PROJECT REPORT

PREDICTING EMPLOYEE ATTRITION

Submitted towards the partial fulfillment of the criteria for an award of Genpact Data Science Prodegree by Imarticus

Submitted By:
MANISHA ANAND

Course and Batch: PRODEGREE DATA SCIENCE_DSP31

Acknowledgments

We are using this opportunity to express my gratitude to everyone who supported us throughout this group project. We are thankful for their aspiring guidance, invaluably constructive criticism, and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on some issues related to the project.

Further, we were fortunate to have **DR. VINOD MURTI** as our mentor. He has readily shared his immense knowledge in data analytics and guides us in a manner that outcome resulted in enhancing our data skills.

We wish to thank, all the faculties, as this project utilized knowledge gained from every course that formed the DSP program.

We certify that the work done by us for conceptualizing and completing this project is original and authentic.

Date: December 22, 2020

Place: Varanasi, UP

Certificate of Completion

I hereby certify that the project titled “PREDICTING EMPLOYEE ATTRITION” was undertaken and completed under my supervision by Nikhil Rajan from the batch of DSP-31 (MAY-2020)

Mentor: Dr. Vinod Murti

Date: December 22, 2020

Place –Varanasi, UP

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 TITLE

We have been given the task of Predicting Employee Attrition and uncovering the factors that cause the Employees to leave the organization. Attrition is defined as a gradual reduction of the size of the workforce through normal means, such as retirement, resignation, or death. This is normal in any business or industry. The attrition rate is defined as the rate of shrinkage in size or number. The study was mainly undertaken to identify the level of employee's attitude, the dissatisfaction factors they face in the organization, and for what reasons they prefer to change their job.

1.2 OBJECTIVE OF THE STUDY

To foster a pattern during the stint of an employee as to when and what factors lead to attrition based on available facts and historical data. To analyze the parameters of a dissatisfied employee and adopt a curated set of innovative ideas focusing on employee needs and desires. To consistently retain a low percentage of attrition to not only acquire top-notch employees but discretely invite the interests of potential investors. To deploy an ascertained model that forecasts attrition beforehand and directly revamps a smooth decision-making process. To constitute the state of the art retention policies and framework that would mutually benefit both the organization and employees while fixating on the reasons for attrition mentioned above.

1.3 DATA SOURCE

The data set for IBM HR Analytics Employee Attrition & Performance was available from KAGGLE. Data contains different attributes of an employee and the target variable Attrition. Employee Number is the primary key. We use this dataset to predict employee churn.

1.4 TOOLS & TECHNIQUES

Tools: R – 4.0.3

R STUDIO – 1.3.1093

Techniques:

The data would be divided into Test and Train data. Will use the training data to build models and finally apply it to test data to measure the performance and accuracy of various models.

Understand the data
EDA and segmentation
Data cleaning
Feature Engineering
Model building
Testing and cross-validation
Final results

CHAPTER 2: DATA PREPARATION

We would now be focusing on preparing the given data to fit a different model. To achieve this, we will be looking at the sequence of steps

2.1 EXPLORATORY DATA ANALYSIS (EDA)

Exploratory data analysis is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

2.1.1 DATA IMPORT AND STATISTICAL ANALYSIS

- **Importing Libraries**
  library (stats)
  library (dplyr)
  library (psych)

- **Aspects of Data**
  dim (Attrition)
  #No. of observation – 1470; No. of variables – 35

  str (Attrition)
  # there are 16 No. of categorical variables and 17 No. of continuous variables and 1 target variable i.e. Attrition

  summary (Attrition)
  describe (Attrition)
  #this step will help us give the statistical analysis of raw data like the shape of data, data types, and min & max values, Mean, Median, Quantile values, Range, Skewness, Kurtosis, Standard Deviation, and Standard Error.

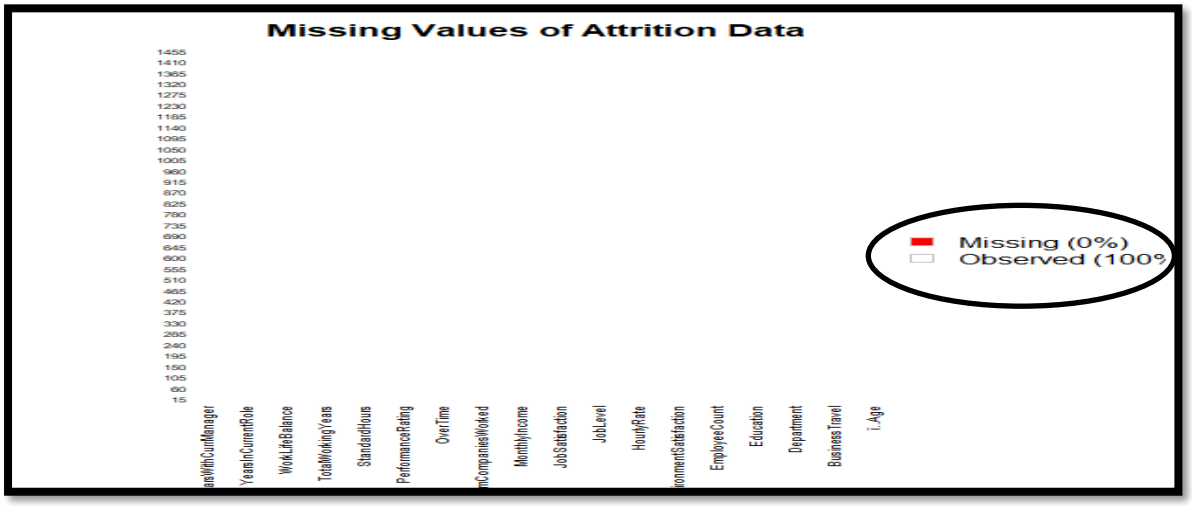
2.2 DATA CLEANSING

2.2.1 CHECKING THE NULL VALUES(NA)

#To check missing values in the whole dataset
is.na (Attrition)

#to get the count of Na's in the data sets
table (is.na(Attrition))

```
#Viewing Missing values using graphs or plots
    missmap (Attrition,
                col=c ('Red', 'White'), x.cex = .6, ylab = "Row Number",
y.cex = .5,
                main = "Missing Values of Attrition Data",rank.order = T)
```

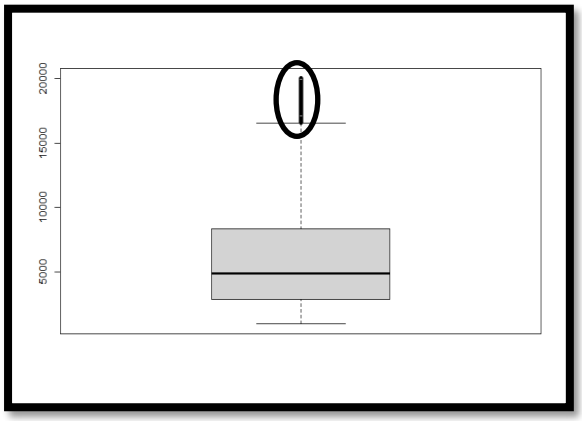


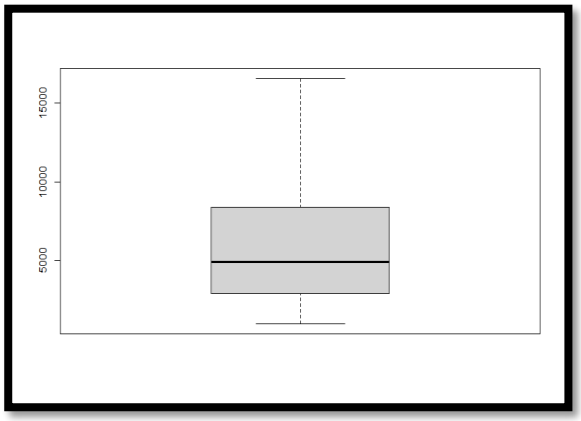
There were no missing values in the entire data set of 1470 observations.

2.2.2 REMOVING OUTLIERS

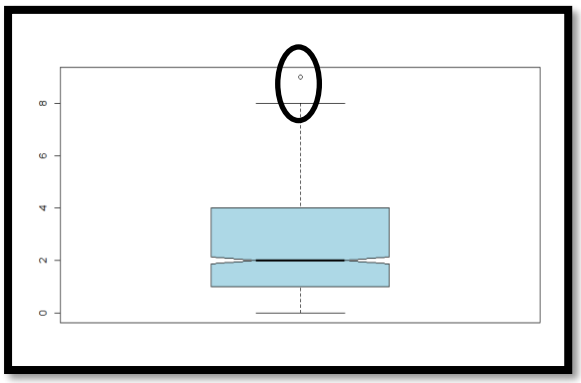
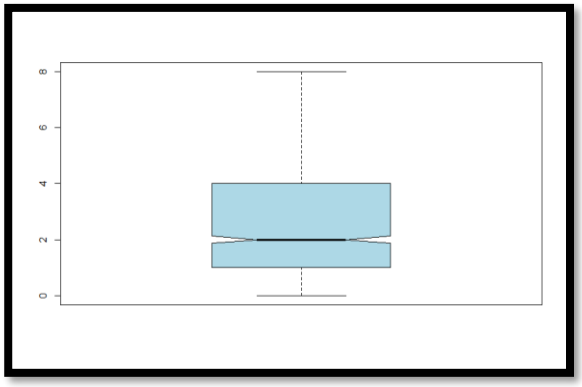
It was observed that there are many Outliers in variables Monthly Income, No. of Companies worked, Stock options level, Total working Hours, Training time last year, Years at the company, Years in Current role, Years since last promotion, Years with current Manager.

```
# removing outliers from monthly income
bx<- boxplot (employee$MonthlyIncome)
quantile (employee$MonthlyIncome, seq(0,1,0.02))
bx$stats
employee$MonthlyIncome<- ifelse (employee$MonthlyIncome >16555 ,
16400,employee$MonthlyIncome)
```

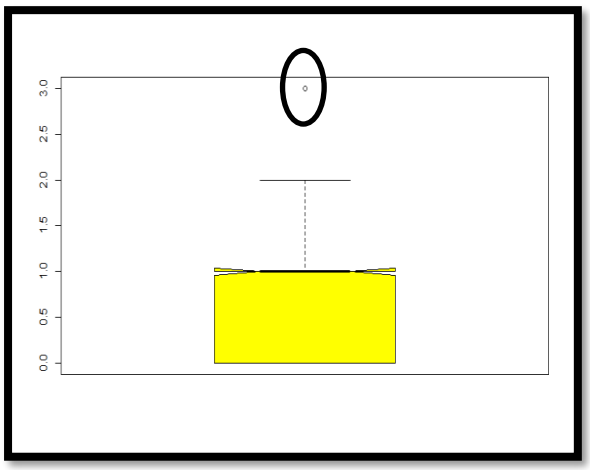
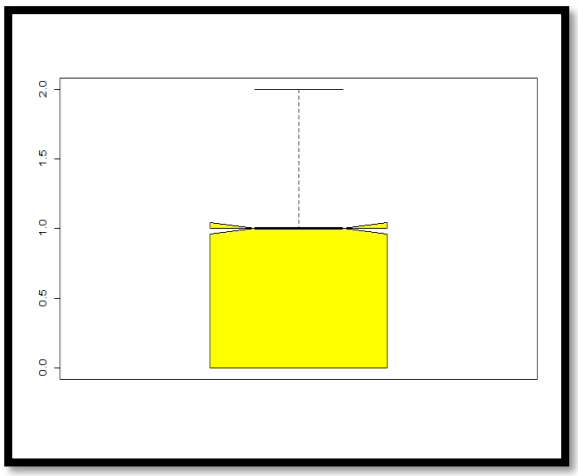




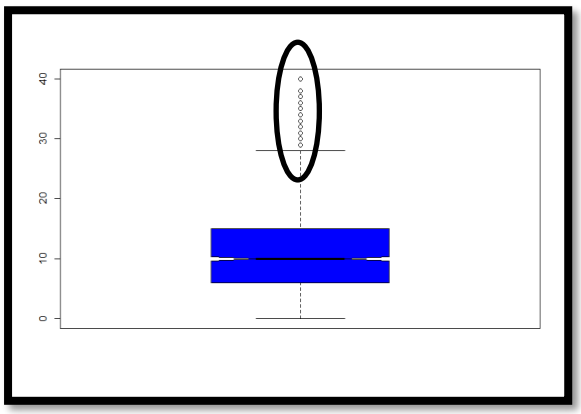
```
# removing outliers from no. of companies worked
bx<- boxplot(employee$NumCompaniesWorked,col = "lightblue", notch= TRUE)
quantile (employee$NumCompaniesWorked,seq(0,1,0.02))
bx$stats
employee$NumCompaniesWorked<- ifelse(employee$NumCompaniesWorked>8 , 7,employee$NumCompaniesWorked)
```

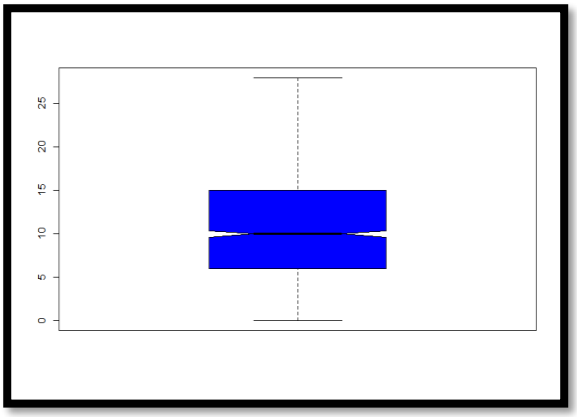



```
# removing outliers from stock option level
bx<- boxplot(employee$StockOptionLevel,col = "yellow", notch= TRUE)
quantile(employee$StockOptionLevel,seq(0,1,0.02))
bx$stats
employee$StockOptionLevel<- ifelse(employee$StockOptionLevel> 2 , 2,employee$StockOptionLevel)
```

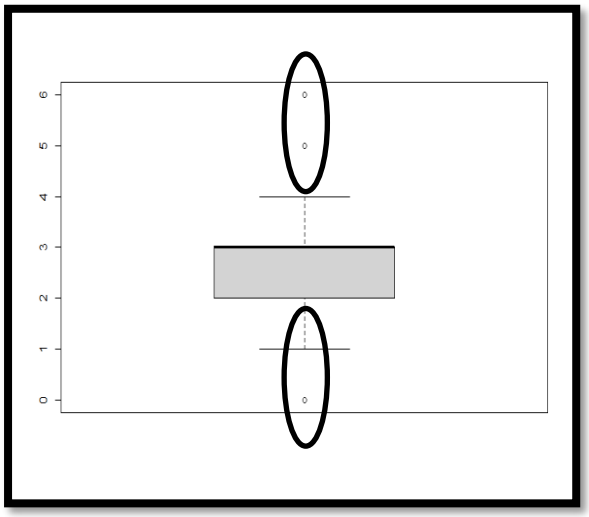



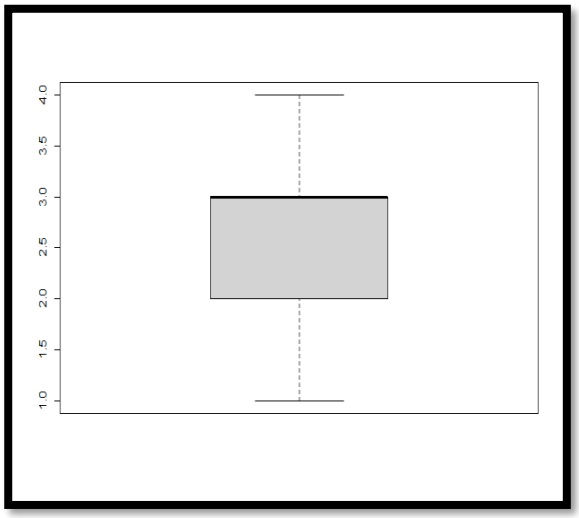
```
# removing outliers from total working year
bx<- boxplot(employee$TotalWorkingYears,col = "blue", notch= TRUE)
Quantile(employee$TotalWorkingYears, seq(0,1,0.02))
bx$stats
employee$TotalWorkingYears<-
ifelse(employee$TotalWorkingYears>28,26,employee$TotalWorkingYears)
```





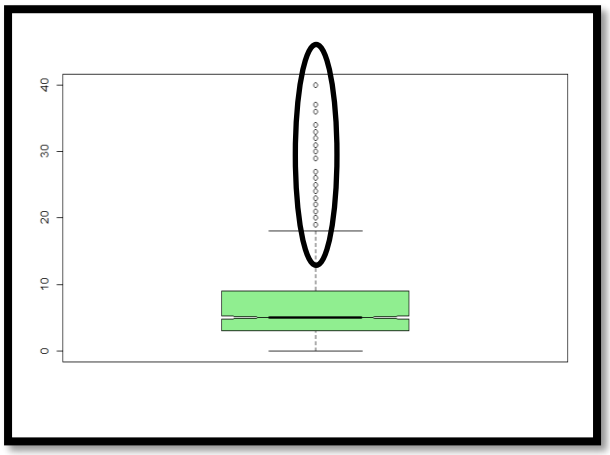

```
# removing outliers from training time last year
bx<- boxplot(employee$TrainingTimesLastYear)
quantile(employee$TrainingTimesLastYear, seq(0,1,0.02))
bx$stats
employee$TrainingTimesLastYear <-
ifelse(employee$TrainingTimesLastYear>4,4,employee$TrainingTimesLastYe
ar)
employee$TrainingTimesLastYear<-
ifelse(employee$TrainingTimesLastYear<1,1,employee$TrainingTimesLastYe
ar)
```

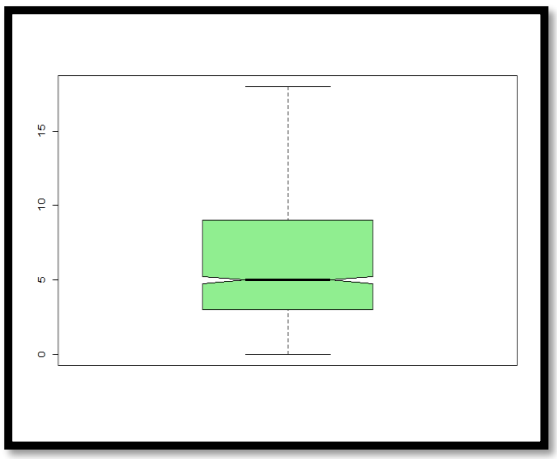




 # removing outliers from years at company

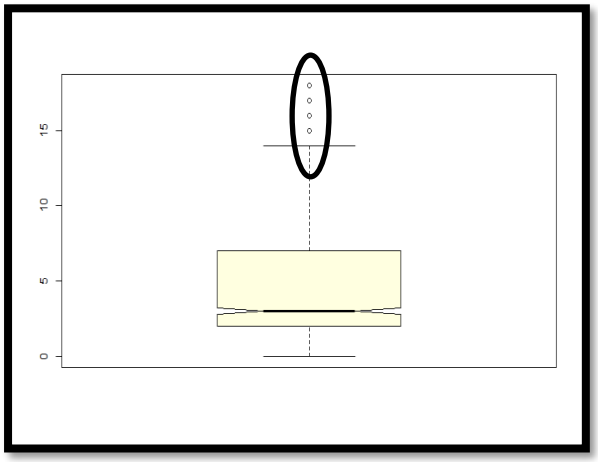
```
bx<- boxplot(employee$YearsAtCompany,col = "lightgreen", notch= TRUE)
quantile(employee$YearsAtCompany, seq(0,1,0.02))
bx$stats
employee$YearsAtCompany<-ifelse(employee$YearsAtCompany>18 ,
17,employee$YearsAtCompany)
```

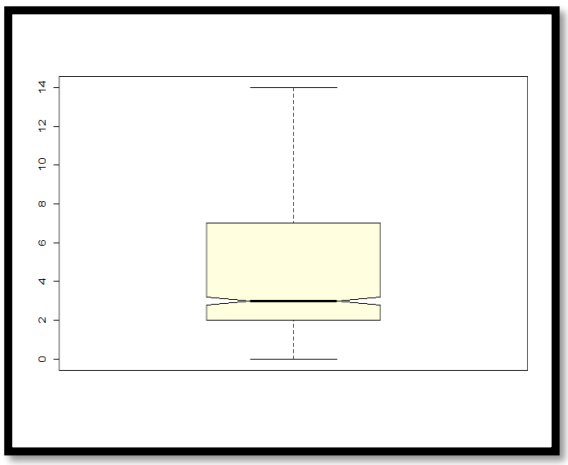




 # removing outliers from years in current role

```
bx<- boxplot(employee$YearsInCurrentRole,col = "lightyellow", notch= TRUE)
quantile(employee$YearsInCurrentRole , seq(0,1,0.02))
bx$stats
employee$YearsInCurrentRole<- ifelse(employee$YearsInCurrentRole >14,
13,employee$YearsInCurrentRole)
```

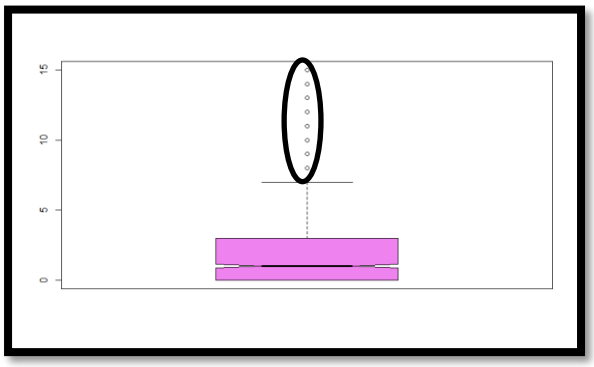
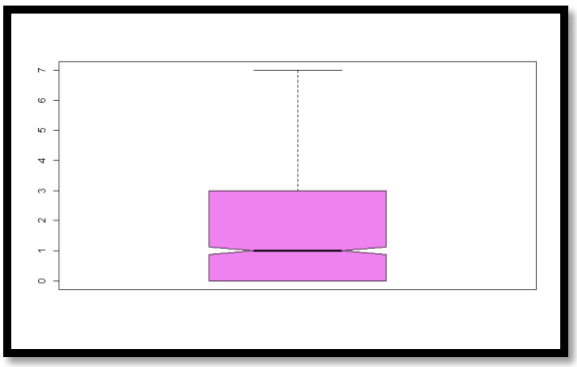




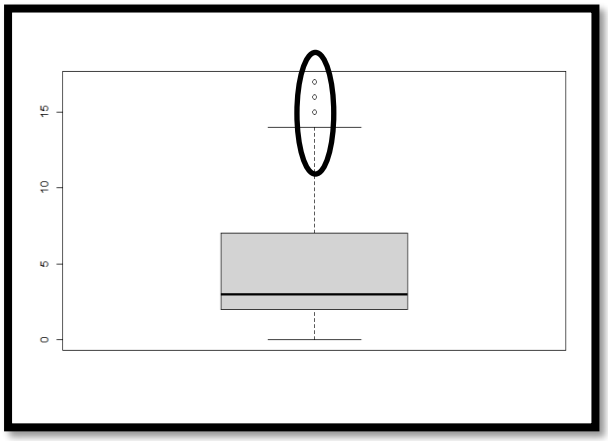
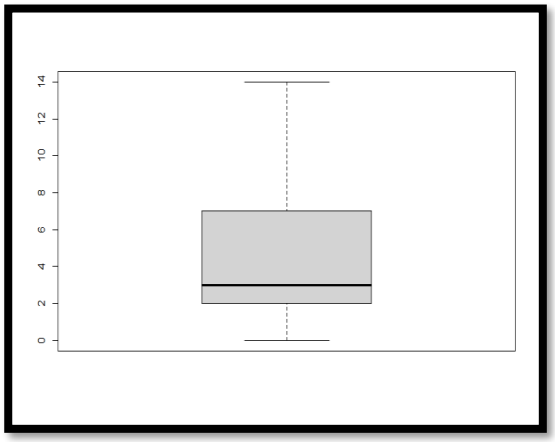
 # removing outliers from years since last promotion

```
bx<- boxplot(employee$YearsSinceLastPromotion,col = "violet", notch=
TRUE)
quantile(employee$YearsSinceLastPromotion , seq(0,1,0.02))
bx$stats
```

employee$YearsSinceLastPromotion <-
ifelse(employee$YearsSinceLastPromotion>7,6,employee$YearsSinceLastPr
omotion)

 # removing outliers from years with current manager
bx<- boxplot(employee$YearsWithCurrManager)
quantile(employee$YearsWithCurrManager , seq(0,1,0.02))
bx$stats
employee$YearsWithCurrManager <-
ifelse(employee$YearsWithCurrManager>14,13,employee$YearsWithCurrMa
nager

CHAPTER 3: DATA VISUALISATION

3.1 Conversion of Categorical to Numerical.

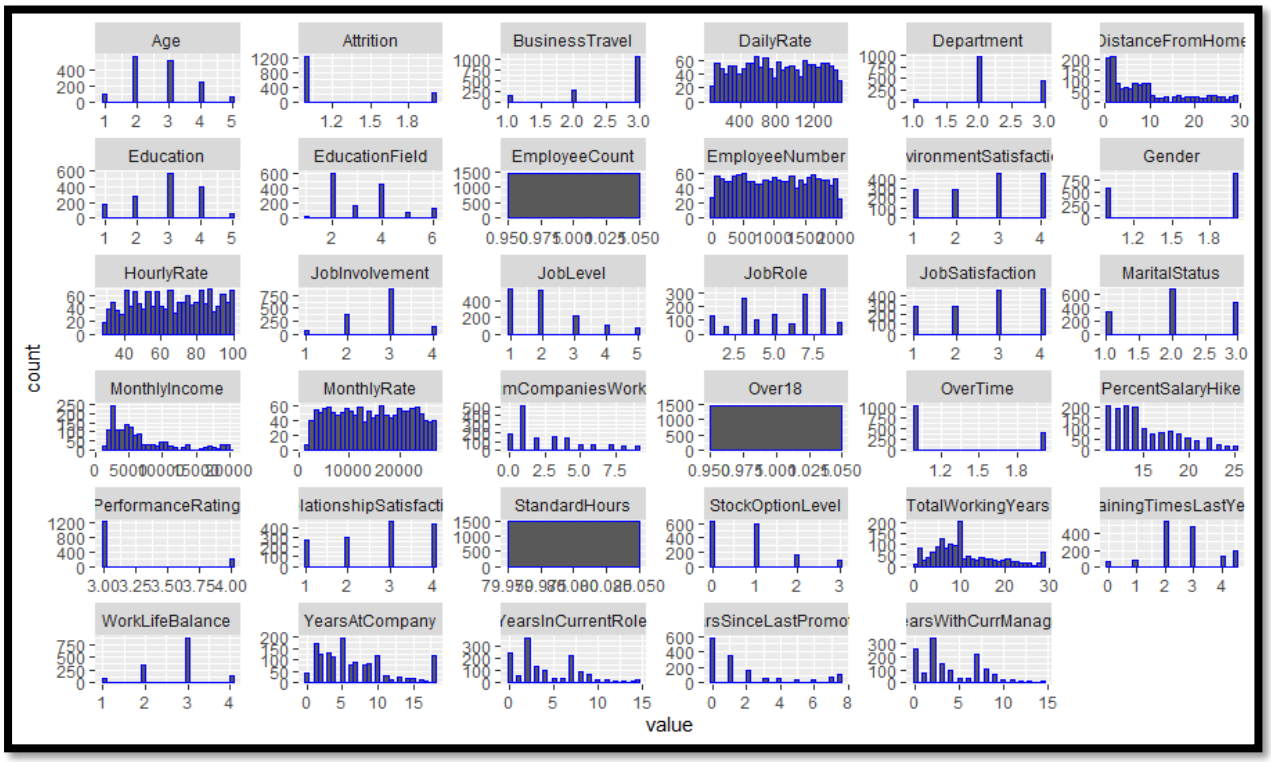
For starting with the Data Visualization we will initially convert all the categorical variables into numerical class for better classification and visualization.

```
#Attrition Variable
Attrition$Attrition = as.factor (Attrition$Attrition)
Attrition$Attrition <- as.numeric (Attrition$Attrition)

#Department Variable
Attrition$Department = as.factor (Attrition$Department)
Attrition$Department <- as.numeric (Attrition$Department)

#EducationField Variable
Attrition$EducationField = as.factor(Attrition$EducationField)
Attrition$EducationField = as.numeric(Attrition$EducationField)

#Over18  Variable
Attrition$Over18 = as.factor (Attrition$Over18)
Attrition$Over18 = as.numeric (Attrition$Over18)

#Business Travel Variable
Attrition$BusinessTravel = as.factor (Attrition$BusinessTravel)
Attrition$BusinessTravel = as.numeric (Attrition$BusinessTravel)

#Gender Variable
Attrition$Gender = as.factor (Attrition$Gender)
Attrition$Gender = as.numeric (Attrition$Gender)

#JobRole Variable
Attrition$JobRole = as.factor (Attrition$JobRole)
Attrition$JobRole = as.numeric (Attrition$JobRole)

#Marital Variable
Attrition$MaritalStatus = as.factor (Attrition$MaritalStatus)
Attrition$MaritalStatus = as.numeric (Attrition$MaritalStatus)

#Over Time Variable
Attrition$OverTime = as.factor (Attrition$OverTime)
Attrition$OverTime = as.numeric (Attrition$OverTime)
#Department Variable
Attrition$Department = as.factor (Attrition$Department)
Attrition$Department = as.numeric (Attrition$Department)
```

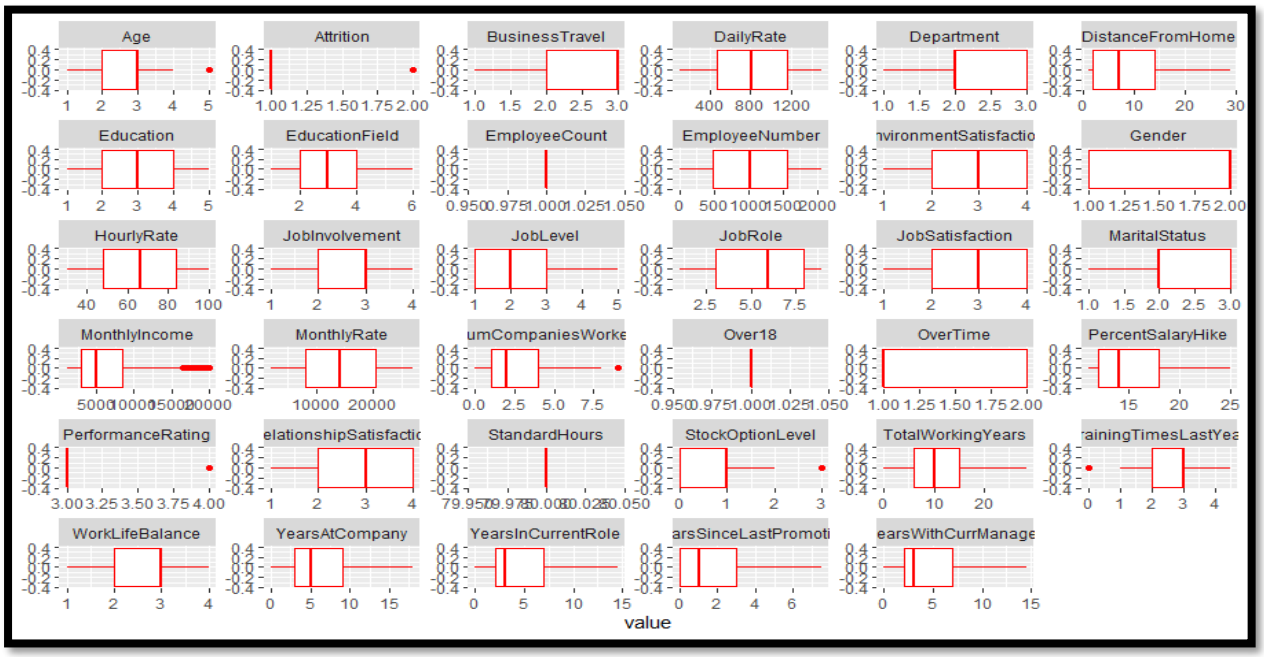
3.2 Visualization of All Variables

Visualization of all the 34 variables to have a complete overview of all the variables in the graphical form.

The Visualization will be done with both Histogram and Boxplot.



Histogram showing overview of all the variables.

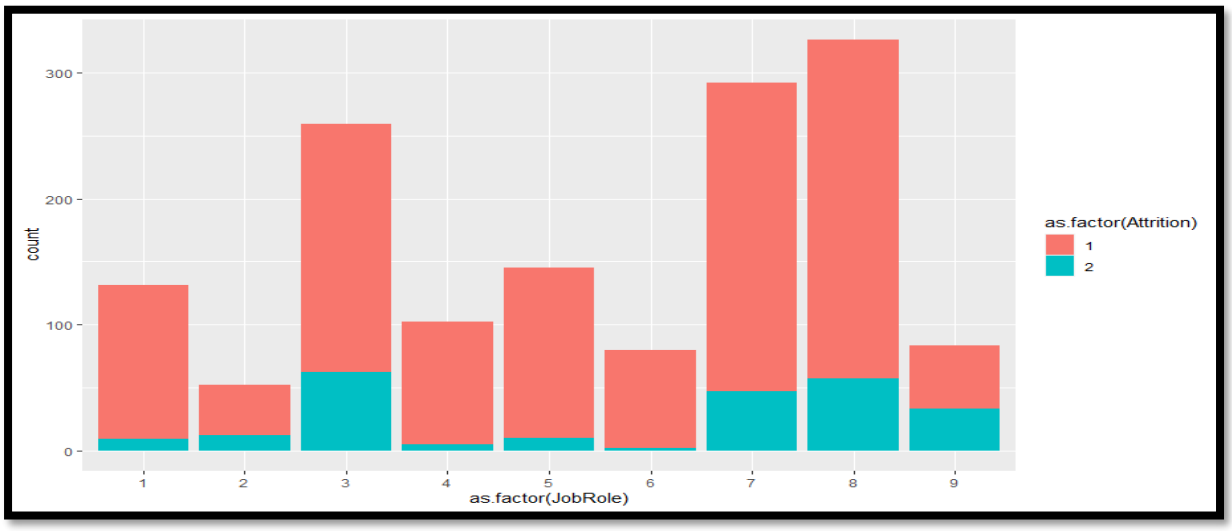


Boxplot showing overview of all the variables.

3.2 Visualization of Selected Variables.

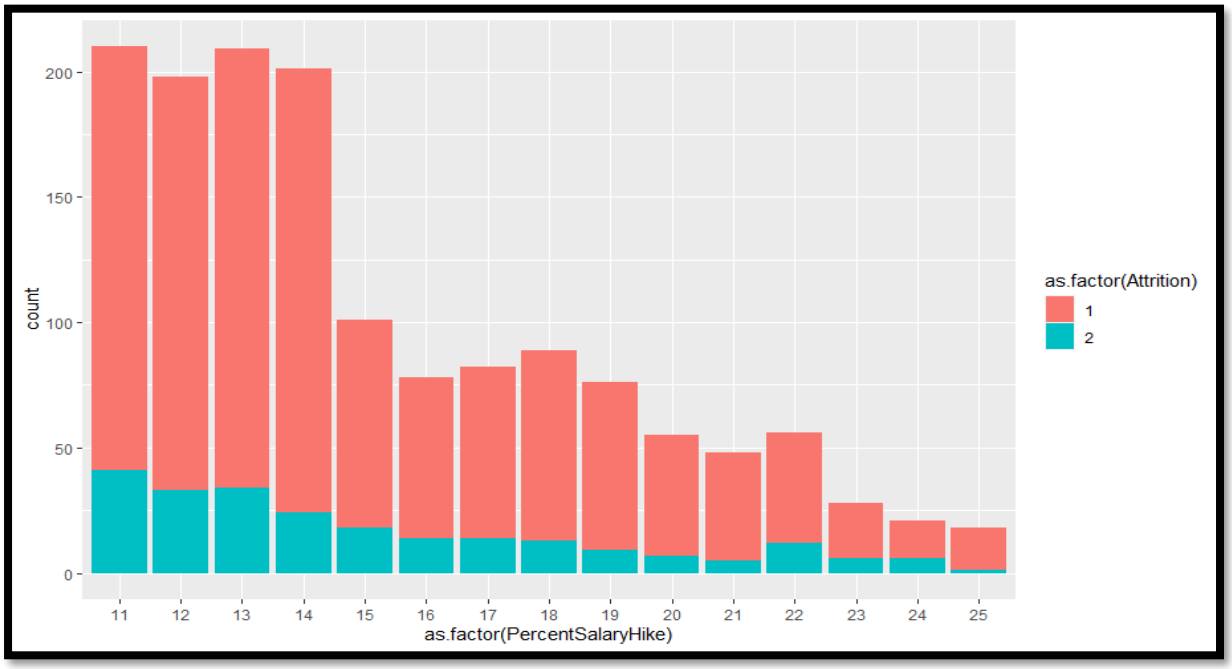
To find the causes of Attrition of the Employees we will visually analyze certain variables which generally considered important in regards to the Attrition process.

- We plot the graph of Job Roles with Attrition as factors.



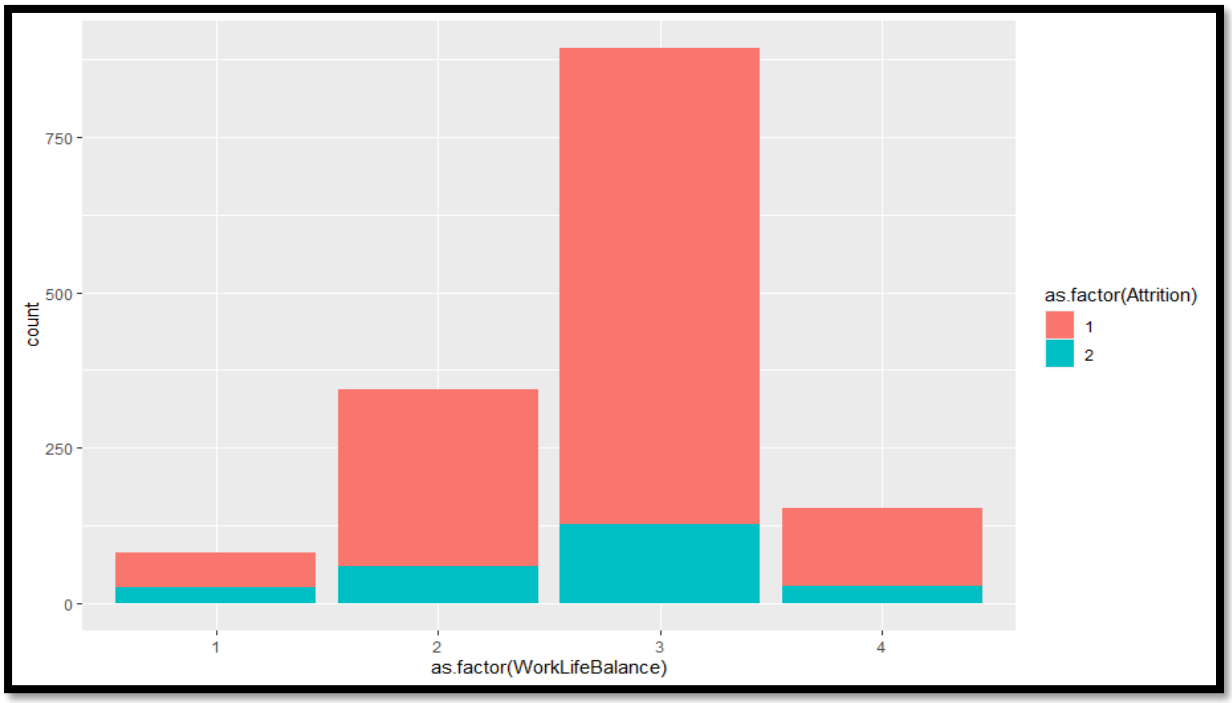
We can visualize that the Highest Attrition has happened for the Job Role – 8 i.e. Sales Executive Job Role and least Attrition has happened for Job Role – 2.

- We plot the graph of the Percentage Salary Hike with Attrition as a factor.



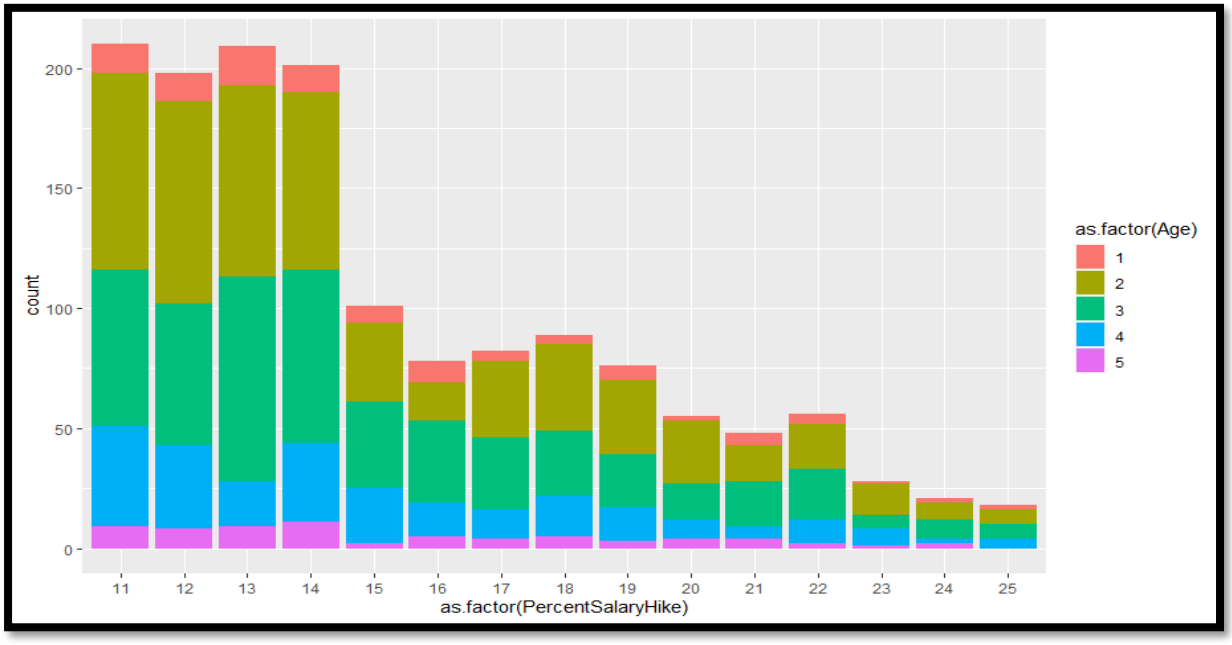
We can easily analyze from the graph that the Highest Attrition rate has happened for the least Percentage of salary hike, i.e. Salary Hike is inversely proportional to Attrition Rate.

- We will plot a graph for Work-Life balance with Attrition rate.



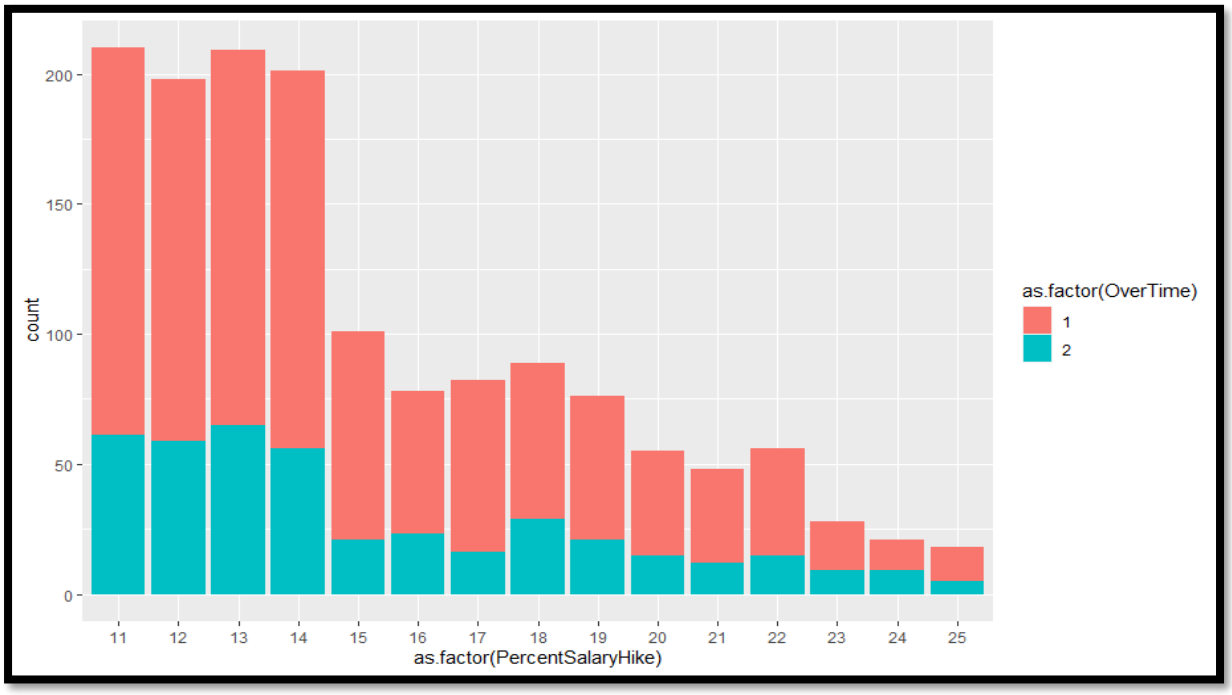
We can see that the Highest Attrition rate is happening with Work-Life Balance Class 3 i.e. having a Better Work-Life Balance.

- We will plot the graph between the Percentage salary Hike and the Age group.



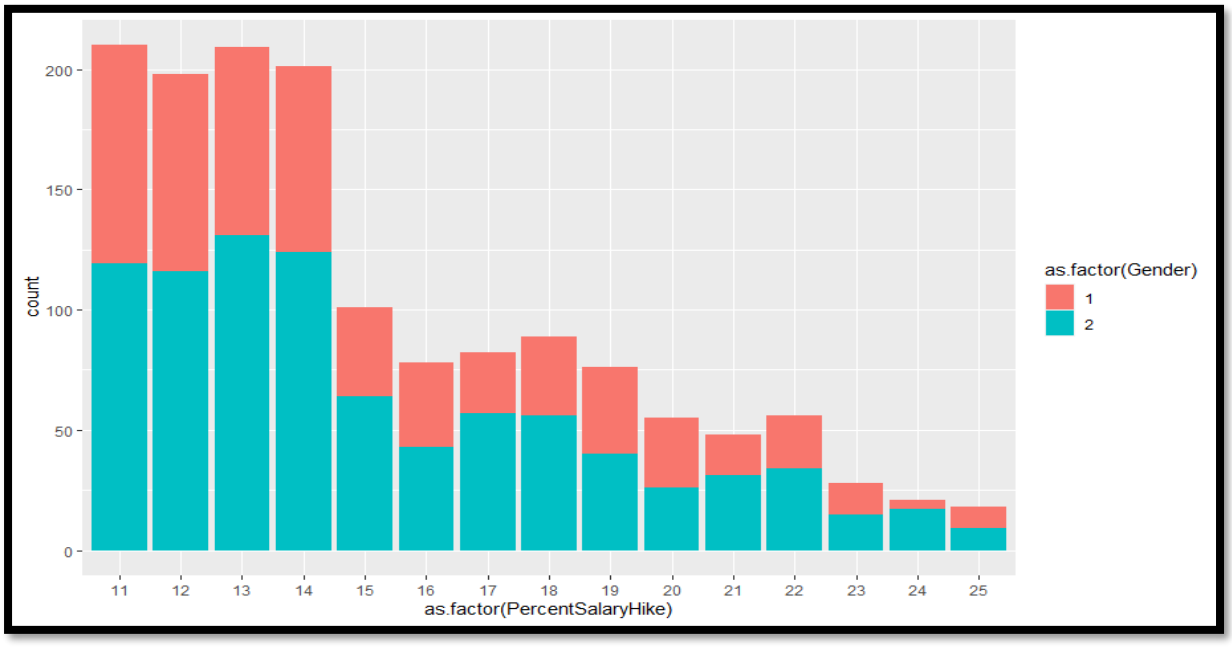
We can analyze that the largest count of Employees with a Salary hike is 11%, i.e. a larger part of the Employees are getting the least Salary Hike. And the Age group getting the least Salary hike is 25 to 34 and the highest salary hike is for the Age group is 35 to 44.

- The graph between Percentage Salary Hike and overtime done.

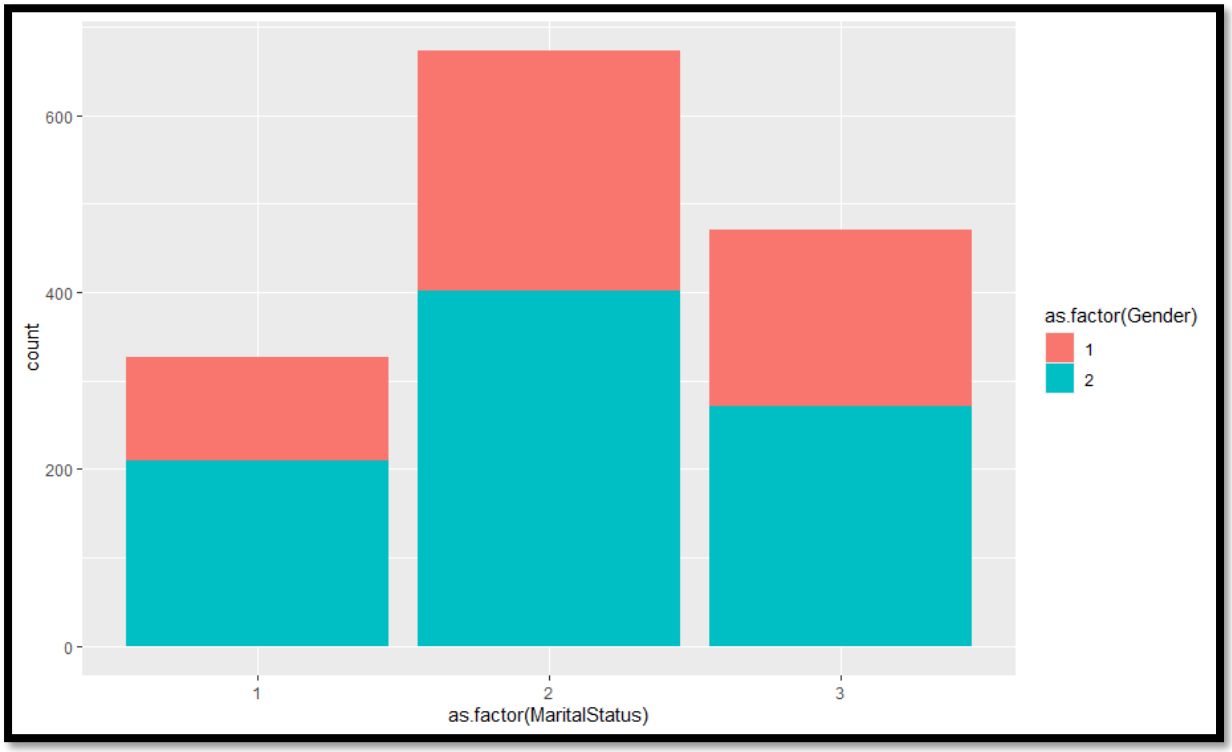


We can analyze from the graph that the least Salary Hike is given to more no. of Employees doing Salary Hike.

- The graph of the Percentage of Salary Hike and Gender.

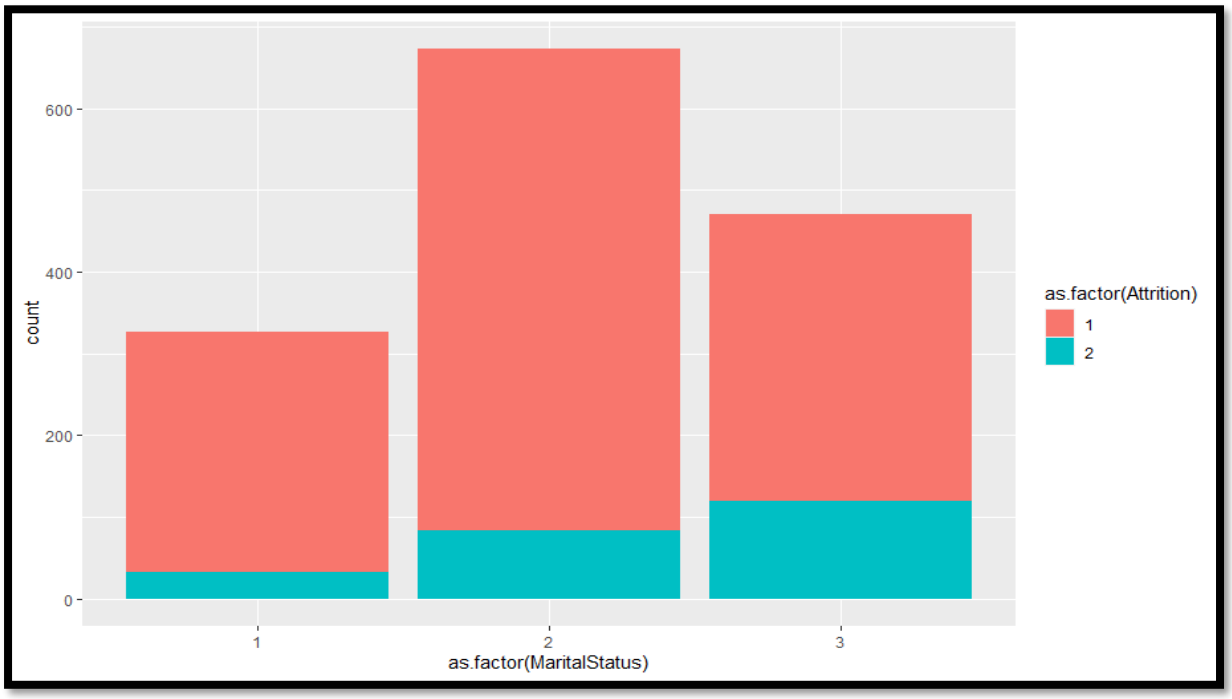


- Graph of Marital status with Gender as Factor.



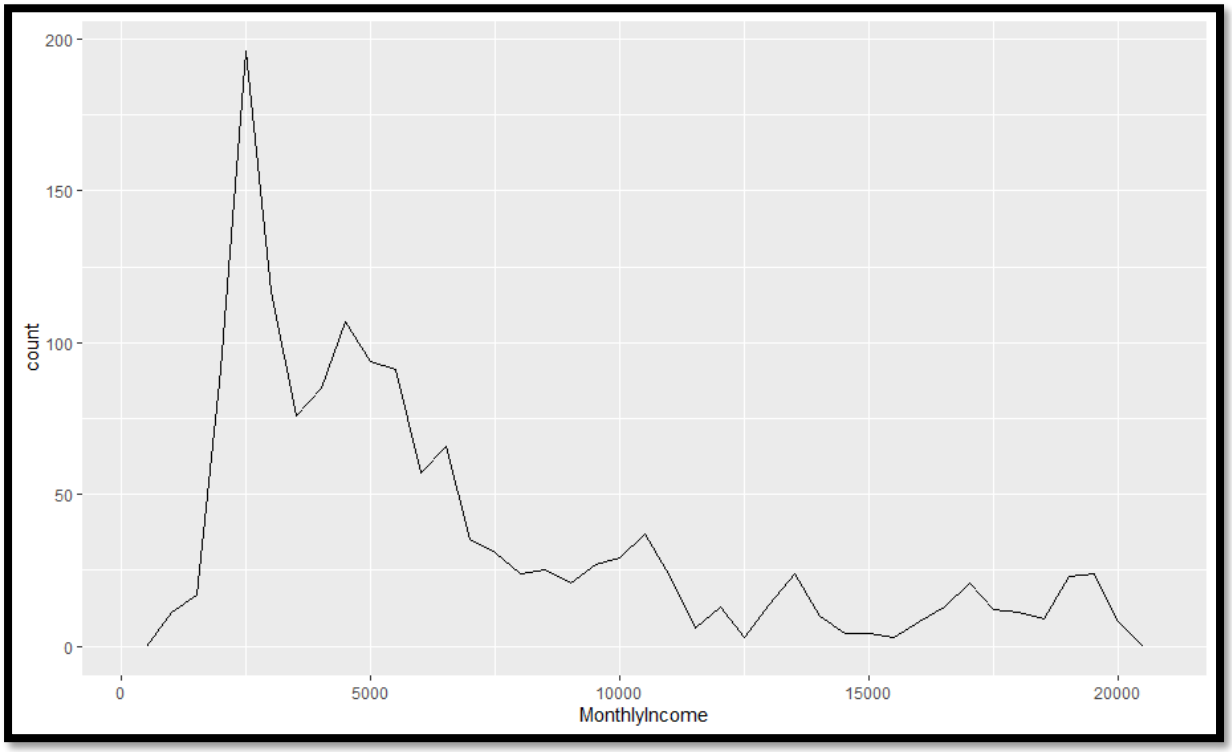
A larger count of employees in the data set are married, followed by single and divorced.

- The graph between Marital Status and Employee Attrition



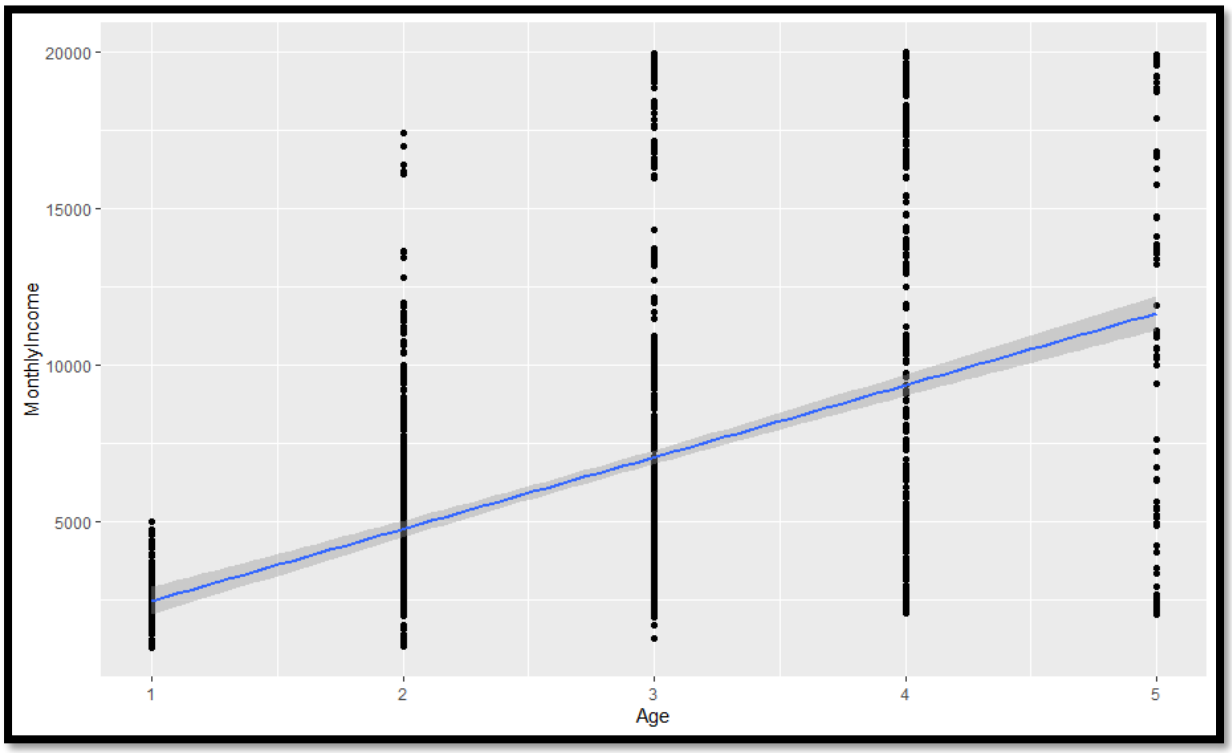
It can be concluded that the Highest Attrition Rate is with Employees with Marital Rate – Married, and the employees with Marital Status- Divorced have the least Attrition rate.

- The graph between Employee count and Monthly Income.



We can see that the maximum no. of employees receive a lower salary range of 2000 to 3000.

- The graph between Age and Monthly Income.



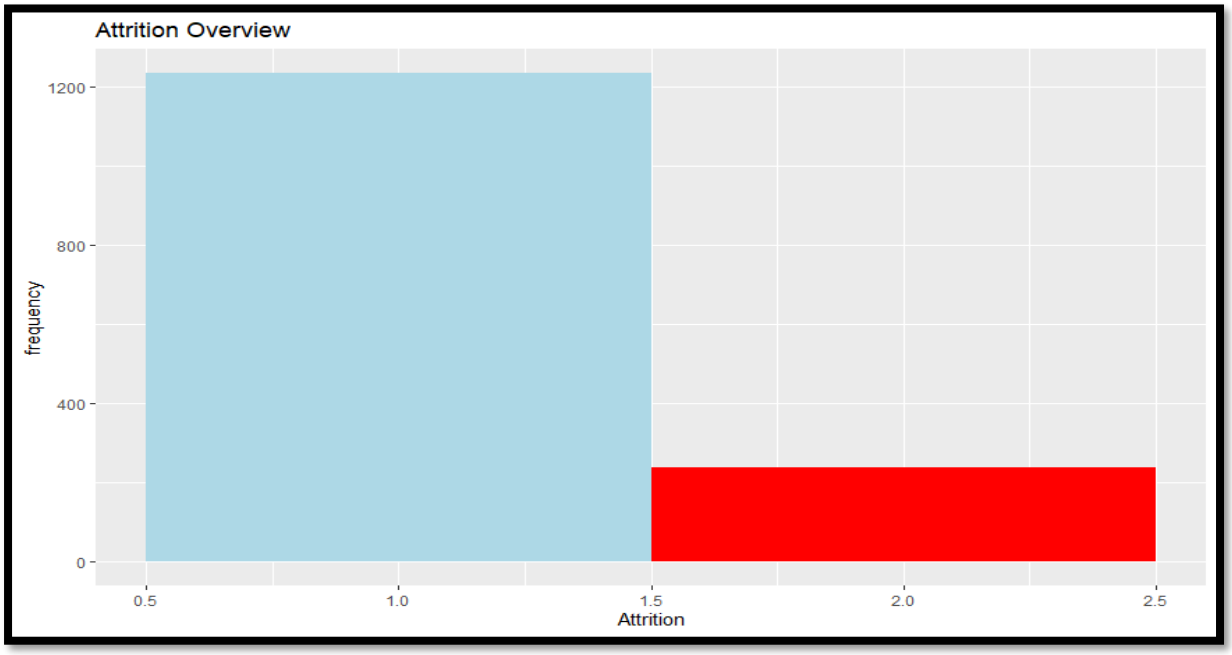
We can easily analyze that the average monthly income increases with the increase in Age.

CHAPTER 4: FEATURE ENGINEERING

Feature Engineering is the process to extract features from raw data via data mining techniques. These features can be used to improve the performance of machine learning algorithms. Feature engineering can be considered as applied machine learning itself.

4.1 DATA BALANCING

Checking the Target variable i.e. Attrition Variable.

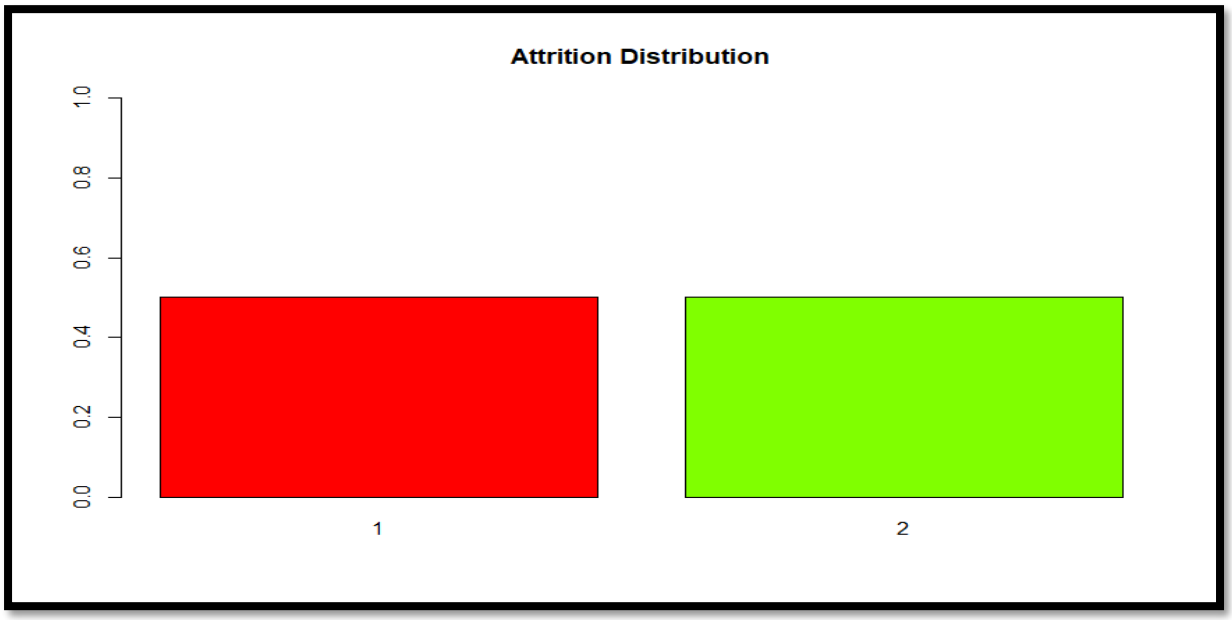


We observe from the target variable that data is highly imbalanced i.e. there is a huge difference between the count of 1's & 0's. Thus also depicting that we have to used imbalance treatment methods later to reduce chances of bias.
We can observe out of total 1470 observation, 1233 belongs to 1 and 237 belong to 237, there is a huge imbalance.
DATA BALANCING can be done by the following methods:

- **Random under Sampling**: Random Under sampling aims to balance class distribution by randomly eliminating majority class examples.
- **Random Sampling:** Over-Sampling increases the number of instances in the minority class by randomly replicating them in order.
- **Cluster-based Over Sampling:** K-means clustering algorithm is applied to minority and majority class instances. This is to identify clusters in the dataset. Subsequently, each cluster is oversampled such that all clusters of the same class have an equal number of instances and all classes have the same size.
- **Synthetic Minority Over-sampling Technique:** A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset.

We did **ROSE (Random over Sampling)** over the complete data to get balanced data.

```
# Applying ROSE library and performing the Over Balancing.

library(ROSE)
Attrition = ovun.sample(Attrition~., data = Attrition, method = "over", N = 2466)$data
table(Attrition$Attrition)
prop.table(table(Attrition$Attrition))
barplot(prop.table(table(Attrition$Attrition)),
        col = rainbow(4),
        ylim = c(0, 1.0),
        main = "Attrition Distribution")
```



We can observe from the graph that there are equal no. of observations for both the classes of the target variables, i.e. 1233 observations each for both the data points.

4.2 DATA SCALING

Data Scaling can be done in 2 ways mentioned below:

- **NORMALIZATION** - Rescaling data to have values between 0 and 1. This is usually called feature scaling.
- **STANDARDIZATION -** Standardization transforms data to have a mean of zero and a standard deviation of 1. This standardization is called a z-score.

Let's scale the data using the standardization method:

```
#standardizing the Age variable
mean(Attrition$Age)
sd(Attrition$Age)
summary(Attrition$Age)
# Standardizing AGE
Attrition$Age = (Attrition$Age-mean(Attrition$Age))/sd(Attrition$Age)
# Checking MEAN & SD after standardization
mean(Attrition$Age)
sd(Attrition$Age)

#standardizing the DISTANCE GROM HOME variable
mean(Attrition$DistanceFromHome)
sd(Attrition$DistanceFromHome)
summary(Attrition$DistanceFromHome)
Attrition$DistanceFromHome = (Attrition$DistanceFromHome-
mean(Attrition$DistanceFromHome))/sd(Attrition$DistanceFromHome)
Attrition$DistanceFromHome
mean(Attrition$DistanceFromHome)
sd (Attrition$DistanceFromHome)

#standardizing the MONTHLY INCOME variable.
mean(Attrition$MonthlyIncome)
sd(Attrition$MonthlyIncome)
Attrition$MonthlyIncome = (Attrition$MonthlyIncome-
mean(Attrition$MonthlyIncome))/sd(Attrition$MonthlyIncome)
Attrition$MonthlyIncome
mean(Attrition$MonthlyIncome)
sd(Attrition$MonthlyIncome)

#standardizing the TOTAL WORKING YEARS variable
mean(Attrition$TotalWorkingYears)
sd(Attrition$TotalWorkingYears)
Attrition$TotalWorkingYears = (Attrition$TotalWorkingYears-
mean(Attrition$TotalWorkingYears))/sd(Attrition$TotalWorkingYears)
Attrition$TotalWorkingYears
mean(Attrition$TotalWorkingYears)
sd(Attrition$TotalWorkingYears)
```

#standardizing the PERCENTAGE SALARY HIKE variable

mean(Attrition$PercentSalaryHike)
sd(Attrition$PercentSalaryHike)
Attrition$PercentSalaryHike = (Attrition$PercentSalaryHike-
mean(Attrition$PercentSalaryHike))/sd(Attrition$PercentSalaryHike)
Attrition$PercentSalaryHike
mean(Attrition$PercentSalaryHike)
sd(Attrition$PercentSalaryHike)

#standardizing the YEARS AT COMPANY variable

mean(Attrition$YearsAtCompany)
sd(Attrition$YearsAtCompany)
Attrition$YearsAtCompany = (Attrition$YearsAtCompany-
mean(Attrition$YearsAtCompany))/sd(Attrition$YearsAtCompany)
Attrition$YearsAtCompany
mean(Attrition$YearsAtCompany)
sd(Attrition$YearsAtCompany)

#standardizing the YEARS IN CURRENT ROLE variable

mean(Attrition$YearsInCurrentRole)
sd(Attrition$YearsInCurrentRole)
Attrition$YearsInCurrentRole = (Attrition$YearsInCurrentRole-
mean(Attrition$YearsInCurrentRole))/sd(Attrition$Years.InCurrentRole)
Attrition$YearsInCurrentRole
mean(Attrition$YearsInCurrentRole)
sd(Attrition$YearsInCurrentRole)

 # standardizing the YEARS WITH CURRENT MANAGER

mean(Attrition$Years.With.Curr.Manager)
sd(Attrition$Years.With.Curr.Manager)
Attrition$Years.With.Curr.Manager = (Attrition$Years.With.Curr.Manager-
mean(Attrition$Years.With.Curr.Manager))/sd(Attrition$Years.With.Curr.Mana
ger)
Attrition$Years.With.Curr.Manager
mean(Attrition$Years.With.Curr.Manager)
sd(Attrition$Years.With.Curr.Manager)

4.3 FEATURE SELECTION

Feature Selection of the variable is done using various methods that are:

5 T-Test for Continuous Variable
6 Chi-Square Test.
7 BORUTO
8 Random Forest

4.3.1 T-Test for Continuous Variable

t.test(Attrition$Age~Attrition$Attrition) #significant difference

t.test(Attrition$Daily.Rate~Attrition$Attrition) #significant difference

t.test(Attrition$Distance.From.Home~Attrition$Attrition) #significant difference

t.test(Attrition$Employee.Number~Attrition$Attrition) #not significant

t.test(Attrition$Environment.Satisfaction~Attrition$Attrition) #significant

t.test (Attrition$Hourly..Rate~Attrition$Attrition) # not significant

t.test(Attrition$Job.Satisfaction~Attrition$Attrition) #significant

t.test (Attrition$Monthly.Income~Attrition$Attrition) #significant

t.test (Attrition$Monthly.Rate~Attrition$Attrition) #not significant

t.test(Attrition$Num.Companies.Worked~Attrition$Attrition) #not significant

t.test(Attrition$Over.Time~Attrition$Attrition) #significant

t.test(Attrition$Percent.Salary.Hike~Attrition$Attrition) #not significant

t.test(Attrition$Performance.Rating~Attrition$Attrition) #not significant

t.test(Attrition$Relationship.Satisfaction~Attrition$Attrition) #not significant

t.test(Attrition$Total.Working.Years~Attrition$Attrition) #significant

t.test(Attrition$Training.Times.Last.Year~Attrition$Attrition) # significant

t.test(Attrition$Work.Life.Balance~Attrition$Attrition) # significant

t.test(Attrition$Years.At.Company~Attrition$Attrition) #significant

t.test(Attrition$Years.In.Current.Role~Attrition$Attrition) #significant

```
t.test(Attrition$Years.Since.Last.Promotion~Attrition$Attrition) #not significant

t.test(Attrition$Years.With.Curr.Manager~Attrition$Attrition) #significant

t.test(Attrition$Job.Involvement~Attrition$Attrition) # Significant

t.test(Attrition$Job.Level~Attrition$Attrition) # significant

t.test(Attrition$Stock.Option.Level~Attrition$Attrition) # significant
```

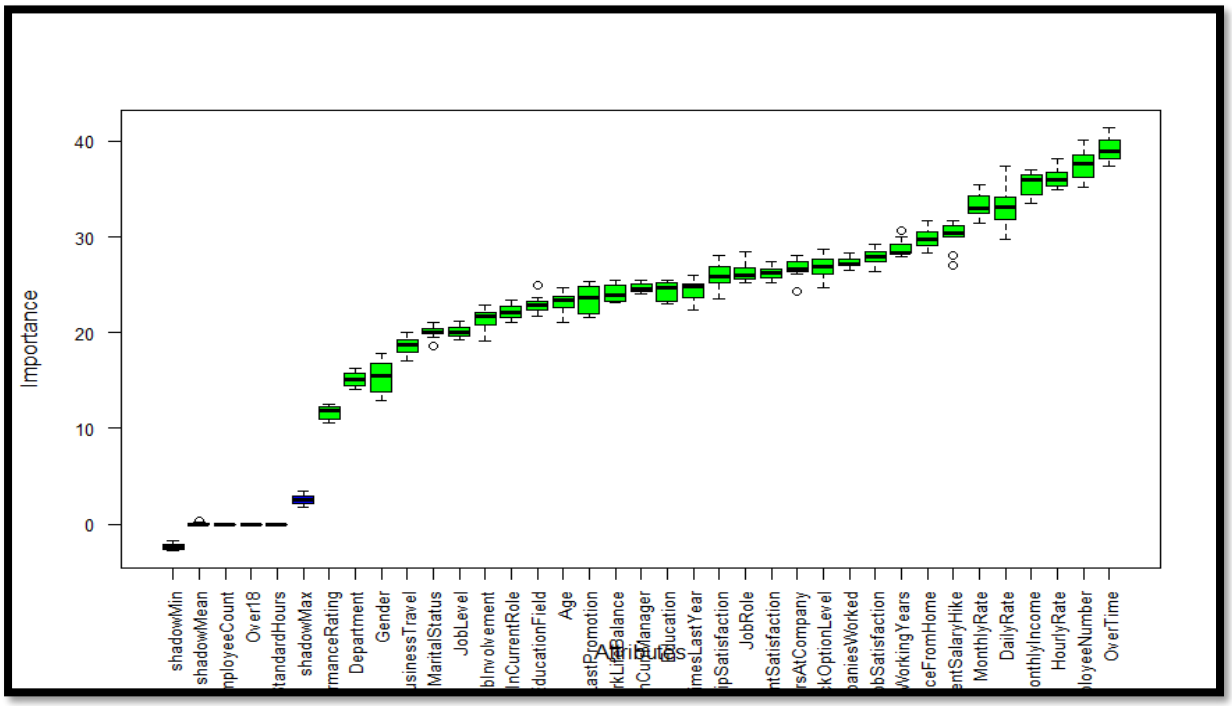
4.3.2 CHI – SQUARE TEST

```
# Create a data frame from the main data -
Att.data = data.frame(Attrition$Attrition, Attrition$BusinessTravel)
# Create a table -
Att.data = table (Attrition$Attrition, Attrition$BusinessTravel)
print ((Att.data))
# Chi-Square
print(chisq.test(Att.data))
# p-value = 5.609e-06 = 0.000005609
# since p value < 0.05 - strong correlation, significant

Att.data = data.frame(Attrition$Attrition, Attrition$Department)
Att.data = table(Attrition$Attrition, Attrition$Department)
print((Att.data))
print(chisq.test(Att.data))
# p value = 0.004526, significant

Att.data = data.frame(Attrition$Attrition, Attrition$Education)
Att.data = table(Attrition$Attrition, Attrition$Education)
print((Att.data))
print(chisq.test(Att.data))
# p value = 0.5, not significant

Att.data = data.frame(Attrition$Attrition, Attrition$EducationField)
Att.data = table(Attrition$Attrition, Attrition$EducationField)
print((Att.data))
print(chisq.test(Att.data))
# p value = 0.006774, significant

Att.data = data.frame (Attrition$Attrition, Attrition$Gender)
Att.data = table(Attrition$Attrition, Attrition$Gender)
print((Att.data))
print(chisq.test(Att.data))# p value = 0.2906, not significant
Att.data = data.frame(Attrition$Attrition, Attrition$JobInvolvement)
Att.data = table(Attrition$Attrition, Attrition$JobInvolvement)
```

```
print((Att.data))
print(chisq.test(Att.data))
# p value = 2.863e-06, significant

Att.data = data.frame(Attrition$Attrition, Attrition$JobLevel)
Att.data = table(Attrition$Attrition, Attrition$JobLevel)
print((Att.data))
print(chisq.test(Att.data))
# p value = 6.635e-15, significant

Att.data = data.frame(Attrition$Attrition, Attrition$JobRole)
Att.data = table(Attrition$Attrition, Attrition$JobRole)
print((Att.data))
print(chisq.test(Att.data))
# p value = 2.752e-15, significant

Att.data = data.frame(Attrition$Attrition, Attrition$MaritalStatus)
Att.data = table(Attrition$Attrition, Attrition$MaritalStatus)
print((Att.data))
print(chisq.test(Att.data))
# p value = 9.456e-11, significant

Att.data = data.frame(Attrition$Attrition, Attrition$Over18)
Att.data = table(Attrition$Attrition, Attrition$Over18)
print((Att.data))
print(chisq.test(Att.data))
# p value = < 2.2e-16, significant

Att.data = data.frame(Attrition$Attrition, Attrition$StandardHours)
Att.data = table(Attrition$Attrition, Attrition$StandardHours)
print((Att.data))
print(chisq.test(Att.data))
# p value = <2.2e-16, significant

Att.data = data.frame(Attrition$Attrition, Attrition$StockOptionLevel)
Att.data = table(Attrition$Attrition, Attrition$StockOptionLevel)
print((Att.data))
print(chisq.test(Att.data))
# p value = 4.379e-13, significant
```

4.3.3 BORUTA

```
# Feature Selection using BORUTA #
library (Boruta)
library(mlbench)
library(caret)
library(randomForest)

set.seed(111)
boruta = Boruta(Attrition ~, data = Attrition, doTrace = 2, maxRuns = 500)
print(boruta)
boruta_sig = getSelectedAttributes(boruta, withTentative = FALSE)
boruta_sig
plot(boruta, las = 2, cex.axis = 0.8)
# blue - shadow attributes, red - confirmed unimportant, green - confirmed important, yellow -tentative.

attStats(boruta)

# For Tentative attribute #
tent = TentativeRoughFix(boruta)
Print (tent)
```

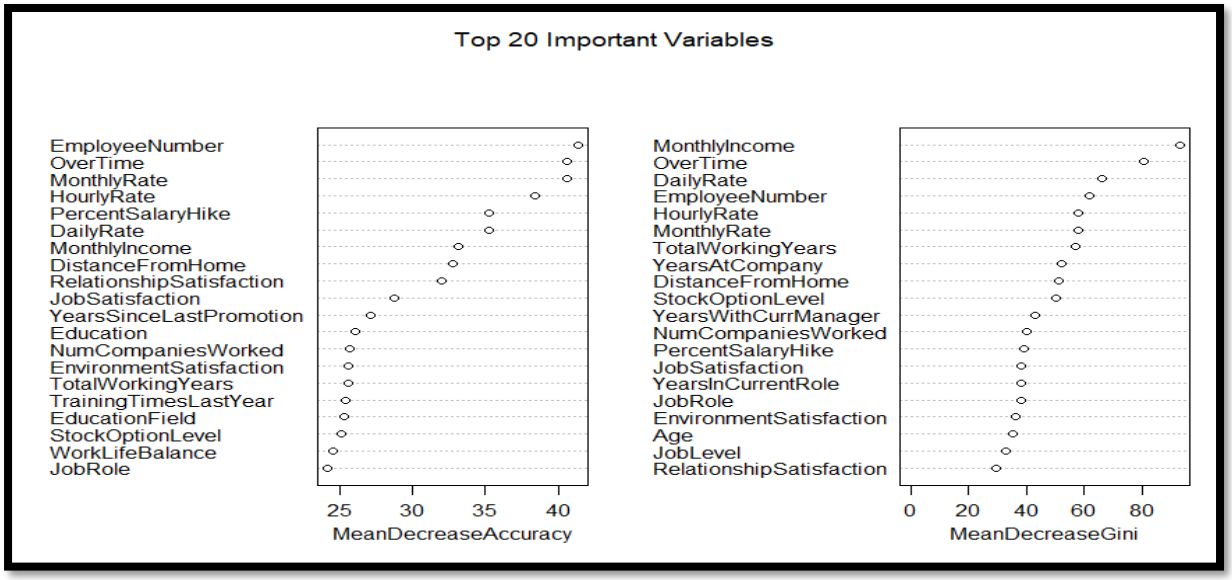


In **Boruta**, the features do not compete with each other, instead they compete with a randomized version of themselves called the shadow feature.

A feature is useful/selected only if it is capable of doing better than the best-randomized feature.
Area of refusal – **RED**, Area of irresolution – **BLUE**, Area of acceptance – **GREEN**.

4.3.4 RANDOM FOREST

```
# Convert Attrition into Factor/Categorical variable #
Attrition$Attrition = as.factor(Attrition$Attrition)

# Random Forest on TRAIN data #
rf <- randomForest(Attrition ~., data = Attrition)
print(rf)

# Putting ntree value from error rate & mtry value into initial model created #
rf <- randomForest(Attrition ~., data = Attrition,
ntree = 300, mtry = 5,
importance = TRUE,
proximity = TRUE)
print(rf)

# Important Variables #
varImpPlot(rf)

# shows for all the variables
varImpPlot(rf,
sort = TRUE,
n.var = 20,
main = "Top 20 Important Variables")
varUsed(rf)
```



4.4 SUBSETTING SELECTED FEATURES

The Features which were found significant enough from the above Feature selection methods were now subsetted into new data sets for the Model Selection and testing.

```
Attrition1 = Attrition[
,c(1,3,6,11,14,15,16,17,18,19,22,23,24,28,29,30,31,32,33,35,2)]
Attrition1
names(Attrition1)
table(Attrition1$Attrition)
```

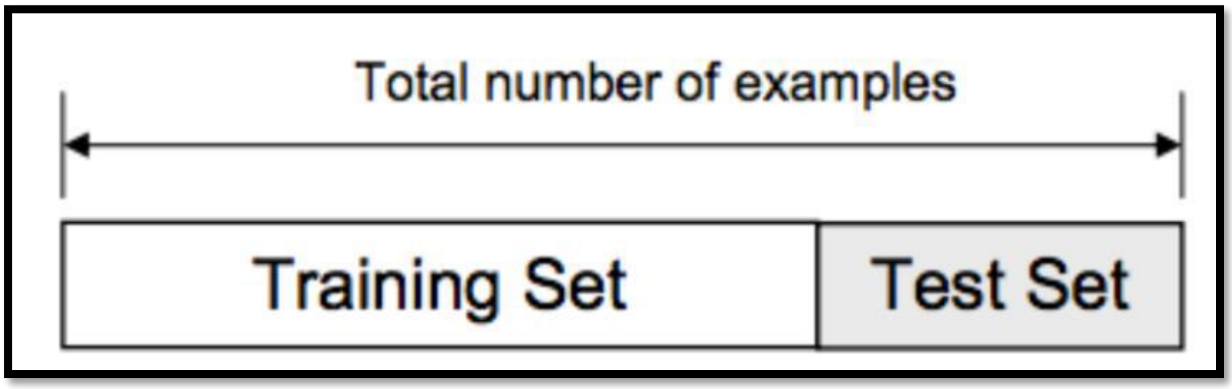
The Variables that were considered important after the feature engineering and Feature Selection are:

- **AGE**
- **BUSINESS TRAVEL**
- **DISTANCE FROM HOME**
- **ENVIRONMENT SATISFACTION**
- **JOB INVOLVEMENT**
- **JOB LEVEL**
- **JOB ROLE**
- **JOB SATISFACTION**
- **MARITAL STATUS**
- **MONTHLY INCOME**
- **OVER18**
- **OVERTIME**
- **PERCENT SALARY HIKE**
- **STOCK OPTION LEVEL**
- **TOTAL WORKING YEARS**
- **TRAINING TIME LAST YEAR**
- **WORK-LIFE BALANCE**
- **YEARS AT COMPANY**
- **YEARS IN CURRENT ROLE**
- **YEARS WITH CURRENT MANAGER**
- **ATTRITION (TARGET VARIABLE**)

CHAPTER 5: MODEL BUILDING

5.1 SPLITTING THE DATA TEST-TRAIN

Before building the Models the complete data should be split now into training and test data. The training set contains a known output and the model learns on this data to be generalized to other data later on. We have the test dataset (or subset) to test our model's prediction on this subset. Further, if we have a large dataset, we can divide the training dataset into Sub Train & validation sets and 1st check accuracy on validation and then on test data.



We will split the data into 70% Train Data and 30% Test Data of the complete Subsetted data.

```
library(caret)

# Split into TEST & TRAIN (70-30)
validation_index = createDataPartition(Attrition1$Attrition, p = 0.7, list = FALSE)
validation = Attrition1[-validation_index,]
Attrition1 = Attrition1 [validation_index,]
```

Boxplot and Whisker Plot for all the variables.

5.2 HOW TO EVALUATE MODELS

To determine the best models out of the models created by the Model building Algorithm we consider the Confusion Matrix, Recall, Precision, F1 Score, and Accuracy.

CONFUSION MATRIX

It shows the actual and predicted labels from a classification problem.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

True positives: data points labeled as positive that are positive.
False positives: data points labeled as positive that are negative.
True negatives: data points labeled as negative that are negative.
False negatives: data points labeled as negative that are positive.

RECALL

The ability of a classification model to identify all relevant instances.

$$recall = \frac{true\ positives}{true\ positives\ + false\ negatives} :$$

PRECISION

The ability of a classification model to return only relevant instances.

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

F1 SCORE

A single metric that combines recall and precision using the harmonic mean.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

ACCURACY

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations.

$$Accuracy = \frac{TrueNegatives + TruePositive}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

5.3 BUILDING MODELS

For this assignment, we will be focusing on **Logistic regression**, **Random Forest**, **Support Vector Machine** (SVM), **KNN.**

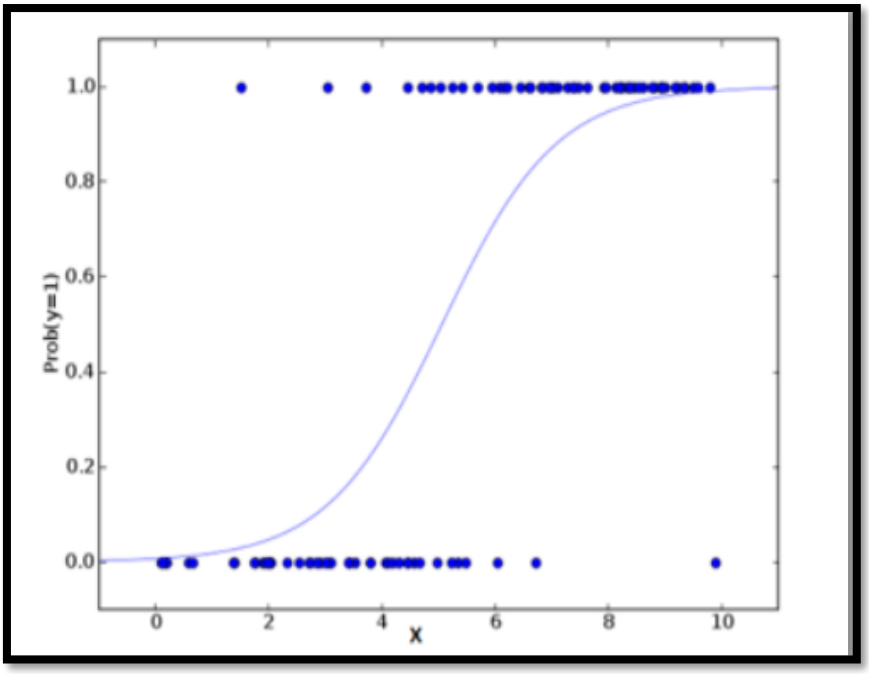
5.3.1 LOGISTIC REGRESSION

Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary/categorical outcomes, we use dummy variables.

The equation for Logistic Regression:

$$\log\left(\frac{p}{1-p}\right) = \beta_o + \beta(Age)$$

Ideal curve For Logistic Regression:

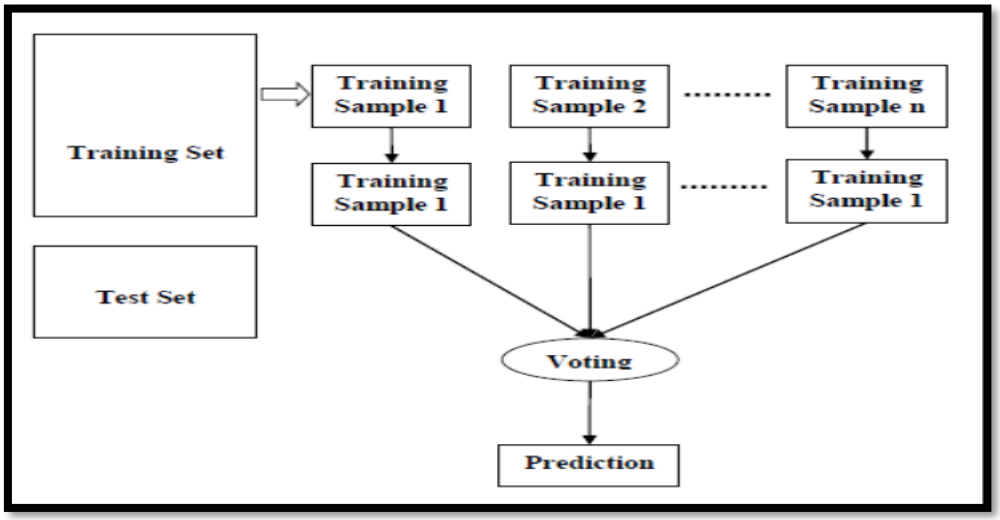


 # Building the Logistic Regression model in the Train data
fit.log = train(Attrition~., data = Attrition1, method = "glm", metric = metric, trControl = control)
 # Estimating the skill of Logistic regression Model
predictions = predict(fit.log, validation)
confusionMatrix(predictions, validation$Attrition)

```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 268 106
         2 101 263

               Accuracy : 0.7195
                 95% CI : (0.6856, 0.7517)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.439

 Mcnemar's Test P-Value : 0.781

            Sensitivity : 0.7263
            Specificity : 0.7127
         Pos Pred Value : 0.7166
         Neg Pred Value : 0.7225
             Prevalence : 0.5000
         Detection Rate : 0.3631
   Detection Prevalence : 0.5068
      Balanced Accuracy : 0.7195

       'Positive' Class : 1
```

5.3.2 RANDOM FOREST

Random forest is a supervised learning algorithm that is used for both classifications as well as regression. However, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means a more robust forest. Similarly, a random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution employing voting.



 # Building the RANDOM FOREST model in the Train data

```
fit.rf = train(Attrition~., data = Attrition1, method = "rf", metric = metric, trControl = control)
# Estimate the skill of RandomForest on the validation data
predictions = predict(fit.rf, validation)
confusionMatrix(predictions, validation$Attrition)
```

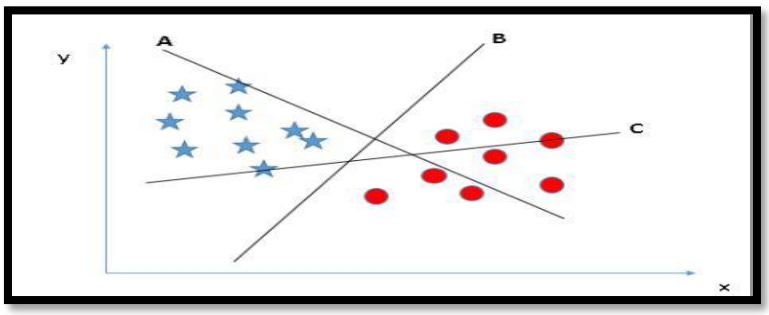
```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 351   7
         2  18 362

               Accuracy : 0.9661
                 95% CI : (0.9504, 0.978)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9322

 Mcnemar's Test P-Value : 0.0455

            Sensitivity : 0.9512
            Specificity : 0.9810
         Pos Pred Value : 0.9804
         Neg Pred Value : 0.9526
             Prevalence : 0.5000
         Detection Rate : 0.4756
   Detection Prevalence : 0.4851
      Balanced Accuracy : 0.9661

       'Positive' Class : 1
```

5.3.3 SUPPORT VECTOR MACHINE

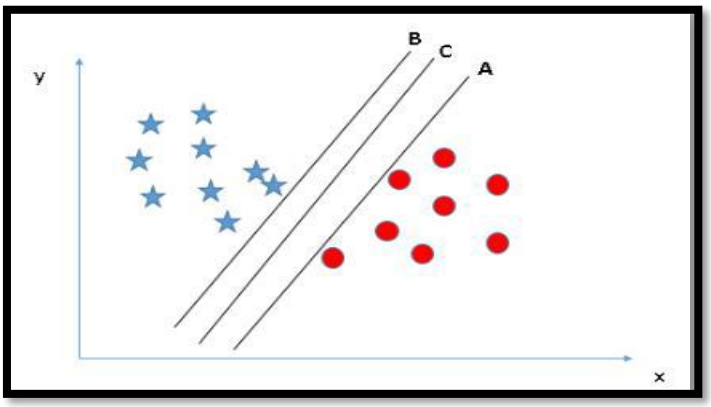
It is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is many features you have) with the value of each feature being the value of a coordinate.

Working principle :

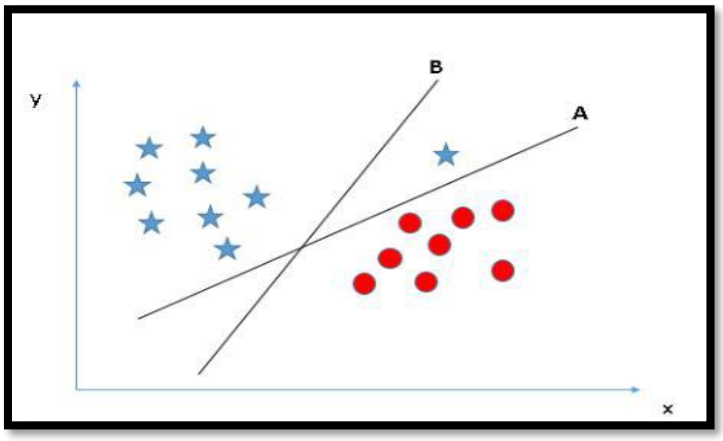
Identify the right hyper-plane (Scenario-1)- Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job.



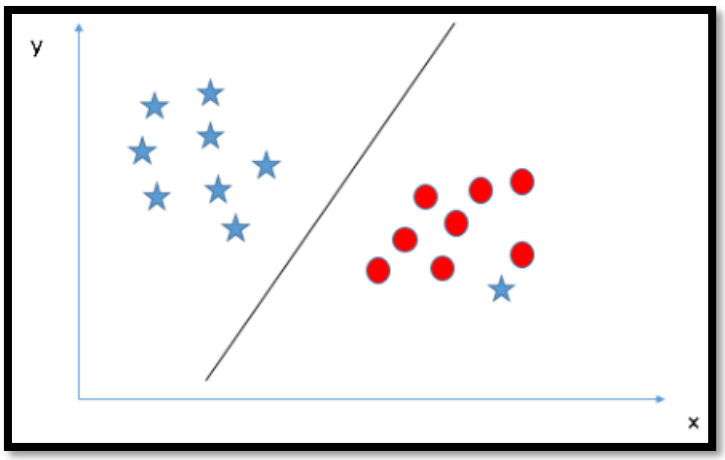
Identify the right hyper-plane (Scenario-2)- Maximizing the distances between the nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called a Margin.



Identify the right hyper-plane (Scenario-3)- SVM selects the hyper-plane which classifies the classes accurately before maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is A.



Can we classify two classes (Scenario-4)- The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.



```
# Building the SVM model in the Train data
fit.svm = train(Attrition~., data = Attrition1, method = "svmRadial", metric = metric, trControl = control)
# Estimate the skill of SVM on the validation data
predictions = predict(fit.svm, validation)
confusionMatrix(predictions, validation$Attrition)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 296  67
         2  73 302

               Accuracy : 0.8103
                 95% CI : (0.7801, 0.838)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.6206

 Mcnemar's Test P-Value : 0.6726

            Sensitivity : 0.8022
            Specificity : 0.8184
         Pos Pred Value : 0.8154
         Neg Pred Value : 0.8053
             Prevalence : 0.5000
         Detection Rate : 0.4011
   Detection Prevalence : 0.4919
      Balanced Accuracy : 0.8103

       'Positive' Class : 1
```

5.3.4 K- NEAREST NEIGHBOUR

In statistics, the k-nearest neighbors' algorithm (k-NN) is a non-parametric method proposed by Thomas Cover used classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

```
# Building the KNN model in the Train data.
fit.knn = train(Attrition~., data = Attrition1, method = "knn", metric = metric, trControl = control)
# Estimate the skill of KNN on the validation data
predictions = predict(fit.knn, validation)
confusionMatrix(predictions, validation$Attrition)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   1   2
         1 240  49
         2 129 320

               Accuracy : 0.7588
                 95% CI : (0.7263, 0.7893)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5176

 Mcnemar's Test P-Value : 3.194e-09

            Sensitivity : 0.6504
            Specificity : 0.8672
         Pos Pred Value : 0.8304
         Neg Pred Value : 0.7127
             Prevalence : 0.5000
         Detection Rate : 0.3252
   Detection Prevalence : 0.3916
      Balanced Accuracy : 0.7588

       'Positive' Class : 1
```

5.4 MODEL SUMMARY

MODEL	ACCURACY	SENSITIVITY	SPECIFICITY	KAPPA VALUE
LOGISTIC REGRESSION	0.7195	0.7263	0.7127	0.439
RANDOM FOREST	0.9661	0.9512	0.9810	0.9322
SVM	0.8103	0.8022	0.8184	0.6206
K NEAREST NEIGHBOUR	0.7575	0.6477	0.8672	0.5149

CHAPTER 6: MODEL SELECTION AND RECOMMENDATION

From the above table, we can say that the best Model to predict that if the employee will leave the company or stay with the company is RANDOM FOREST
The model followed by the SVM model.

- Accuracy – 96.61%
- Specificity – 98.10%
- Sensitivity – 95.12%
- Kappa Value – 93.22%

CHAPTER 7: CONCLUSION

Employee retention is a basic issue as organizations seek ability in a tight economy. The expenses of representative turnover are progressively high — as much as 2.5 occasions a worker's compensation relying upon the job.

So, what can you do to keep your employees happy and avoid job-hopping?

- First of all, enlist the opportune individuals. Individuals who meet the association's desires and whose desires you can satisfy.
- Helping employees achieve their short-term and long-term goals is one of the most crucial employee retention strategies.
- You can make a work environment where representatives aren't reluctant to communicate their sentiments. All in all, a working environment where they can unreservedly scrutinize their chiefs and voice their interests.
- An "open entryway strategy" is a compelling method to set up a culture of open correspondence. It shows that you're generally accessible to tune in to their sentiments and concerns.
- 57% of representatives feel less gainful and separated because of work pressure. Pinnacle Watson says focusing on your representatives can be unfavorable. Thus, urging colleagues to share the remaining task at hand and get some much-needed rest can be compelling.
- Organizations have incredible amounts of employee data available — why not use it to identify who’s most likely to leave, why, and then take steps to prevent that, says Dave Weisbeck, CSO at workforce analytics software firm Visier. While on the surface, an employee’s departure may seem obvious or to fit a pattern, but using AI and advanced analytics can help pinpoint underlying factors that contribute to attrition, ones that might not be as obvious as you thought.
- The response to every one of these lies in the previously mentioned representative maintenance techniques. The methodologies can be utilized as an agenda to investigate the territories you need to create.

So, it's truly an ideal opportunity to venture up your worker maintenance game!