CS 513-A Knowledge Discovery & Data mining

PROFESSOR : KHASHA DEHNAD

Final Project: Employee Promotion Prediction
Section A

Meet our team: Group 3

Venkata Sumanth Nagabhairu
CWID : 20012395

Sai Shruthi Sistla
CWID : 20016100

Ramya Sree Meduri
CWID : 20009063

Prathamesh Kshirsagar
CWID: 20011500

Employee Promotion Prediction

Source of Dataset:
https://www.kaggle.com/datasets/shivan118/hranalysis

Project Motive

- For the given employee details, we would like to predict if the employee is going to be promoted or not.
- Applied KNN, Nave Bayes, Random Forest, Logistic Regression, Support Vector Machine, Multilayer Perceptron, AdaBoost, and XGBoost & compare models and pick the best model which suits to our data set to predict target label promotion

Let load & see what we have in our train data

Loading the training dataset

```{r}
rm(list=ls())
train_df= read.csv("/Users/sumanth/Desktop/train.csv")
str(train_df)
summary(train_df)
head(train_df)
```

```
'data.frame':   54808 obs. of  14 variables:
 $ employee_id         : int  65438 65141 7513 2542 48945 58896 20379 16290 73202
28911 ...
 $ department          : chr  "Sales & Marketing" "Operations" "Sales & Marketing"
"Sales & Marketing" ...
 $ region              : chr  "region_7" "region_22" "region_19" "region_23" ...
 $ education           : chr  "Master's & above" "Bachelor's" "Bachelor's"
"Bachelor's" ...
 $ gender              : chr  "f" "m" "m" "m" ...
 $ recruitment_channel : chr  "sourcing" "other" "sourcing" "other" ...
 $ no_of_trainings     : int  1 1 1 2 1 2 1 1 1 1 ...
 $ age                 : int  35 30 34 39 45 31 31 33 28 32 ...
 $ previous_year_rating: int  5 5 3 1 3 3 3 3 4 5 ...
 $ length_of_service   : int  8 4 7 10 2 7 5 6 5 5 ...
 $ KPIs_met..80.       : int  1 0 0 0 0 0 0 0 0 1 ...
 $ awards_won.         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ avg_training_score  : int  49 60 50 50 73 85 59 63 83 54 ...
 $ is_promoted         : int  0 0 0 0 0 0 0 0 0 0 ...
```

Let Know what each column is

employee_id: unique id of every employee

Department: name of department they are working

Region: which region does he belongs to

Education: education level of the employee

Gender: gender of employee

recruitment_channel: how he/she is recruited

No*of*trainings: number of trainings taken by employee

age : age of employee

Previous*year*rating: previous year rating of that employee

Length*of*service: how long employee being working in the company

KPIs_met >80%: able to meet 80% of KPIs

awards_won:

Avg*training*score:

is_promoted: our target label

Let check is there any missing values in our data set

Checking for missing values

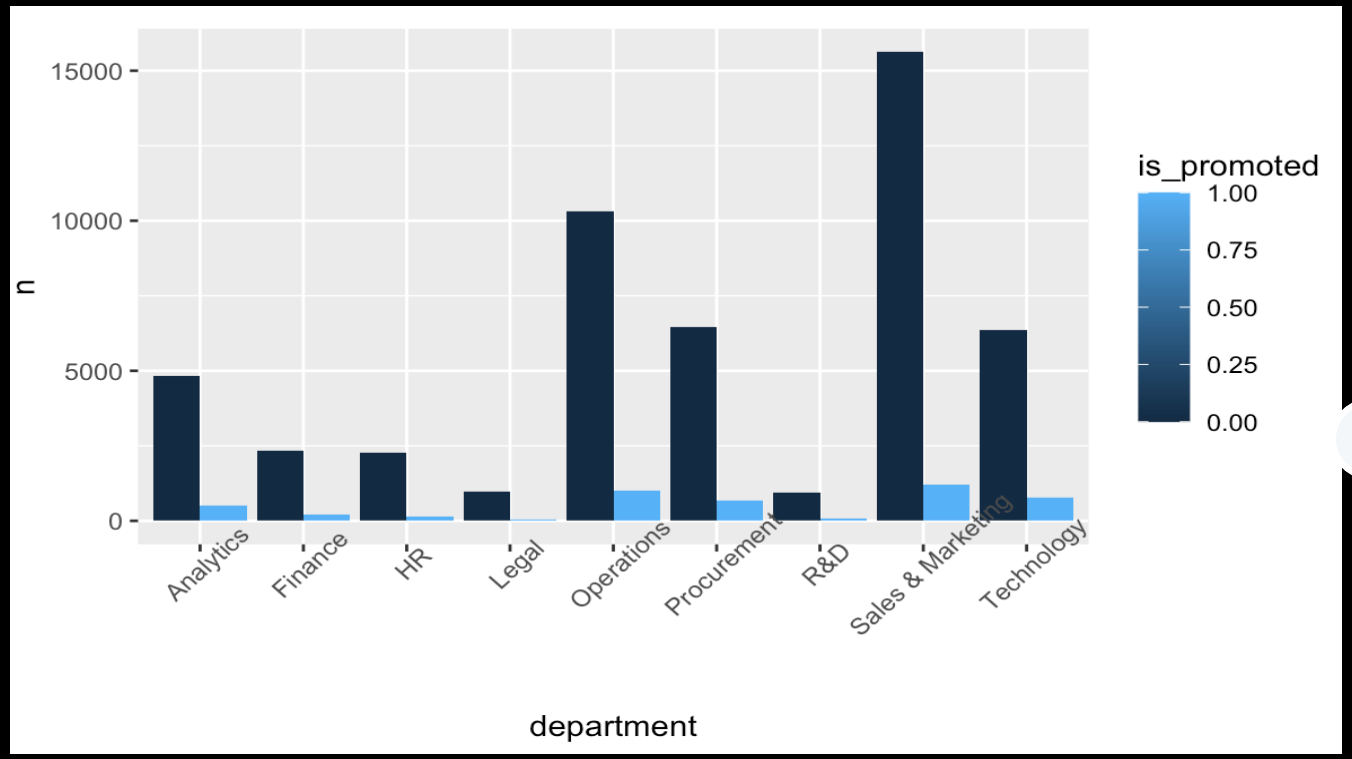
```
sum(is.na(train_df))
```

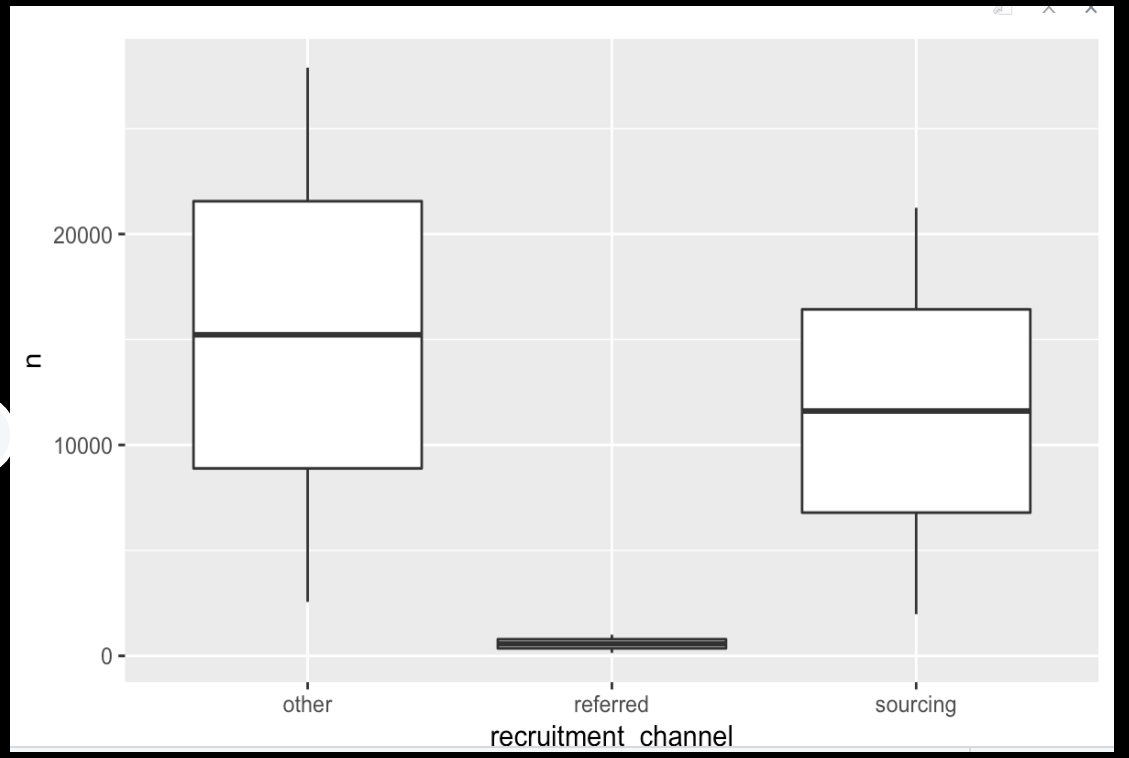
```
## [1] 4124
```

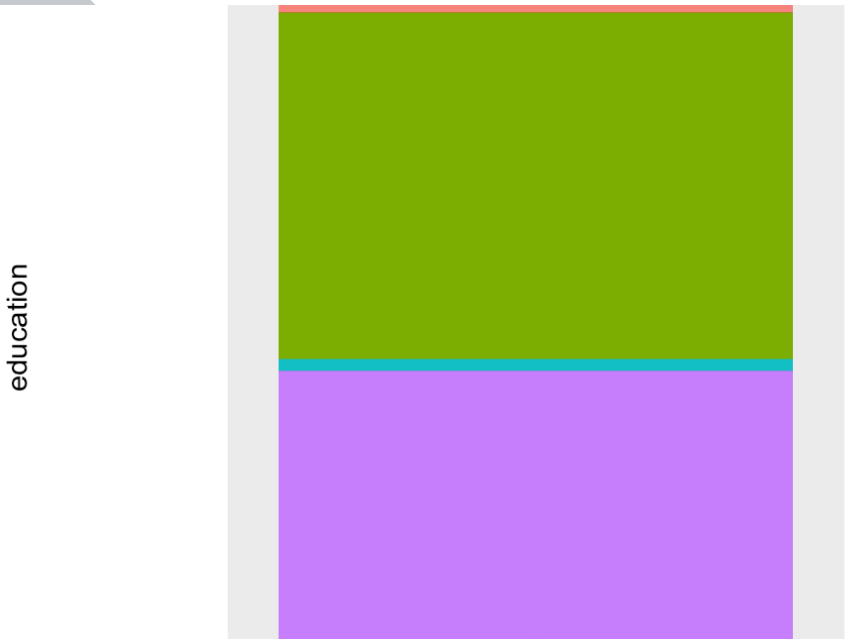
- Filled the columns containing the missing values with the necessary substitutes.
- Use mode for filling missing values
- Removing employee_id because every employee has it and it does not have impact on our target variable

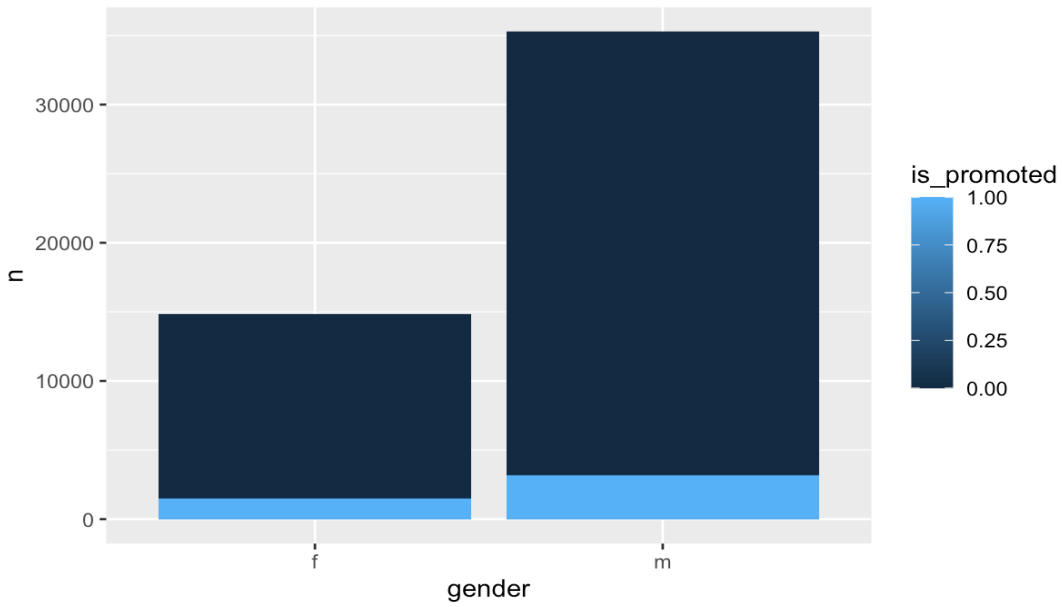
DATA VISUALIZATION

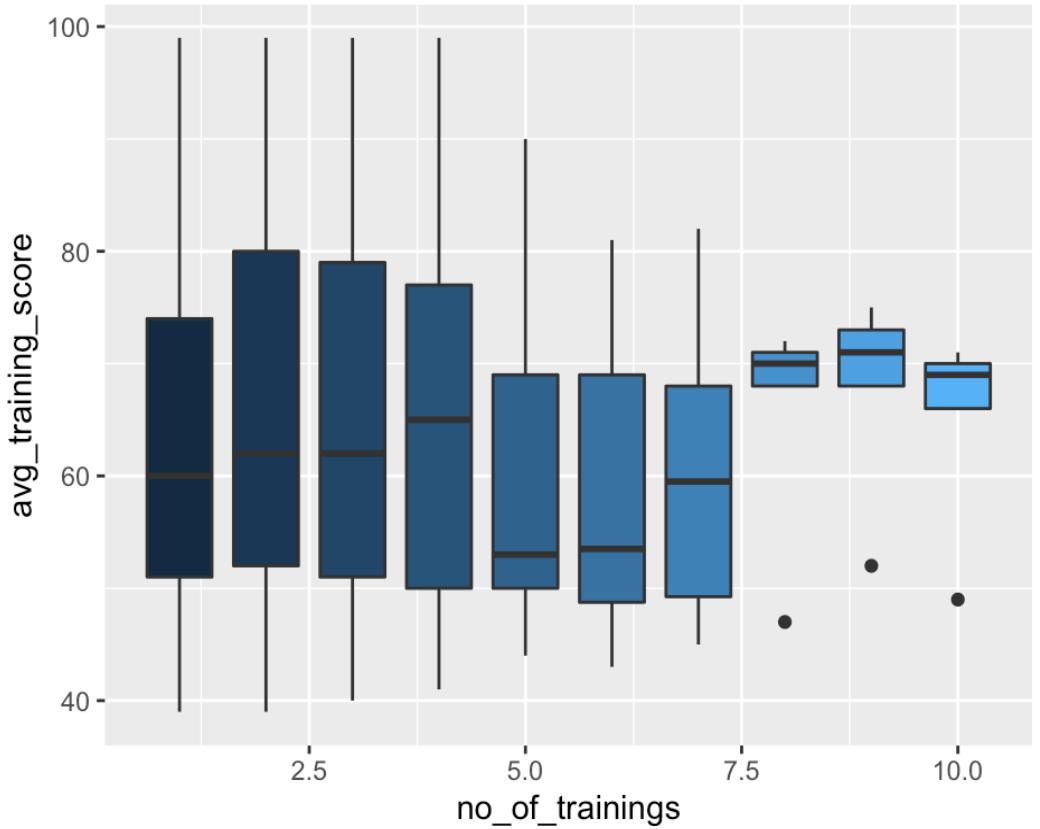
Let get how each label is related to our target label

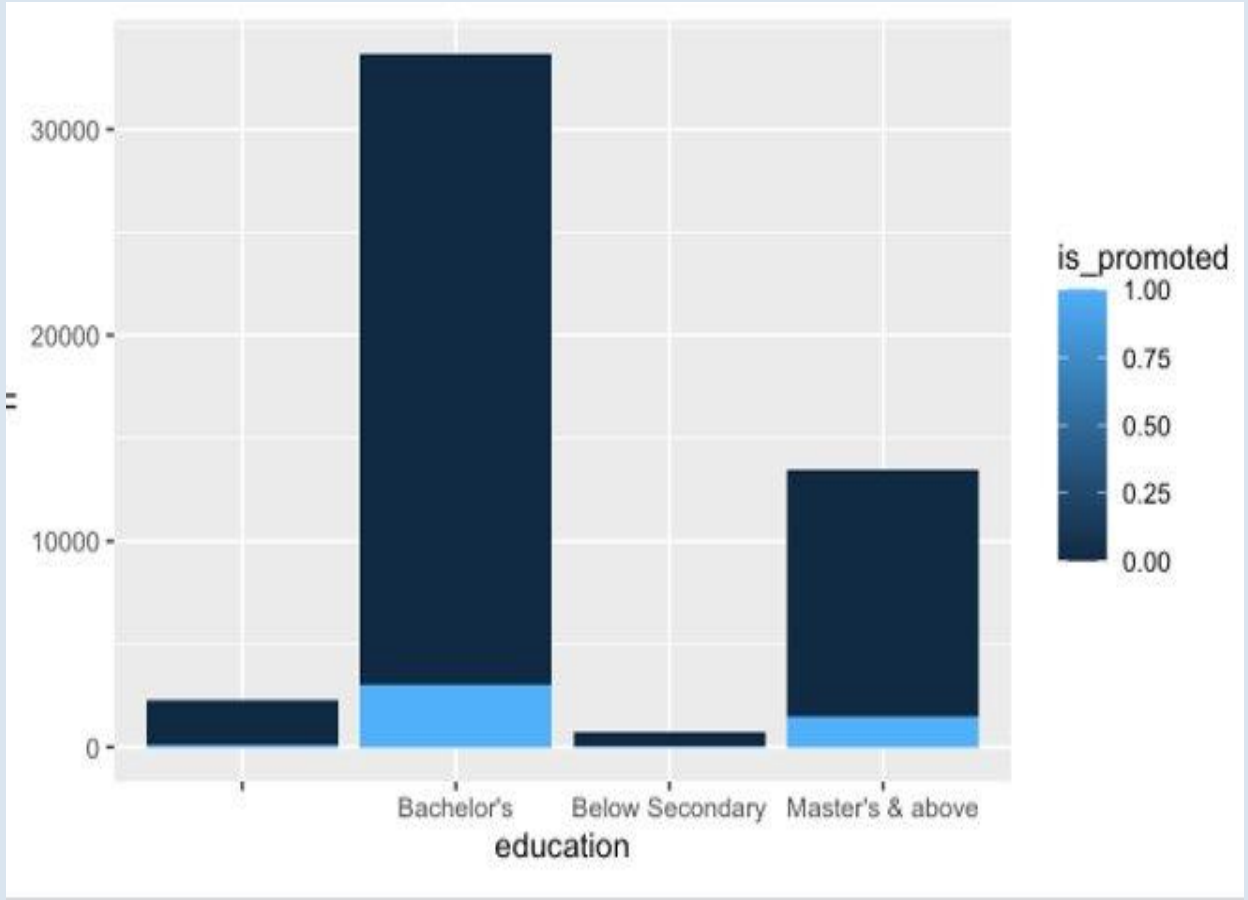


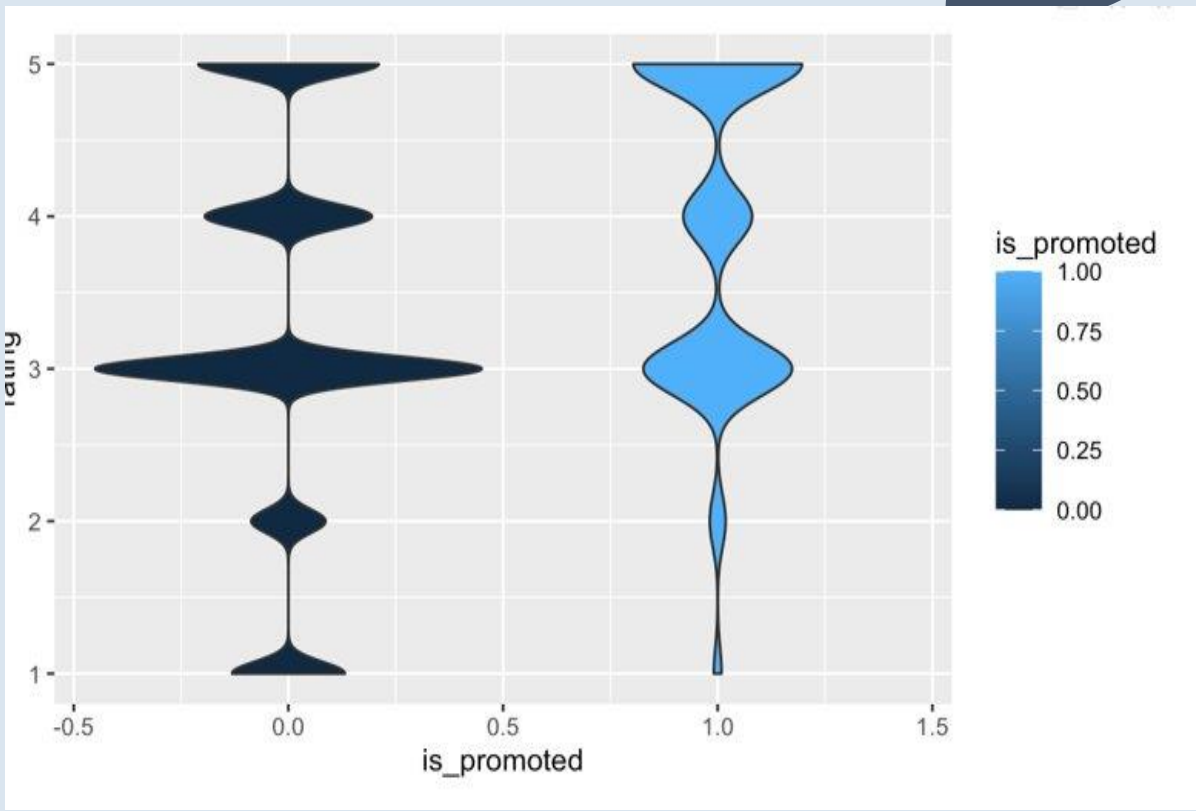


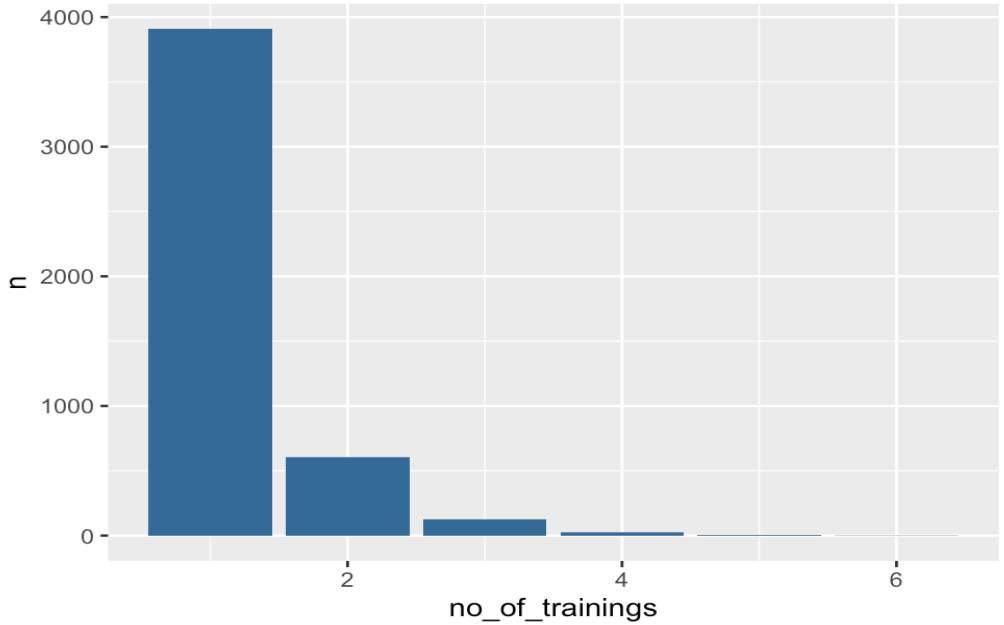

education
education
Bachelor's
Below Secondary
Master's & above

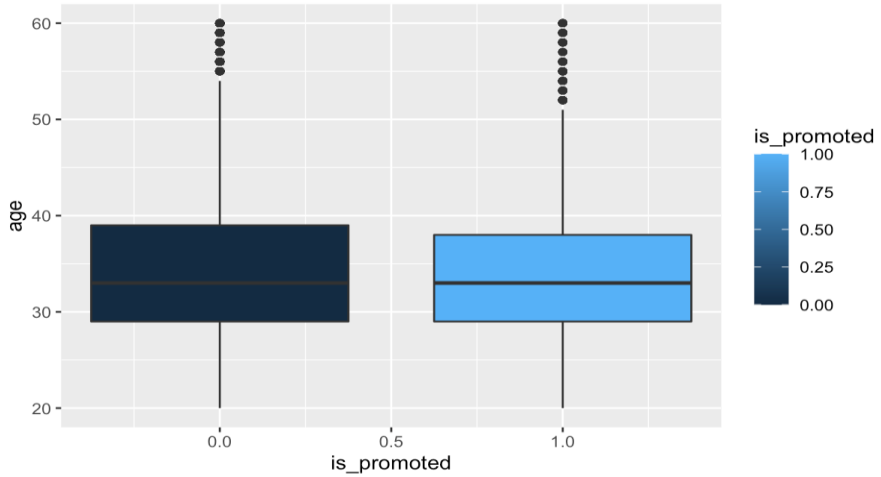

30000
20000
10000
0
n
f
m
gender
is_promoted
1.00
0.75
0.50
0.25
0.00

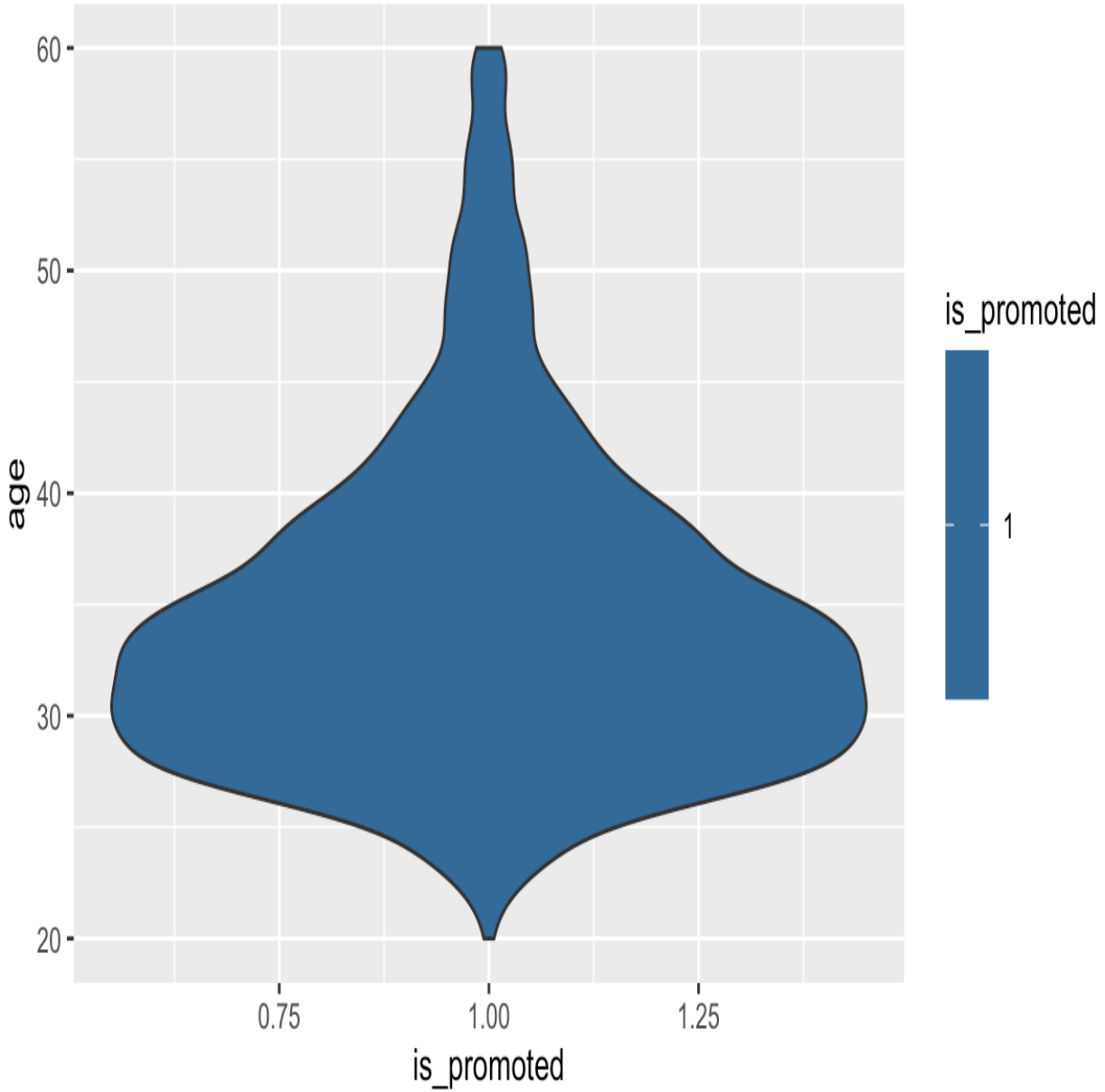

100
80
60
40
avg_training_score
2.5
5.0
7.5
10.0
no_of_trainings
no_of_training
10.0
7.5
5.0
2.5

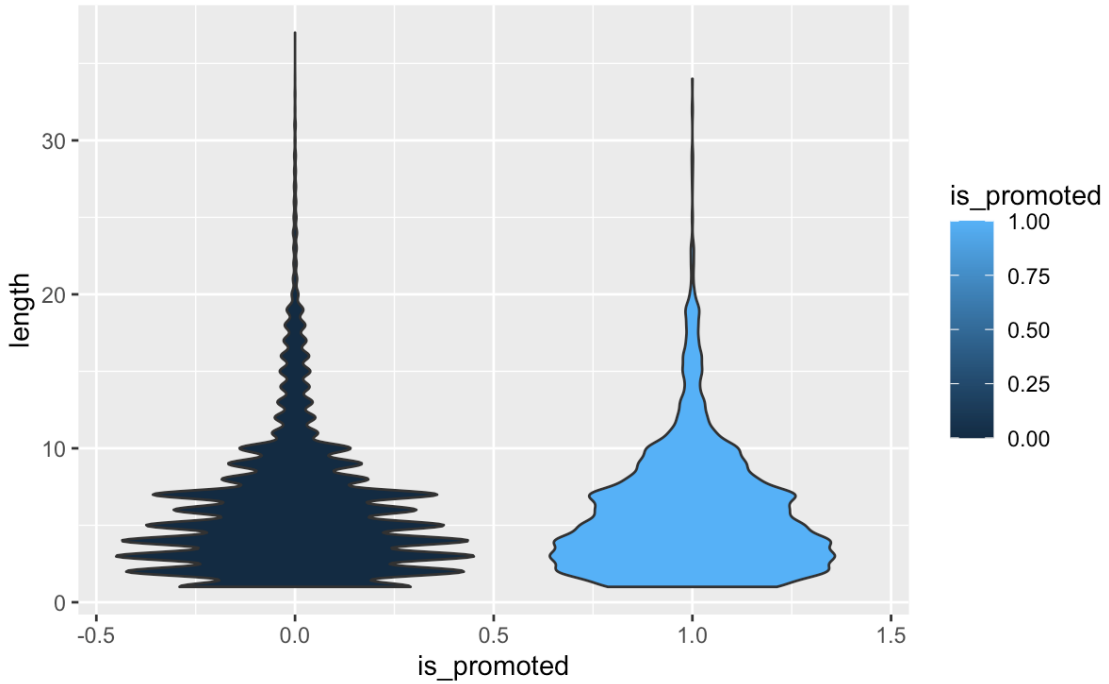

30000
20000
10000
0
Bachelor's
Below Secondary
Master's & above
education
is_promoted
1.00
0.75
0.50
0.25
0.00

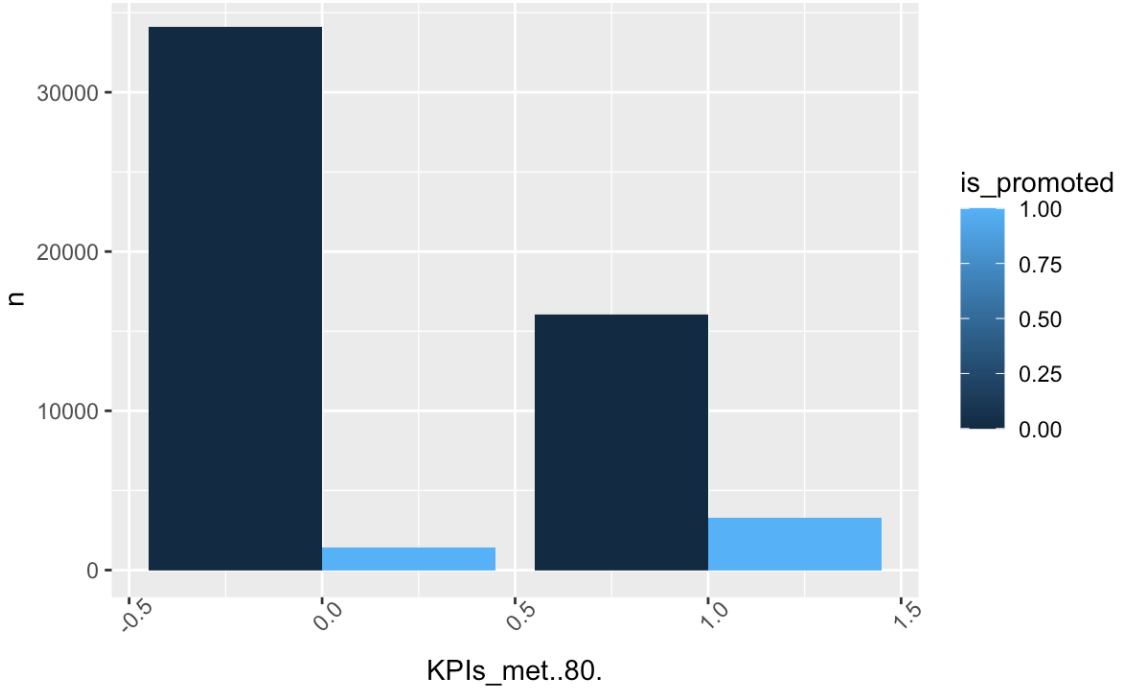

5
4
3
2
1
-0.5
0.0
0.5
1.0
1.5
is_promoted
is_promoted
1.00
0.75
0.50
0.25
0.00


4000
3000
2000
1000
0
n
2
4
6
no_of_trainings
is_promoted
1


60
50
40
30
20
age
0.0
0.5
1.0
is_promoted
is_promoted
1.00
0.75
0.50
0.25
0.00


60
50
40
30
20
age
0.75
1.00
1.25
is_promoted
is_promoted
1


length
30
20
10
0
-0.5
0.0
0.5
1.0
1.5
is_promoted
is_promoted
1.00
0.75
0.50
0.25
0.00


n
30000
20000
10000
0
-0.5
0.0
0.5
1.0
1.5
KPIs_met..80.
is_promoted
1.00
0.75
0.50
0.25
0.00

Inputs we get from EDA

If the employee rating for the previous year is more than 3, then the chances of promotion are high

If the training score of the employee is at least 70, then the chances of promotion are high

Most of the employees between the age group of 25 to 45 have higher chances of promotion

Number of training has an inverse relation with promotion

If the employee is hired through Sourcing or other then the chance of promotion is hiring than that for referred people

Male and Female ratio in the company is 2:1 and their promotion ratio is also in the ratio 2:1

If the education of the employee is Bachelors or Masters&above, then the chances of promotion are high

Sales & marketing, operations, procurement, technology, and analytics are the top 5 departments to which the promoted employees belong

Split the data set into Train and validation

- We need to predict the is_promoted label in test data, so we can't apply our models directly to test data set because we don't know how our model predict the results.
- So, we split our train data set into train data and validation data in (70:30) .
- So, we can implement our models and check them with our validation data.
- We used KNN, Nave Bayes, Random Forest, Logistic Regression, Support Vector Machine, Multilayer Perceptron, AdaBoost, and XGBoost
- We evaluate the model using various metrics like Confusion Matrix, Precision, Accuracy, Recall, ROC, F1score.
- Based the above evaluations we can pick our best model and using that we can predict the promotions of employees in our test data set.

KNN

- k refers to the number of neighbors
- quality of the algorithm depends on the value of k & distance measure.

```
myknn <- knn(X_train,X_valid,cl=y_train, k=5)
```

Accuracy

```
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(confusion_matrix)
```

```
## [1] 91.6805
```

Precision

```
# precision = True Positive / (True Positive + False Positive)
precision <- function(x){x[4]/sum(x[4],x[2])}
precision(confusion_matrix)
```

```
## [1] 0.5496536
```

Recall

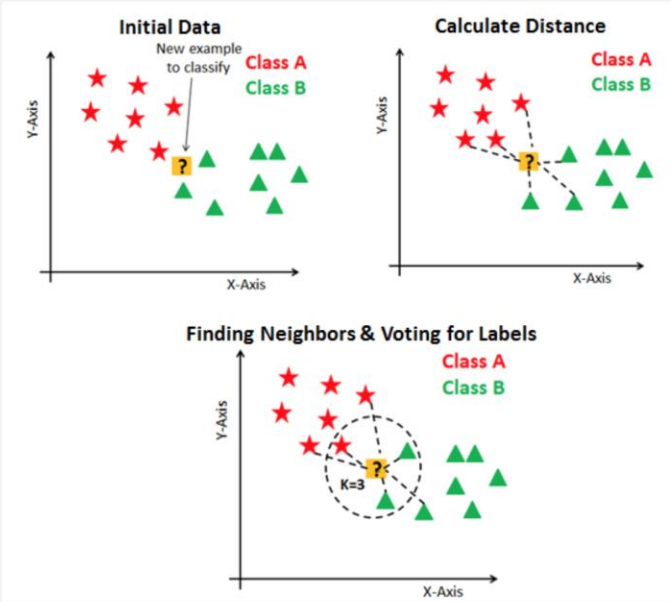
```
#recall= True Positive / (True Positive + False Negative)
recall <- function(x){x[4]/sum(x[4],x[3])}
recall(confusion_matrix)
```

```
## [1] 0.164592
```

F1 value

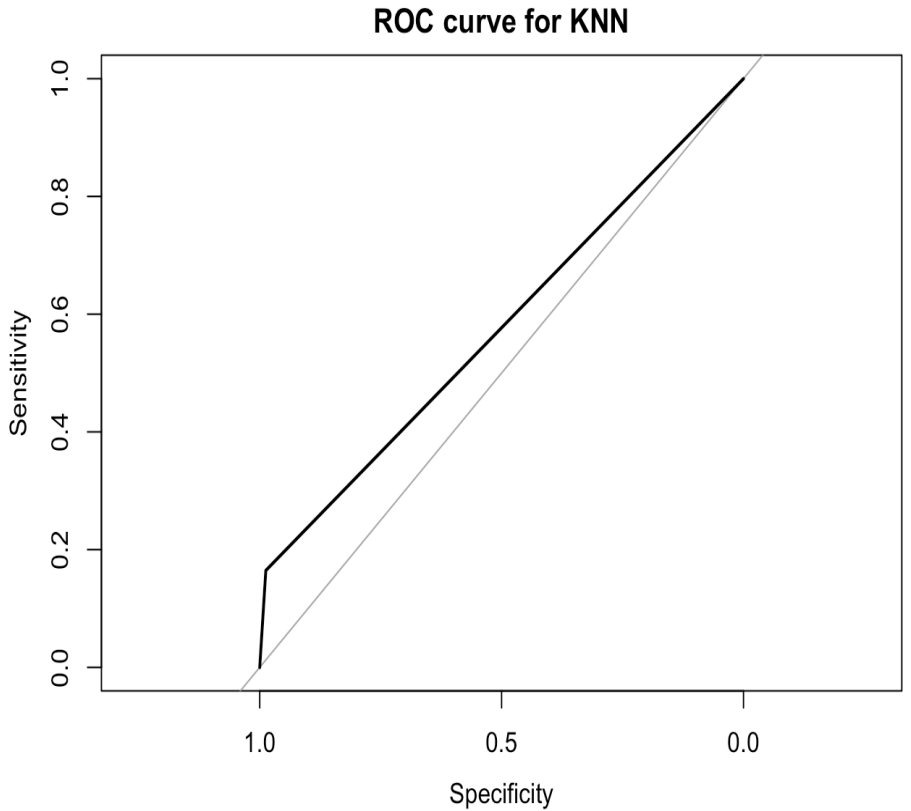
```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

```
## [1] 0.2533262
```




```
## Area under the curve: 0.576
```

```
plot(roc_score ,main ="ROC curve for KNN")
```

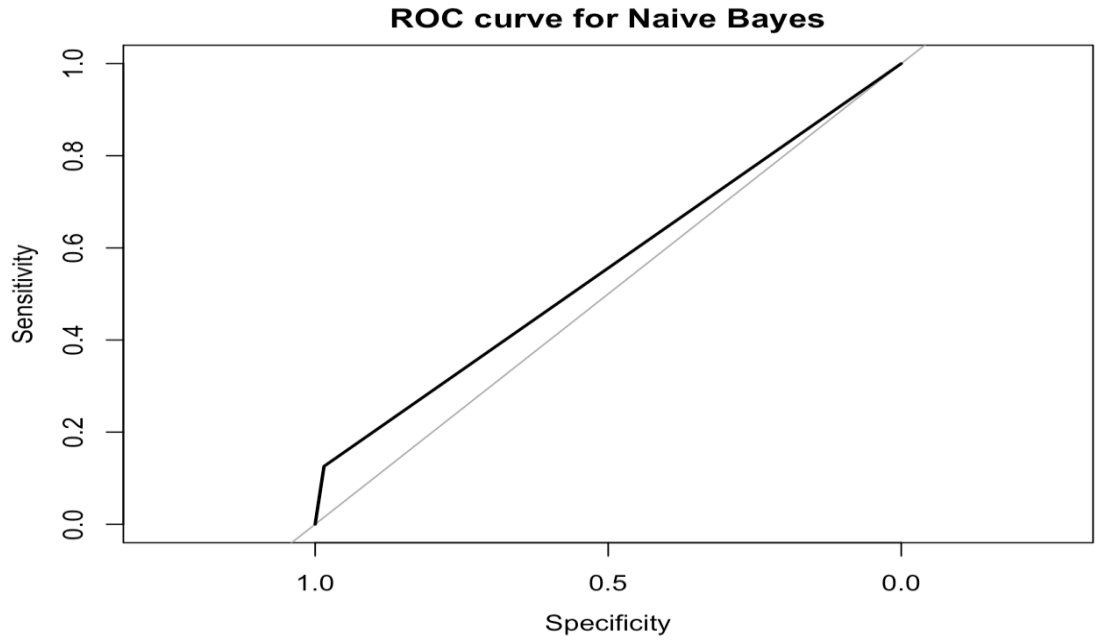


Naive Bayes

- Naive Bayes is a probabilistic classification algorithm that works on Bayes Theorem. Bayes theorem . Bayes Theorem operates on conditional probability


```
## Area under the curve: 0.5553
```

```
plot(roc_score ,main ="ROC curve for Naive Bayes")
```




```
accuracy(confusion_matrix)
```

```
## [1] 91.11717
```

```
precision(confusion_matrix)
```

```
## [1] 0.4375
```

```
recall(confusion_matrix)
```

```
## [1] 0.1258645
```

F1 value

```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

```
## [1] 0.1954887
```

Random Forest

```
accuracy(confusion_matrix)
```

```
## [1] 93.75
```

```
precision(confusion_matrix)
```

```
## [1] 0.8726236
```

```
recall(confusion_matrix)
```

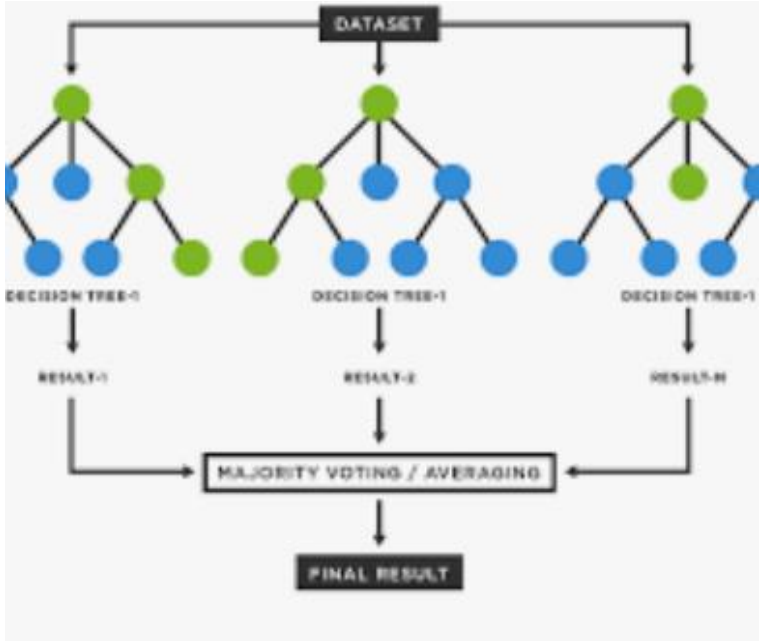
```
## [1] 0.3174274
```

f1 value

```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

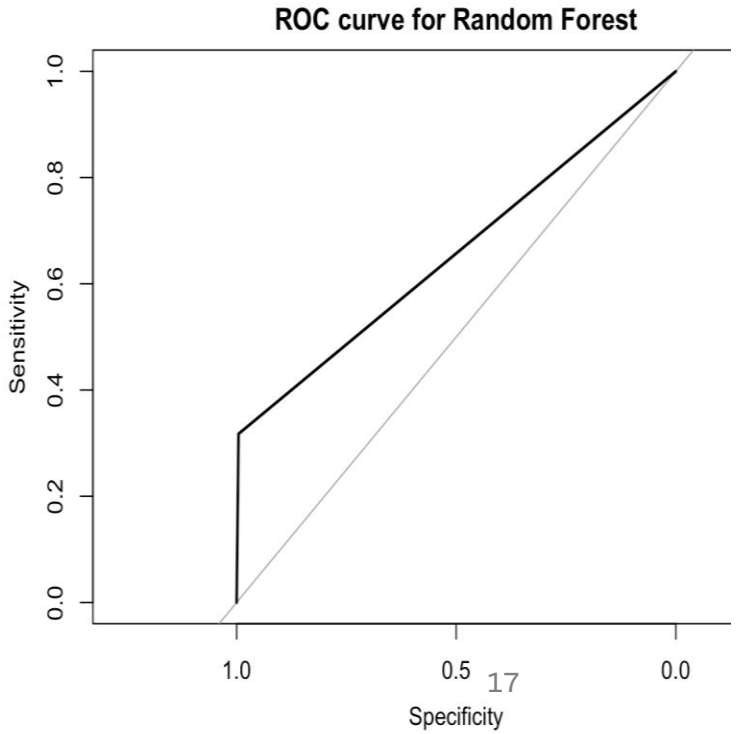
```
## [1] 0.4655172
```

Random forest is a machine learning model that consists of large number of individual decision trees that operate together as an ensemble. Each individual decision tree will predict the class of the target variable and the class with maximum number of votes is the model's prediction.




```
## Area under the curve: 0.6565
```

```
plot(roc_score ,main ="ROC curve for Random Forest")
```



Logistic Regression

Logistic Regression is a **supervised classification algorithm** that is used for prediction when the target variable is binary. It helps explain the relationship between one dependent binary variable and one or more independent variable.

```
accuracy(confusion_matrix)
```

```
## [1] 91.6212
```

```
precision(confusion_matrix)
```

```
## [1] 0.5964912
```

```
recall(confusion_matrix)
```

```
## [1] 0.07053942
```

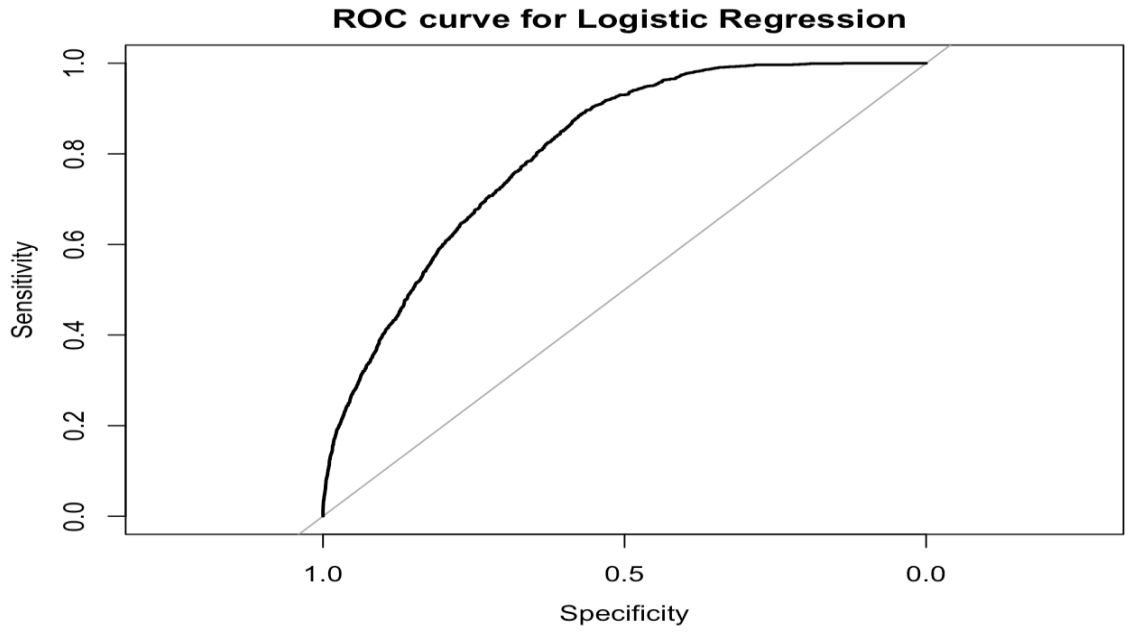
f1 value

```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

```
## [1] 0.1261596
```

```
## Area under the curve: 0.8062
```

```
plot(roc_score ,main ="ROC curve for Logistic Regression ")
```




```
model <- glm(is_promoted~., family="binomial", data=traindata)
```

Support Vector Machine

Support Vector Machine is a machine learning model whose objective is to find a hyperplane in an N dimensional space that distinctly classifies the data points. N here corresponds to the number of features . The margin of the classifier is maximized using support vectors which are data points that are close to the hyperplane and influence the position and orientation of the hyperplane .

```
accuracy(confusion_matrix)
```

```
## [1] 86.18952
```

```
precision(confusion_matrix)
```

```
## [1] 0.1717472
```

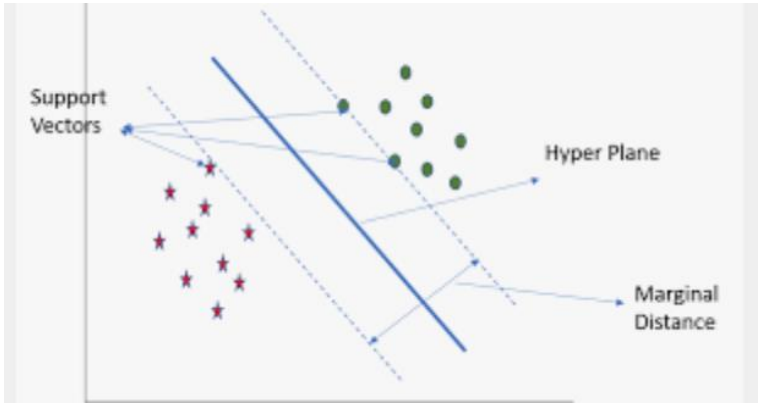
```
recall(confusion_matrix)
```

```
## [1] 0.159751
```

f1 value

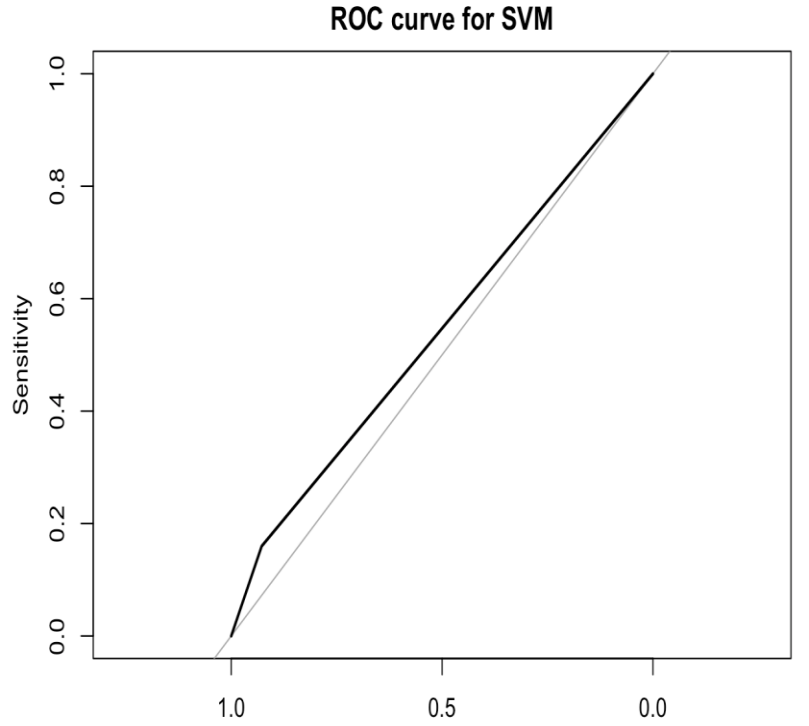
```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

```
## [1] 0.1655321
```




```
## Area under the curve: 0.5437
```

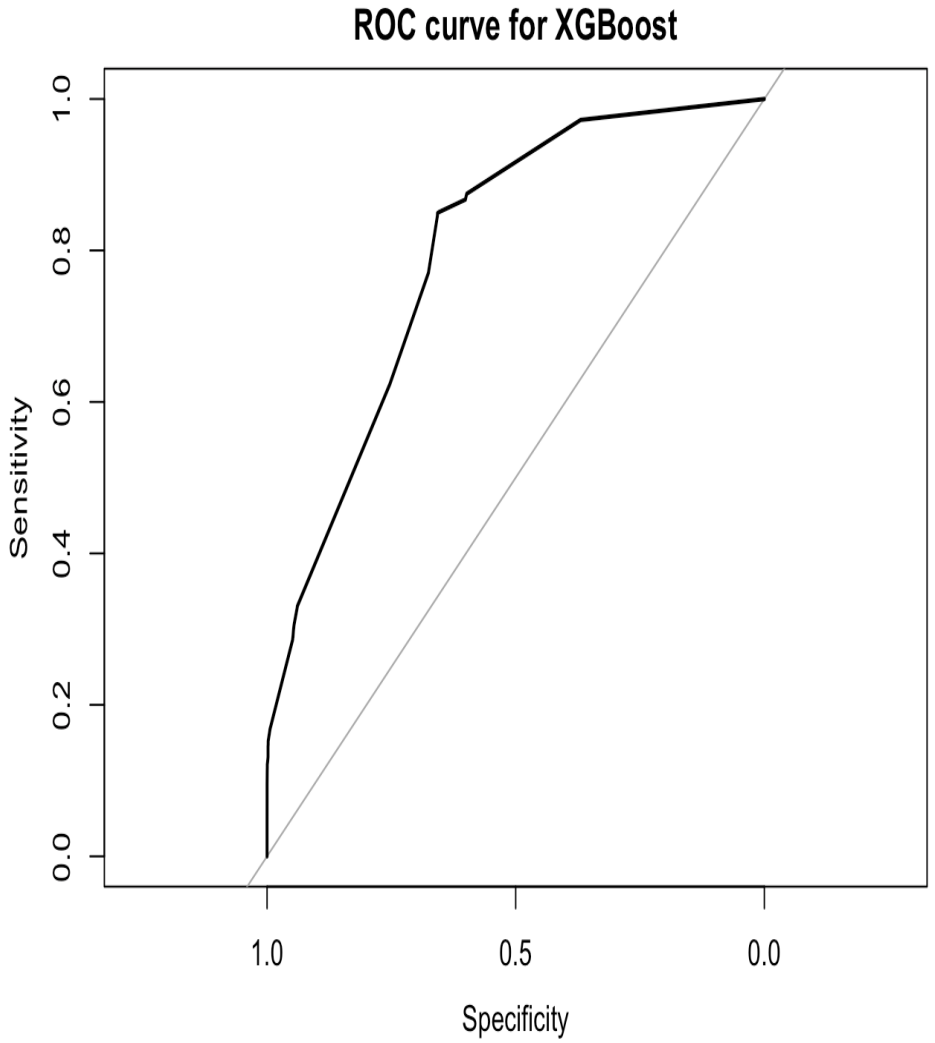
```
plot(roc_score ,main ="ROC curve for SVM")
```




```
svm_c<- svm(formula =is_promoted ~ .,data = traindata,type = 'C-classification',kernel = 'sigmoid')
```

XGBoost

XGBoost is a decision tree based ensemble Machine Learning algorithm that uses a gradient boosting framework. It is an implementation of Gradient Boosted Decision Trees. It can work on classification, regression and user defined prediction problems.




```
bst <- xgboost(data = as.matrix(X_train), label = as.matrix(y_train), max_depth = 2,
               eta = 0.5, nthread = 2, nrounds = 5, objective = "binary:logistic")
```

```
accuracy(confusion_matrix)
```

```
## [1] 92.4158
```

```
precision(confusion_matrix)
```

```
## [1] 0.120332
```

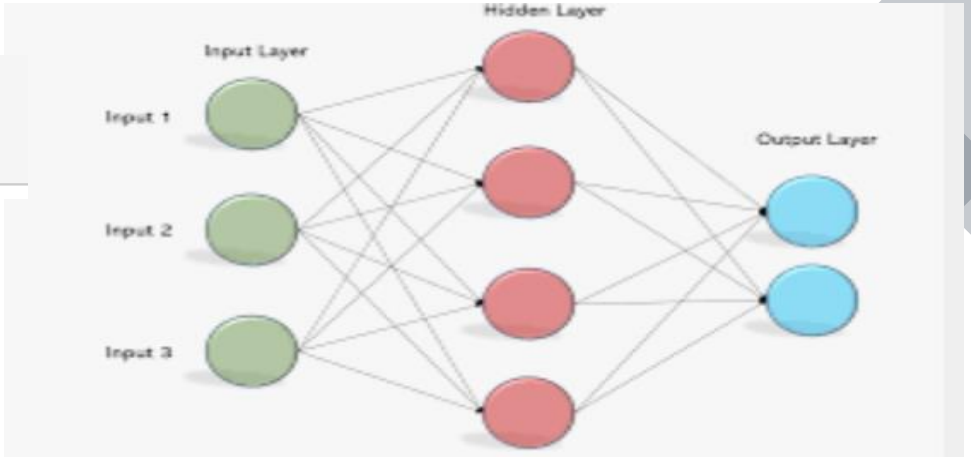
```
recall(confusion_matrix)
```

```
## [1] 0.961326
```

Multilayer Perceptron

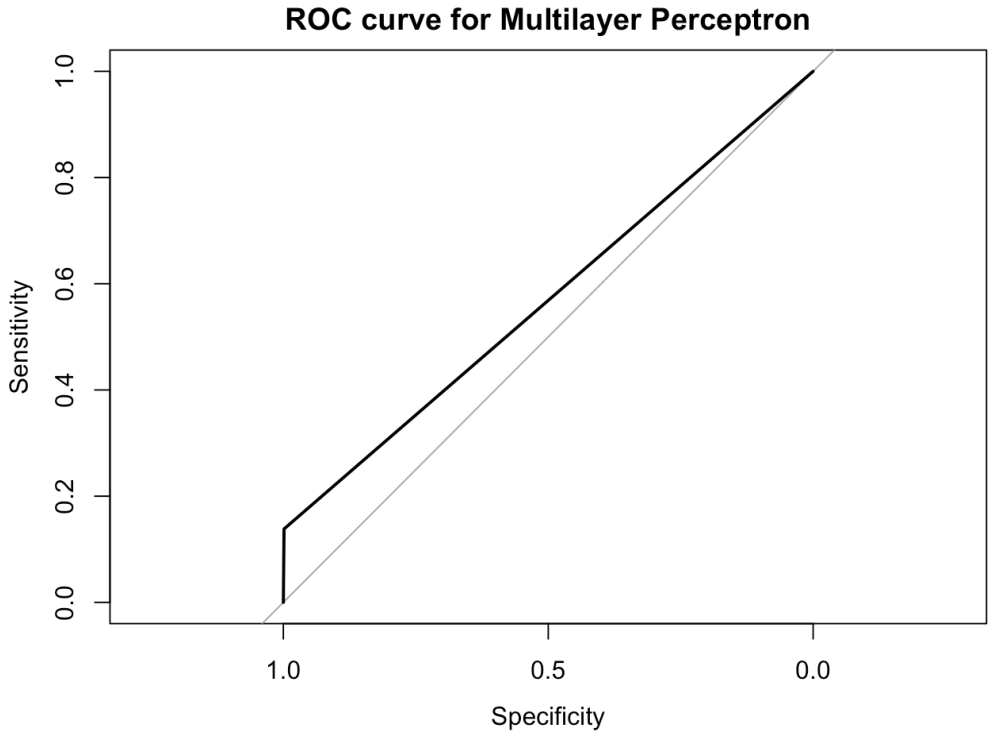
A multilayer perceptron is neural network where the mapping between the input and the output variables is non-linear. It has an input and output layer and one or more hidden layers with. Many neutrons stacked together. It is a feedforward algorithm.

```
library(nnet)
nn5 <-nnet(is_promoted ~ .,data = traindata,size = 3,maxit = 150)
```




```
## Area under the curve: 0.5685
```

```
plot(roc_score ,main ="ROC curve for Multilayer Perceptron")
```




```
accuracy(confusion_matrix)
```

```
## [1] 92.48695
```

```
precision(confusion_matrix)
```

```
## [1] 0.1383126
```

```
recall(confusion_matrix)
```

```
## [1] 0.9049774
```

AdaBoost

AdaBoost is an ensemble learning algorithm that uses an iterative approach to learn from the mistakes of weak classifiers and turn them into strong ones. It is based on the concept that a single classier may not be able to accurately predict the class of the target variable, but when we group multiple weak classifiers which each learning from the others wrong predicted objects, a strong model can be built.

Accuracy

```
accuracy(confusion_matrix)
```

```
## [1] 92.39801
```

```
precision(confusion_matrix)
```

```
## [1] 0.9555556
```

```
recall(confusion_matrix)
```

```
## [1] 0.1189488
```

f1 value

```
#F1=2 * (Precision * Recall) / (Precision + Recall)
p<- precision(confusion_matrix)
r<- recall(confusion_matrix)
f1<- (2*(p*r))/(p+r)
f1
```

```
## [1] 0.2115621
```

```
model_adaboost <- boosting(is_promoted~., data=traindata, boos=TRUE, mfinal=5)
summary(model_adaboost)
```

Deciding the model to predict test data set

evaluated the models using various metrics like Confusion Matrix, Precision, Accuracy, Recall, ROC, F1score.

We can observe XGBoost and Logistic Regression giving better results to our train and validation data

So, we use these 2 models to predict the promotion of employees in our test data

Predicting promotion using XGBoost

pred <- predict(bst, as.matrix(test_df))

```
finalres<-as.numeric(pred > 0.5)
table(finalres)
```

```
## finalres
##     0     1
## 23264   226
```

```
test_df['is_promoted']<-finalres
head(test_df)
```

```
##   department    region education gender recruitment_channel no_of_trainings
## 1      1.000 0.5454545 0.3333333      1                   1             0.00
## 2      0.250 0.8484848 0.3333333      0                   0             0.00
## 3      0.875 0.1212121 0.3333333      1                   0             0.00
## 4      0.625 0.3333333 0.3333333      0                   0             0.25
## 5      0.125 0.6363636 0.3333333      1                   1             0.00
## 6      0.625 0.9393939 0.3333333      1                   1             0.00
##     age previous_year_rating length_of_service KPIs_met..80. awards_won.
## 1 0.100                0.50        0.00000000              1           0
## 2 0.275                0.50        0.12121212              0           0
## 3 0.275                0.00        0.09090909              0           0
## 4 0.275                0.25        0.24242424              0           0
## 5 0.250                0.75        0.18181818              0           0
## 6 0.400                0.50        0.03030303              0           0
##   avg_training_score is_promoted
## 1          0.6333333           0
## 2          0.2000000           0
## 3          0.1333333           0
## 4          0.4333333           0
## 5          0.3666667           0
## 6          0.4833333           0
```

By using XGBoost we predicted 226 employees will promoted

Predicting promotion using Logistic Regression

```
predicted <- predict(model, test_df[-13], type="response")
finalres<-as.numeric(predicted > 0.5)
table(finalres)
```

```
## finalres
##     0     1
## 23259   231
```

```
test_df['is_promoted']<-finalres
head(test_df)
```

```
##   department    region education gender recruitment_channel no_of_trainings
## 1      1.000 0.5454545 0.3333333      1                   1              0.00
## 2      0.250 0.8484848 0.3333333      0                   0              0.00
## 3      0.875 0.1212121 0.3333333      1                   0              0.00
## 4      0.625 0.3333333 0.3333333      0                   0              0.25
## 5      0.125 0.6363636 0.3333333      1                   1              0.00
## 6      0.625 0.9393939 0.3333333      1                   1              0.00
##     age previous_year_rating length_of_service KPIs_met..80. awards_won.
## 1 0.100                0.50         0.00000000              1           0
## 2 0.275                0.50         0.12121212              0           0
## 3 0.275                0.00         0.09090909              0           0
## 4 0.275                0.25         0.24242424              0           0
## 5 0.250                0.75         0.18181818              0           0
## 6 0.400                0.50         0.03030303              0           0
##   avg_training_score is_promoted
## 1          0.6333333           0
## 2          0.2000000           0
## 3          0.1333333           0
## 4          0.4333333           0
## 5          0.3666667           0
## 6          0.4833333           0
```

By using Logistic Regression **we predicted 231 employees will promoted**

Thank you