy to predict using this model.
install.packages("e1071")
library(e1071)
NBmodel = naiveBayes(Transport ~., data = cars_train)
NBpredTest = predict(NBmodel, newdata = cars_test)
tabNB = table(cars_test$Transport, NBpredTest)
tabNB
sum(diag(tabNB))/sum(tabNB)
#Using, naive model, this model is 98.4% accurate
#Bagging
install.packages("xgboost")
install.packages("caret")
install.packages("ipred")
install.packages("rpart")
library(xgboost)
library(caret)
library(ipred)
library(rpart)
BAGmodel <- bagging(as.numeric(Transport) ~.,data=cars_train, control=rpart.control(maxdepth=10, minsplit=50))
BAGpredTest = predict(BAGmodel, cars_test)
tabBAG = table(cars_test$Transport,BAGpredTest > 0.5)
tabBAG
```

```r
sum(diag(tabBAG))/sum(tabBAG)
#Bagging model shows the model is 91.2% accurate
#Boosting
features_train<-as.matrix(cars_train[,1:8])
label_train<-as.matrix(cars_train[,9])
features_test<-as.matrix(cars_test[,1:8])
XGBmodel = xgboost(
  data = features_train,
  label = label_train,
  eta = .001,
  max_depth = 5,
  min_child_weight = 3,
  nrounds = 10,
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,                    # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)
XGBpredTest = predict(XGBmodel, features_test)
tabXGB = table(cars_test$Transport, XGBpredTest>0.5)
tabXGB
sum(diag(tabXGB))/sum(tabXGB)
#Boosting shows, model is 98.4% accurate
#SMOTE
install.packages("DMwR")
library(DMwR)
table(carsData$Transport)
smote.train<-subset(carsData, split == T)
smote.test<-subset(carsData, split == F)
table(smote.train$Transport)
balanced.data <- SMOTE(Transport ~., smote.train, k = 5)
table(balanced.data$Transport)
smote_features_train<-as.matrix(balanced.data[,1:8])
smote_label_train<-as.matrix(balanced.data$Transport)
```

```r

smote.xgb.fit <- xgboost(
  data = smote_features_train,
  label = smote_label_train,
  eta = 0.001,
  max_depth = 5,
  nrounds = 10,
  nfold = 5,
  objective = "binary:logistic",  # for regression models
  verbose = 0,               # silent,
  early_stopping_rounds = 10 # stop if no improvement for 10 consecutive trees
)

smote_features_test<-as.matrix(smote.test[,1:8])
smote.test$smote.pred.class <- predict(smote.xgb.fit, smote_features_test)

tabSmt = table(smote.test$Transport,smote.test$smote.pred.class>=0.5)
tabSmt
sum(diag(tabSmt))/sum(tabSmt)
#Smote Test also shows the model is 99.2% accurate
```