Group7_Final Project_64018

Contents

Data Preparation

```
getwd()
```

```
## [1] "C:/Users/Mukht/OneDrive/Desktop/Kent State University/College of Business Admin-Bus. Analytics I
```

```
setwd("C:\\Users\\Mukht\\OneDrive\\Desktop\\Kent State University\\College of Business Admin-Bus. Analy
```

```
FinalProjectx<-read.csv("GroupsevenQMM.csv")
str(FinalProjectx)
```

```
## 'data.frame':    312 obs. of  6 variables:
##  $ ï..InvestFacilitation: num  2.8 4.2 4.8 5 4.2 5 5 4.8 3.4 3 ...
##  $ FDIInflow            : int  7 12 12 2 1 13 12 2 4 5 ...
##  $ ZInvestFacilitation  : num  -2.107 0.517 1.642 2.017 0.517 ...
##  $ InvestProm           : num  0.18 1.13 1.09 1.13 1.02 1.11 1.21 1.15 1.21 0.13 ...
##  $ ZInvestProm          : num  0.287 1.267 1.223 1.258 1.153 ...
##  $ InvProm_X_InvestFacil: num  0.51 4.77 5.24 5.63 4.3 5.56 6.03 5.51 4.12 0.38 ...
```

```
head(FinalProjectx)
```

```
##   ï..InvestFacilitation FDIInflow ZInvestFacilitation InvestProm ZInvestProm
## 1                   2.8         7            -2.10736       0.18     0.28696
## 2                   4.2        12             0.51747       1.13     1.26689
## 3                   4.8        12             1.64239       1.09     1.22251
```

```
## 4                   5.0         2             2.01737       1.13     1.25787
## 5                   4.2         1             0.51747       1.02     1.15287
## 6                   5.0        13             2.01737       1.11     1.24445
##   InvProm_X_InvestFacil
## 1                  0.51
## 2                  4.77
## 3                  5.24
## 4                  5.63
## 5                  4.30
## 6                  5.56
```

```
library(class)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(lattice)
library(ggplot2)
library(ISLR)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.4     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(e1071)
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.1.2
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(esquisse)
```

```
## Warning: package 'esquisse' was built under R version 4.1.2
```

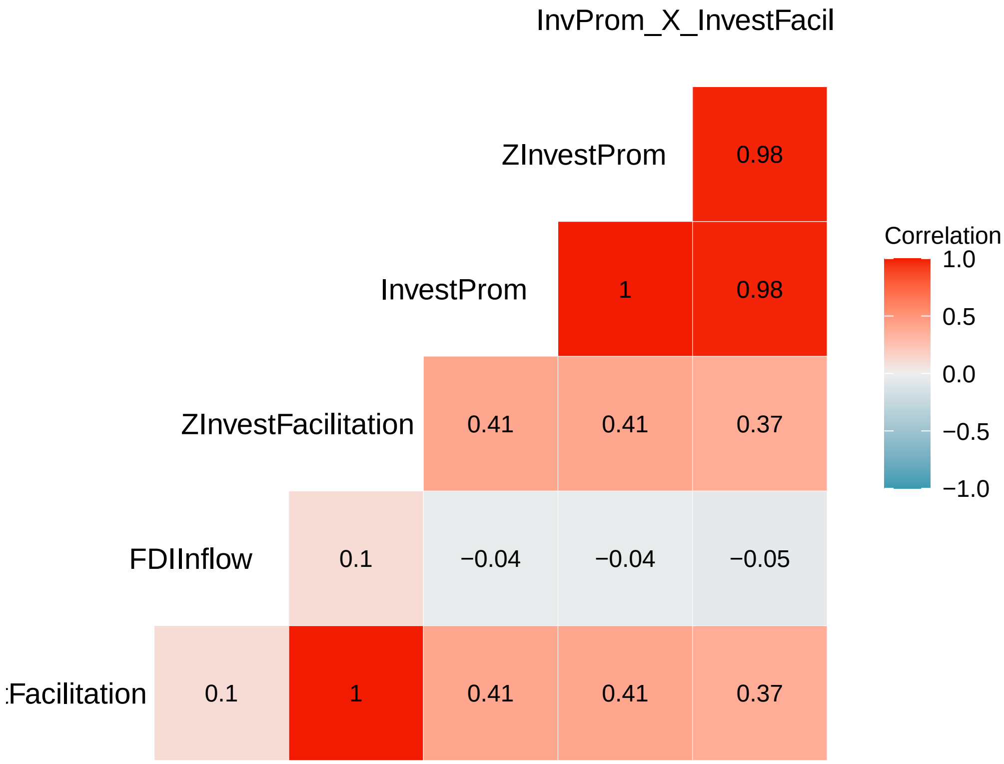
#Plot correlation headmap

```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.1.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggcorr(FinalProjectx, label = TRUE, palette = "RdBu", name = "Correlation", hjust = 0.75, label_size =3
```




```
FinalProjectx_range_normalized<-preProcess(FinalProjectx, method = "range")
FinalProjectx_normalized = predict(FinalProjectx_range_normalized, FinalProjectx)
summary(FinalProjectx_normalized)
```

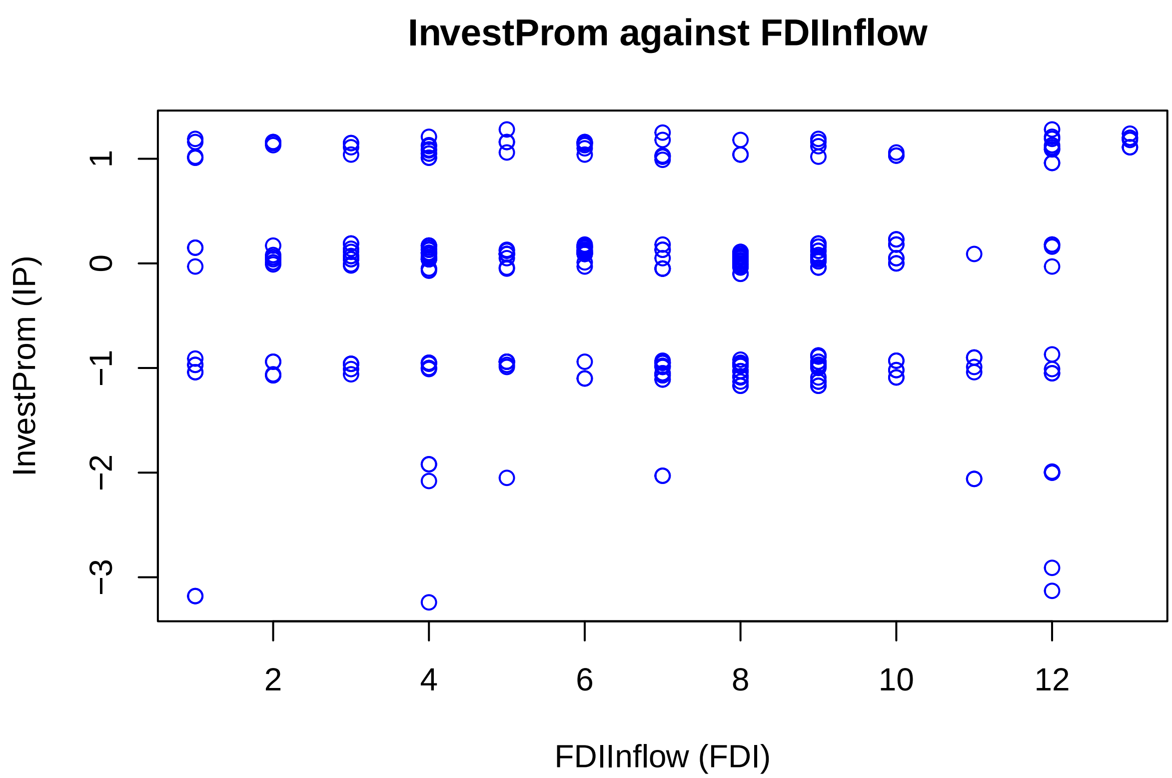
```
##  ï..InvestFacilitation    FDIInflow       ZInvestFacilitation   InvestProm
##  Min.   :0.0000         Min.   :0.0000   Min.   :0.0000       Min.   :0.0000
##  1st Qu.:0.6500         1st Qu.:0.2500   1st Qu.:0.6500       1st Qu.:0.5044
##  Median :0.7000         Median :0.5000   Median :0.7000       Median :0.7257
##  Mean   :0.7183         Mean   :0.4725   Mean   :0.7183       Mean   :0.6821
##  3rd Qu.:0.8000         3rd Qu.:0.6667   3rd Qu.:0.8000       3rd Qu.:0.7566
##  Max.   :1.0000         Max.   :1.0000   Max.   :1.0000       Max.   :1.0000
##   ZInvestProm      InvProm_X_InvestFacil
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.5048   1st Qu.:0.4970
##  Median :0.7255   Median :0.6852
##  Mean   :0.6818   Mean   :0.6556
##  3rd Qu.:0.7556   3rd Qu.:0.7120
##  Max.   :1.0000   Max.   :1.0000
```

```
#Linear Regression
# Creates a linear model for Investment Promotion vs FDI Inflow and displays a plot of the points
Modela = lm(FDIInflow ~  FinalProjectx$InvestProm, data = FinalProjectx)
summary(Modela)
```

```
##
```

```
## Call:
## lm(formula = FDIInflow ~ FinalProjectx$InvestProm, data = FinalProjectx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0378 -2.6358  0.2294  2.2412  6.5002
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                6.6508     0.1781  37.338   <2e-16 ***
## FinalProjectx$InvestProm  -0.1217     0.1894  -0.643    0.521
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.102 on 310 degrees of freedom
## Multiple R-squared:  0.00133,    Adjusted R-squared:  -0.001891
## F-statistic: 0.413 on 1 and 310 DF,  p-value: 0.5209
```

```
plot(FinalProjectx$FDIInflow, FinalProjectx$InvestProm, xlab = "FDIInflow (FDI)", ylab = "InvestProm (I
```

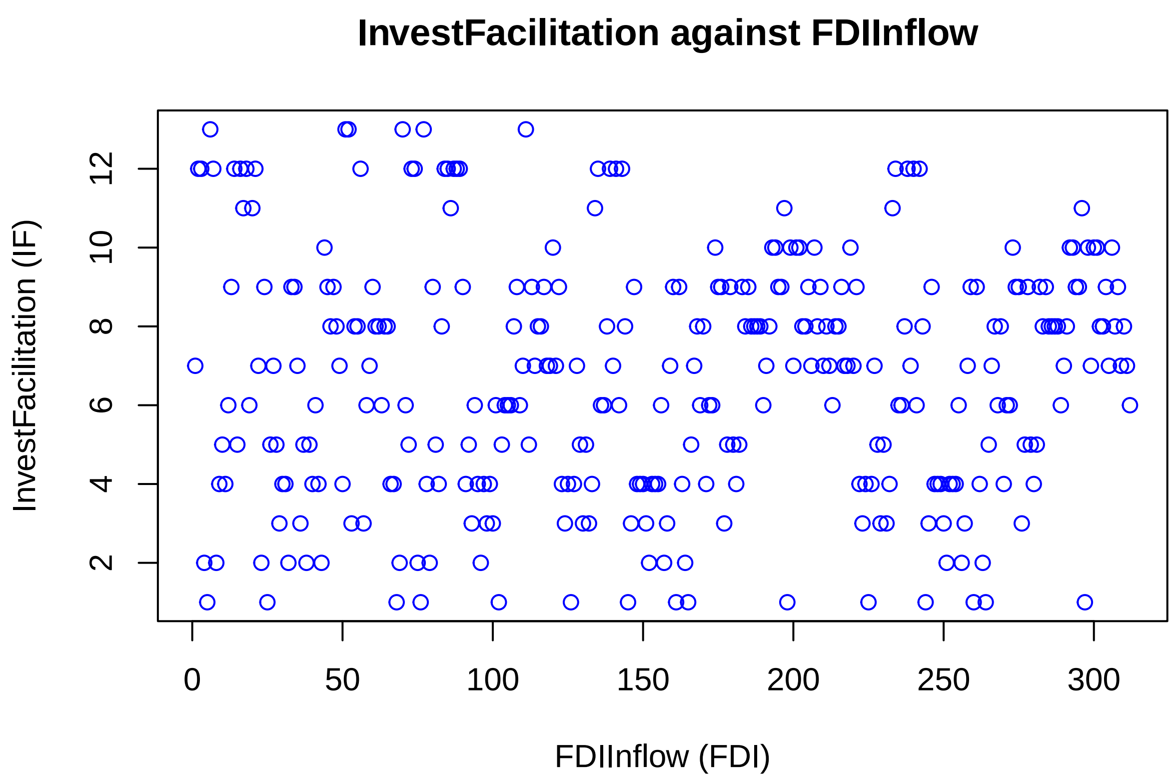


*** #From this linear model we can see that Investment Promotion Services results in a model that the P Value is insignificant (0.52), and that it accounts for 0.13% of the variation in Foreign Direct Investment Inflow.

```
# Creates a linear model for Investment facilitation vs FDI Inflow and displays a plot of the points
Modelb = lm(FDIInflow ~ FinalProjectx$ï..InvestFacilitation, data = FinalProjectx)
summary(Modelb)
```

```
##
## Call:
## lm(formula = FDIInflow ~ FinalProjectx$ï..InvestFacilitation,
##     data = FinalProjectx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3836 -2.4969  0.3764  2.2497  6.8831
##
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                           4.2170     1.3448   3.136  0.00188 **
## FinalProjectx$ï..InvestFacilitation   0.6333     0.3443   1.840  0.06680 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.087 on 310 degrees of freedom
## Multiple R-squared:  0.0108, Adjusted R-squared:  0.007607
## F-statistic: 3.384 on 1 and 310 DF,  p-value: 0.0668
```

```
plot(FinalProjectx$FDIInflow, FinalProjectx$InvestFacilitationx, xlab = "FDIInflow (FDI)", ylab = "Inves
```

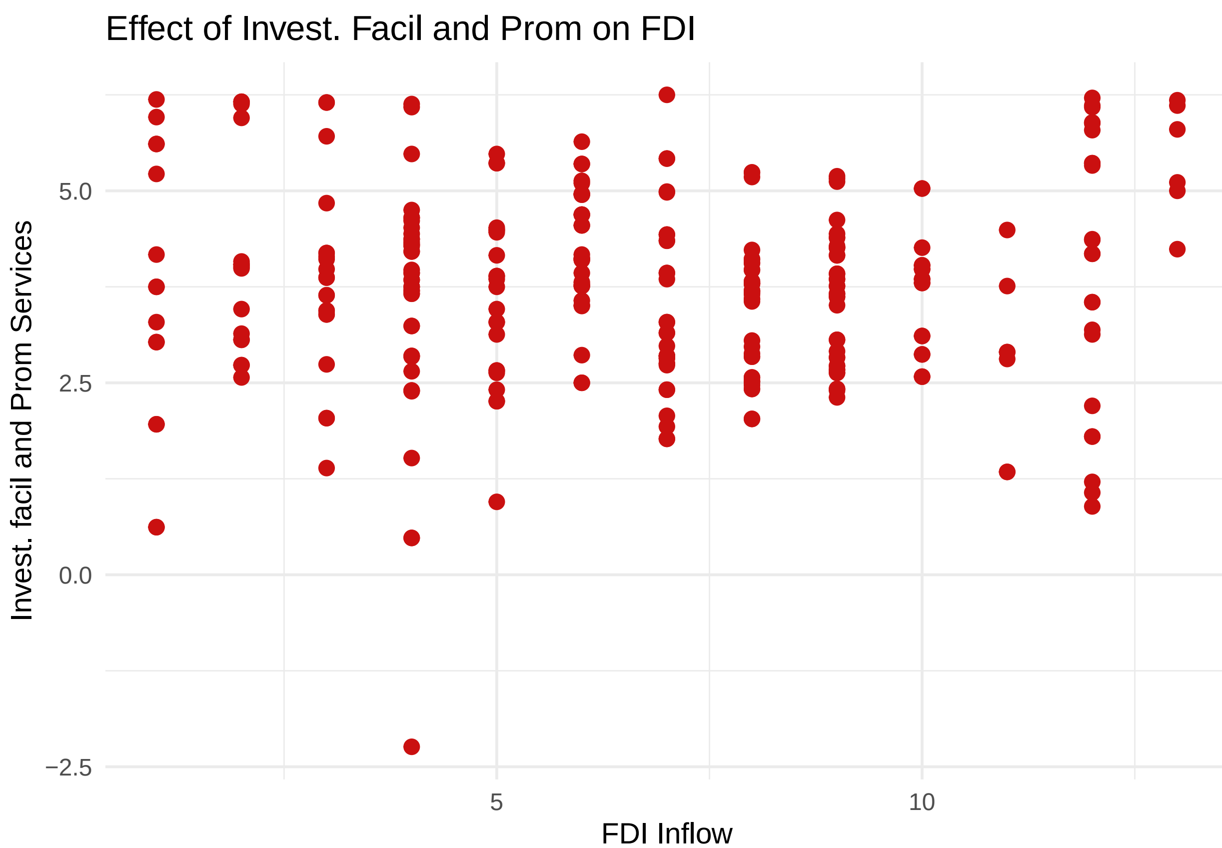


#From this linear model we can see that Investment Facilitation Services results in a model that the P Value is almost significant (0.067), and that it accounts for 1.08% of the variation in Foreign Direct Investment Inflow.

```
# Creates a linear model for a combined IF and IP vs FDI Inflow and displays a plot of the points
Modelc = lm(FDIInflow ~ ï..InvestFacilitation+InvestProm, data = FinalProjectx)
summary(Modelc)
```

```
##
## Call:
## lm(formula = FDIInflow ~ ï..InvestFacilitation + InvestProm,
##     data = FinalProjectx)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.564 -2.408  0.122  2.116  7.532
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             3.2500     1.4822   2.193   0.0291 *
## ï..InvestFacilitation   0.8701     0.3765   2.311   0.0215 *
## InvestProm             -0.3168     0.2062  -1.536   0.1254
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.081 on 309 degrees of freedom
## Multiple R-squared:  0.0183, Adjusted R-squared:  0.01194
## F-statistic:  2.88 on 2 and 309 DF,  p-value: 0.05766
```

```
ggplot(FinalProjectx) +
  aes(x = FDIInflow, y = ï..InvestFacilitation+InvestProm,) + geom_point(shape = "circle", size = 2.25,
```



#When we created a model that contains both Investment Facilitation and Investment Promotion, this time we have a highly significant P value of IF (0.022), and we see that the total model accounts for 1.83% of the variability in FDI Inflow.

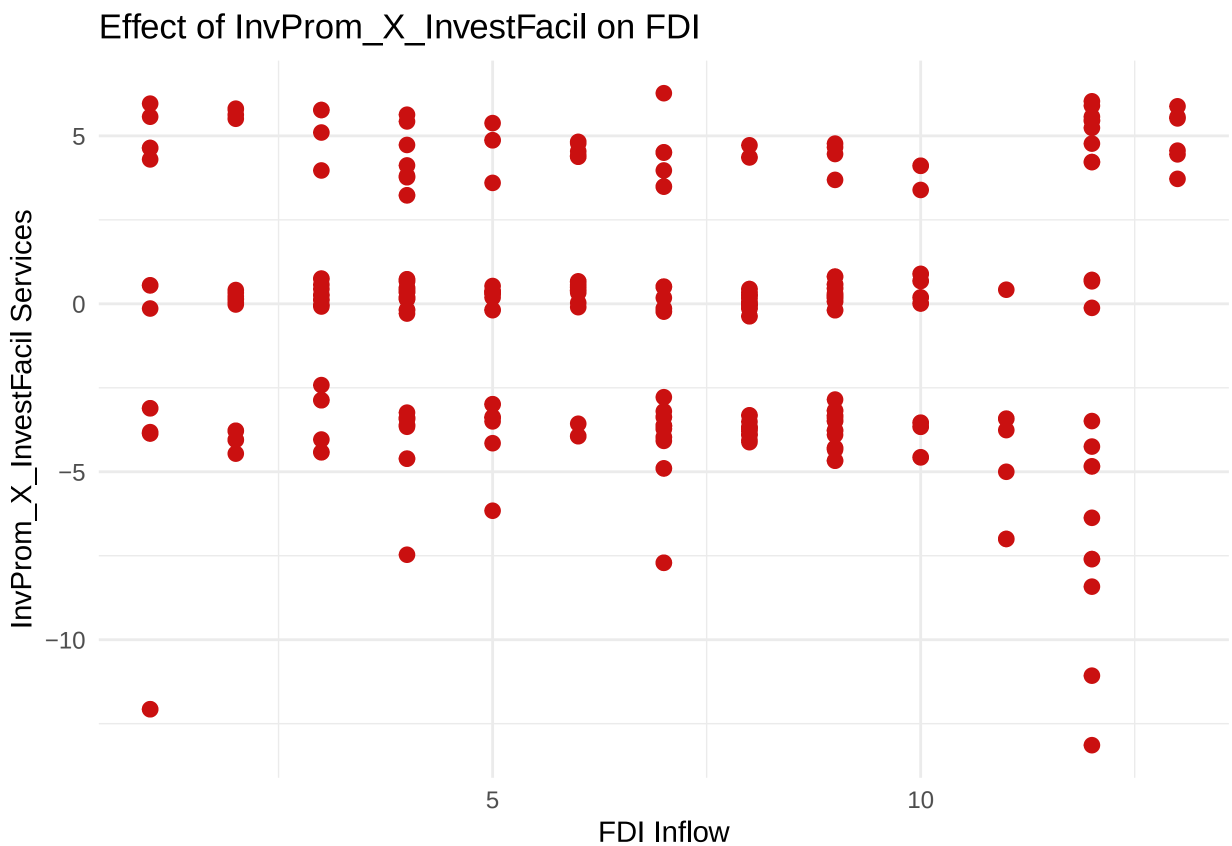
b) Build a model that uses the ZInvestProm, ZInvestFacilitation, and InvProm_X_InvestFacil to predict theFDI Inflow.

```
#A combined moderated effects of Standardized (Z-Score)IP and IF on FDI Inflow
Modeld = lm(FDIInflow ~ ZInvestProm + ZInvestFacilitation + InvProm_X_InvestFacil, data = FinalProjectx)
summary(Modeld)
```

```
##
## Call:
## lm(formula = FDIInflow ~ ZInvestProm + ZInvestFacilitation +
##     InvProm_X_InvestFacil, data = FinalProjectx)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5898 -2.4615  0.0806  2.0880  7.0462
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             6.6270     0.1824  36.341   <2e-16 ***
## ZInvestProm             0.9537     0.9564   0.997   0.3195
## ZInvestFacilitation     0.4200     0.2032   2.067   0.0395 *
```

```
## InvProm_X_InvestFacil  -0.3376     0.2502  -1.349   0.1782
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.076 on 308 degrees of freedom
## Multiple R-squared:  0.02406,    Adjusted R-squared:  0.01455
## F-statistic: 2.531 on 3 and 308 DF,  p-value: 0.05725
```

```
ggplot(FinalProjectx) +
  aes(x = FDIInflow, y = InvProm_X_InvestFacil) + geom_point(shape = "circle", size = 2.25, colour = "#(
```



#Now we created a model that contains the standardized (Z-Score) of both Investment Facilitation and Investment Promotion and Investment FacilitationXInvestment Promotion, we have a highly significant P value of ZIF (0.039). Although the result indicates there was a non-significant interaction effect (p-value = 0.178) of FacilitationXInvestmen, we may still consider meaningful moderation to be present (Hayes 2013, Matthes and Jörg 2020)Johnson-Neyman Plot. It could also be seen that the total model accounts for 2.4 % of the variability in FDI Inflow.

We now use Naive Bayes on Moderated (Interactive) variables to predict Foreign Direct Investment Inflows.

We will use the e1070 package.

```
library(caret)
library(ISLR)
library(e1071)
```

we divide data set into training and test

```
#Divide data into test and train
FinalProjectx_Index_Train<-createDataPartition(FinalProjectx$FDIInflow, p=0.8, list=FALSE)
Train <-FinalProjectx[FinalProjectx_Index_Train,]
Test  <-FinalProjectx[-FinalProjectx_Index_Train,]
```

#Now, run the Naive Bayes clasifier model, and predict FDI status on the test set

```
# Build a naïve Bayes classifier
FinalProjectx_nb_model <-naiveBayes(FDIInflow~ZInvestProm + ZInvestFacilitation + InvProm_X_InvestFacil
FinalProjectx_nb_model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##          1          2          3          4          5          6          7
## 0.03984064 0.05577689 0.07569721 0.12749004 0.07171315 0.08764940 0.11155378
##          8          9         10         11         12         13
## 0.14342629 0.11553785 0.05976096 0.01992032 0.07171315 0.01992032
##
## Conditional probabilities:
##     ZInvestProm
## Y            [,1]       [,2]
##   1  -0.722485000 1.54083497
##   2   0.009213571 0.85629925
##   3   0.056443158 0.79821500
##   4   0.026470313 1.11380510
##   5  -0.113303333 0.97323185
##   6   0.333008636 0.70920845
##   7  -0.246778214 1.02071406
##   8  -0.172886389 0.66633522
##   9  -0.212913448 0.74069381
##   10  0.036948667 0.79322148
##   11 -0.904426000 0.78354630
```

```
##   12 -0.244712778 1.45574687
##   13  1.292660000 0.04561618
##
##     ZInvestFacilitation
## Y            [,1]      [,2]
##   1  -0.11999100 1.1046071
##   2   0.19605929 1.2380492
##   3  -0.13380684 1.3099856
##   4  -0.46684313 1.4373623
##   5  -0.35747556 0.8433896
##   6  -0.04499591 0.6622388
##   7  -0.29944357 0.9467181
##   8  -0.27414722 0.4370894
##   9  -0.14197276 0.5813874
##   10 -0.25748000 0.4123922
##   11  0.21748600 1.0405138
##   12  0.70495611 0.8933915
##   13  1.04243400 1.0472493
##
##     InvProm_X_InvestFacil
## Y           [,1]      [,2]
##   1  -2.7410000 5.9602022
##   2  -0.1371429 3.6894516
##   3   0.1042105 3.2268910
##   4   0.2340625 3.2273437
##   5  -0.5605556 3.4639852
##   6   0.9527273 2.6799204
##   7  -1.1803571 3.8990231
##   8  -0.9244444 2.4568998
##   9  -1.0931034 2.7685222
##   10 -0.2553333 2.9230485
##   11 -3.7520000 2.7213453
##   12 -1.0005556 5.8544548
##   13  5.1940000 0.6454301
```

#The first part of the output above shows the ratios of default (yes) and default (no) in the training set (called a priori probabilities), followed by a table giving for each target class, mean and standard deviation of the (sub-)variable. Also, note that the Naive Bayes algorithm assumes a Normal distribution for the independent variables. In accordance with the rule of the use of categorical predictors (the independent variables have been converted to categorical), we now have the conditional probabilities p(X|Y) for each attribute level given the default status.

Now, use the model on the test set

```r
# Predict the default status of test dataset
FinalProjectx_Predicted_Test_labels <-predict(FinalProjectx_nb_model,Test)
library(gmodels)
```

```
##
## Attaching package: 'gmodels'

## The following object is masked from 'package:pROC':
##
##     ci
```

```r
# Show the confusion matrix of the classifier
CrossTable(x=Test$FDIInflow,y=FinalProjectx_Predicted_Test_labels, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  61
##
##
##                | FinalProjectx_Predicted_Test_labels
## Test$FDIInflow |          1 |          3 |          4 |          6 |          7 |          8 |          9 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              1 |          0 |          0 |          0 |          0 |          0 |          1 |          1 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.200 |      0.200 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.032 |      0.500 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.016 |      0.016 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              2 |          0 |          0 |          0 |          0 |          0 |          1 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.500 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.032 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.016 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              3 |          0 |          0 |          0 |          0 |          0 |          2 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      1.000 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.065 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.033 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              4 |          0 |          0 |          1 |          2 |          0 |          6 |          0 |
##                |      0.000 |      0.000 |      0.111 |      0.222 |      0.000 |      0.667 |      0.000 |
##                |      0.000 |      0.000 |      0.250 |      0.200 |      0.000 |      0.194 |      0.000 |
##                |      0.000 |      0.000 |      0.016 |      0.033 |      0.000 |      0.098 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              5 |          0 |          0 |          2 |          2 |          0 |          1 |          0 |
##                |      0.000 |      0.000 |      0.400 |      0.400 |      0.000 |      0.200 |      0.000 |
##                |      0.000 |      0.000 |      0.500 |      0.200 |      0.000 |      0.032 |      0.000 |
##                |      0.000 |      0.000 |      0.033 |      0.033 |      0.000 |      0.016 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              6 |          0 |          1 |          0 |          2 |          0 |          2 |          0 |
##                |      0.000 |      0.125 |      0.000 |      0.250 |      0.000 |      0.250 |      0.000 |
##                |      0.000 |      1.000 |      0.000 |      0.200 |      0.000 |      0.065 |      0.000 |
##                |      0.000 |      0.016 |      0.000 |      0.033 |      0.000 |      0.033 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|-
##              7 |          0 |          0 |          0 |          1 |          0 |          3 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.250 |      0.000 |      0.750 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.100 |      0.000 |      0.097 |      0.000 |
```

```
##                 |      0.000 |      0.000 |      0.000 |      0.016 |      0.000 |      0.049 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##               8 |          0 |          0 |          0 |          0 |          1 |          5 |          0 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.167 |      0.833 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.333 |      0.161 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.016 |      0.082 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##               9 |          0 |          0 |          0 |          3 |          1 |          6 |          0 |
##                 |      0.000 |      0.000 |      0.000 |      0.300 |      0.100 |      0.600 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.300 |      0.333 |      0.194 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.049 |      0.016 |      0.098 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##              10 |          0 |          0 |          0 |          0 |          0 |          1 |          1 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.500 |      0.500 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.032 |      0.500 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.016 |      0.016 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##              11 |          0 |          0 |          0 |          0 |          1 |          1 |          0 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.500 |      0.500 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.333 |      0.032 |      0.000 |
##                 |      0.000 |      0.000 |      0.000 |      0.000 |      0.016 |      0.016 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##              12 |          1 |          0 |          0 |          0 |          0 |          2 |          0 |
##                 |      0.200 |      0.000 |      0.000 |      0.000 |      0.000 |      0.400 |      0.000 |
##                 |      1.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.065 |      0.000 |
##                 |      0.016 |      0.000 |      0.000 |      0.000 |      0.000 |      0.033 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##              13 |          0 |          0 |          1 |          0 |          0 |          0 |          0 |
##                 |      0.000 |      0.000 |      1.000 |      0.000 |      0.000 |      0.000 |      0.000 |
##                 |      0.000 |      0.000 |      0.250 |      0.000 |      0.000 |      0.000 |      0.000 |
##                 |      0.000 |      0.000 |      0.016 |      0.000 |      0.000 |      0.000 |      0.000 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##    Column Total |          1 |          1 |          4 |         10 |          3 |         31 |          2 |
##                 |      0.016 |      0.016 |      0.066 |      0.164 |      0.049 |      0.508 |      0.033 |
## ----------------|------------|------------|------------|------------|------------|------------|------------|
##
##
```

#Our results indicate that we mis-classified a total of 61 cases. X as False Positives, and X as False Negatives.

#It is sometimes useful to output the raw prediction probabilities rather than the predicted class. To do that, we use the raw option in the model.

```
FinalProjectx_nb_model <- naiveBayes(FDIInflow~ZInvestProm + ZInvestFacilitation + InvProm_X_InvestFaci
#Make predictions and return probability of each class
FinalProjectx_Predicted_Test_labels <-predict(FinalProjectx_nb_model,Test, type = "raw")
#show the first few values
head(FinalProjectx_Predicted_Test_labels)
```

```
##                 1           2           3           4            5            6
## [1,] 0.0011644780 0.005256853 0.004993379 0.008165373 0.0008375215 0.0019763934
```

```
## [2,] 0.0190977896 0.079574763 0.203206344 0.322557169 0.1185177170 0.0514498860
## [3,] 0.0170700583 0.049681619 0.079274841 0.149257722 0.0876675007 0.2654368020
## [4,] 0.0061754117 0.027651669 0.046535818 0.050708746 0.0492030231 0.1304118069
## [5,] 0.0447313521 0.081081202 0.075092435 0.092744193 0.0724049915 0.0239930385
## [6,] 0.0003418451 0.001593819 0.001366817 0.002430832 0.0001155969 0.0001570619
##                 7            8            9           10           11
## [1,] 0.0019022070 3.697715e-07 5.613707e-05 4.951443e-07 2.163980e-06
## [2,] 0.1526802881 1.086029e-02 3.264722e-02 1.938162e-03 3.350445e-03
## [3,] 0.0903512920 6.367680e-02 5.185610e-02 1.170996e-01 8.901817e-05
## [4,] 0.0539033179 3.348152e-01 1.595487e-01 1.278881e-01 2.414641e-03
## [5,] 0.1597461990 3.064064e-02 1.523207e-01 5.895453e-03 1.275929e-01
## [6,] 0.0003544651 1.661618e-09 2.056211e-06 1.902307e-09 3.046744e-07
##               12            13
## [1,] 0.010659395  9.649852e-01
## [2,] 0.004119921 1.992891e-130
## [3,] 0.026351297  2.187321e-03
## [4,] 0.010743620 1.733831e-149
## [5,] 0.133756860  1.007111e-46
## [6,] 0.003471704  9.901655e-01
```

```r
set.seed(123)
data <- data.frame(FinalProjectx = sample(c("True","False"), 250, replace = TRUE),
                   FinalProjectx_Predicted_Test_labels = sample(c("True","False"), 250, replace = TRUE)
                   )
table(data$FinalProjectx_Predicted_Test_labels, data$FinalProjectx)
```

```
##
##         False True
##   False    58   54
##   True     69   69
```

#The confusionMatrix function is very helpful as not only does it display a confusion matrix, it calculates many relevant statistics alongside:

```r
set.seed(123)
data <- data.frame(FinalProjectx = sample(c("True","False"), 250, replace = TRUE),
FinalProjectx_Predicted_Test_labels = sample(c("True","False"), 250, replace = TRUE)
)
library(caret)
confusionMatrix(as.factor(data$FinalProjectx_Predicted_Test_labels), as.factor(data$FinalProjectx), pos
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False    58   54
##      True     69   69
##
```

```
##                Accuracy : 0.508
##                  95% CI : (0.4443, 0.5716)
##     No Information Rate : 0.508
##     P-Value [Acc > NIR] : 0.5253
##
##                   Kappa : 0.0176
##
##  Mcnemar's Test P-Value : 0.2068
##
##             Sensitivity : 0.5610
##             Specificity : 0.4567
##          Pos Pred Value : 0.5000
##          Neg Pred Value : 0.5179
##              Prevalence : 0.4920
##          Detection Rate : 0.2760
##    Detection Prevalence : 0.5520
##       Balanced Accuracy : 0.5088
##
##        'Positive' Class : True
##
```

ROC Curves

We can now output the ROC curves. we should emember that ROC curves plot sensitivity (true positive rate) versus (1 - specificity), which is (1 - TNR) or false positive rate. See here for more details

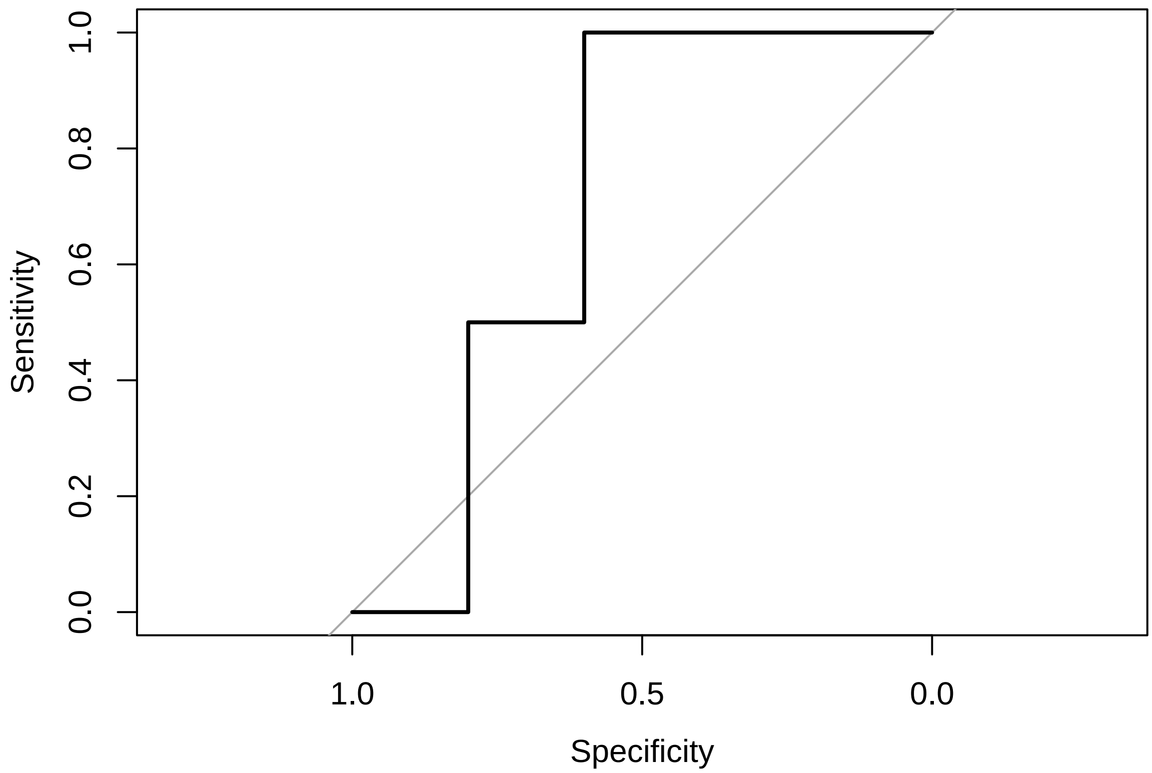
```
# install.packages("pROC") # install if necessary
library(pROC)
#Passing the second column of the predicted probabilities
#That column contains the probability associate to 'yes'
roc(Test$FDIInflow, FinalProjectx_Predicted_Test_labels[, 2])
```

```
## Warning in roc.default(Test$FDIInflow, FinalProjectx_Predicted_Test_labels[, :
## 'response' has more than two levels. Consider setting 'levels' explicitly or
## using 'multiclass.roc' instead

## Setting levels: control = 1, case = 2

## Setting direction: controls > cases

##
## Call:
## roc.default(response = Test$FDIInflow, predictor = FinalProjectx_Predicted_Test_labels[,     2])
##
## Data: FinalProjectx_Predicted_Test_labels[, 2] in 5 controls (Test$FDIInflow 1) > 2 cases (Test$FDIIn
## Area under the curve: 0.7
```

```
plot.roc(Test$FDIInflow,FinalProjectx_Predicted_Test_labels[,2])
```

```
## Warning in roc.default(x, predictor, plot = TRUE, ...): 'response' has more
## than two levels. Consider setting 'levels' explicitly or using 'multiclass.roc'
## instead
```

```
## Setting levels: control = 1, case = 2
## Setting direction: controls > cases
```



The AUC is 0.6667. The ROC curve is also plotted, though note that the X-Axis is Specificity (True Negative Rate), rather than 1-Specificity (False Positive Rate). This function can also be thought of as a plot of the FDI as a function of the Type I Error of the decision rule.

Box-Cox Transformation

We first illustrate the transformation of data using the Box-Cox transformation approach

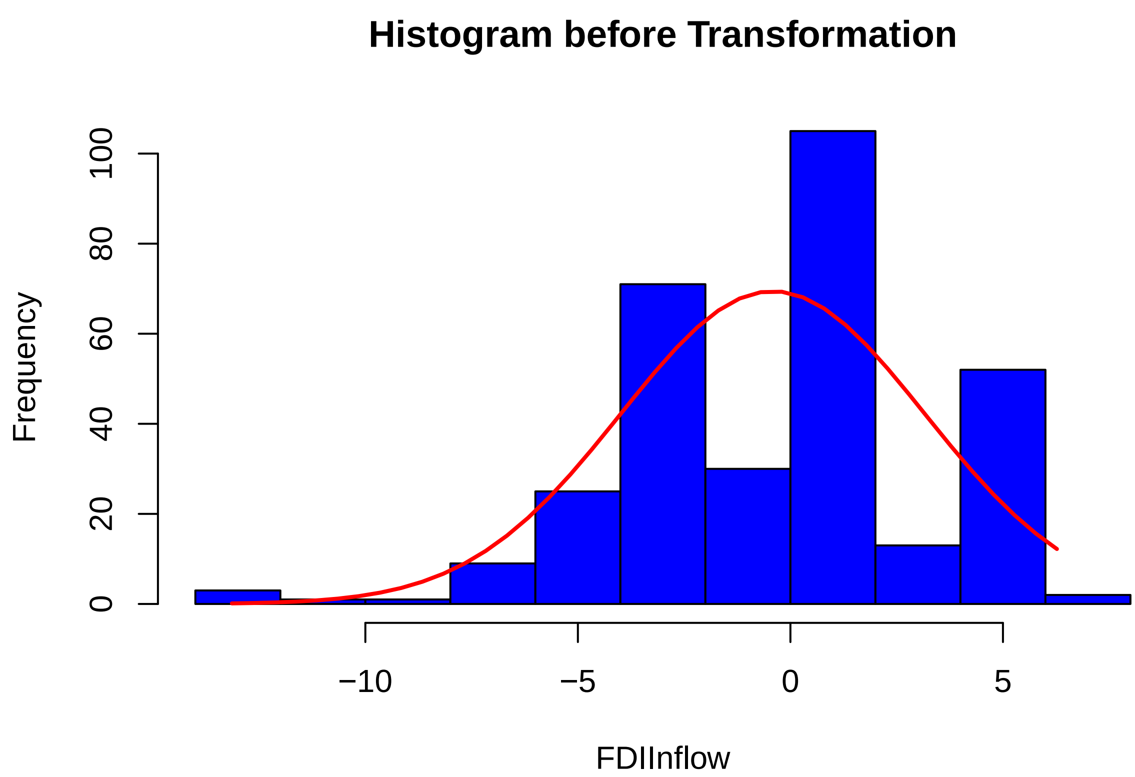
```
library(ISLR)
library(caret)
#Create a Box-Cox Transformation Model
FinalProjectx_Box_Cox_Transform<-preProcess(FinalProjectx,method = "BoxCox")
FinalProjectx_Box_Cox_Transform
```

```
## Created from 312 samples and 2 variables
##
## Pre-processing:
##   - Box-Cox transformation (2)
##   - ignored (0)
```

```
##
## Lambda estimates for Box-Cox transformation:
## 1.6, 0.8
```

Now, we apply the transformation

```
FinalProjectx_Transformed=predict(FinalProjectx_Box_Cox_Transform, FinalProjectx)
y <- FinalProjectx_Transformed$InvProm_X_InvestFacil
h<-hist(y, breaks=10, col="blue", xlab="FDIInflow",
   main="Histogram before Transformation")
xfit<-seq(min(y),max(y),length=40)
yfit<-dnorm(xfit,mean=mean(y),sd=sd(y))
yfit <- yfit*diff(h$mids[1:2])*length(y)
lines(xfit, yfit, col="red", lwd=2)
```



***

Hypertuning

```
library(caret)
library(ISLR)

set.seed(123)
#Divide data into test and train
```

```
Index_Train<-createDataPartition(FinalProjectx$FDIInflow, p=0.8, list=FALSE)
Train <-FinalProjectx[Index_Train,]
Test  <-FinalProjectx[-Index_Train,]
```

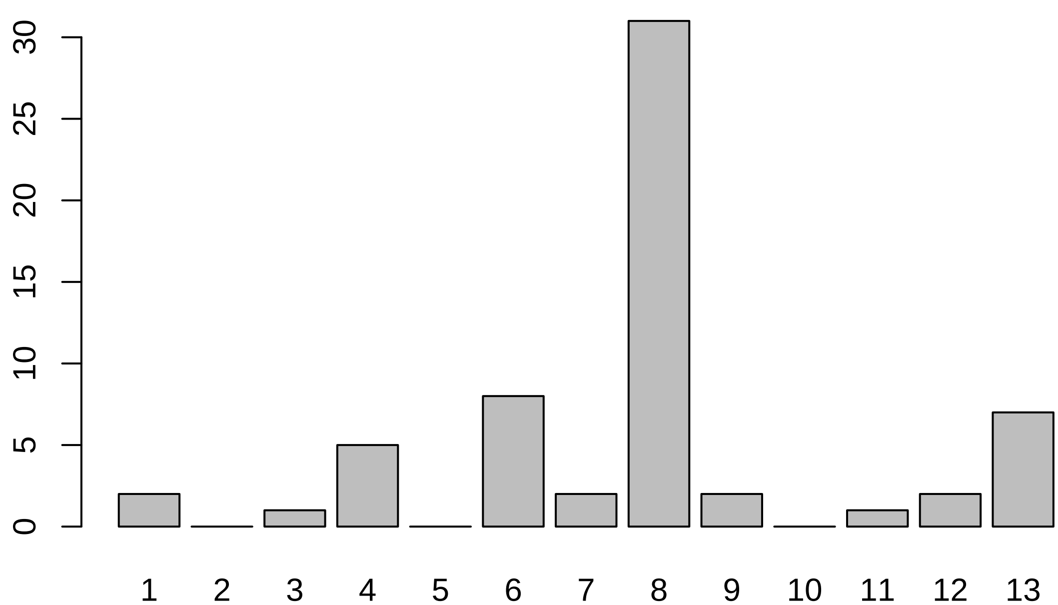
```
nb_model <-train(FDIInflow~ZInvestProm+ZInvestFacilitation+InvProm_X_InvestFacil, data = Train, preProc
```

```
## note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

```
# Predict the default status of test dataset
Predicted_Test_labels <-predict(FinalProjectx_nb_model,Test)
summary(Predicted_Test_labels)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13
##  2  0  1  5  0  8  2 31  2  0  1  2  7
```

```
plot(Predicted_Test_labels)
```




```
library(gmodels)
# Show the confusion matrix of the classifier
CrossTable(x=Test$FDIInflow,y=Predicted_Test_labels, prop.chisq = FALSE)
```

```
##
```

```
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  61
##
##
##                | Predicted_Test_labels
## Test$FDIInflow |          1 |          3 |          4 |          6 |          7 |          8 |          9 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              1 |          0 |          0 |          0 |          1 |          0 |          1 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.333 |      0.000 |      0.333 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.125 |      0.000 |      0.032 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.016 |      0.000 |      0.016 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              2 |          0 |          0 |          0 |          0 |          0 |          3 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.750 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.097 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.049 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              3 |          0 |          0 |          1 |          0 |          0 |          3 |          0 |
##                |      0.000 |      0.000 |      0.250 |      0.000 |      0.000 |      0.750 |      0.000 |
##                |      0.000 |      0.000 |      0.200 |      0.000 |      0.000 |      0.097 |      0.000 |
##                |      0.000 |      0.000 |      0.016 |      0.000 |      0.000 |      0.049 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              4 |          1 |          0 |          1 |          3 |          0 |          2 |          0 |
##                |      0.143 |      0.000 |      0.143 |      0.429 |      0.000 |      0.286 |      0.000 |
##                |      0.500 |      0.000 |      0.200 |      0.375 |      0.000 |      0.065 |      0.000 |
##                |      0.016 |      0.000 |      0.016 |      0.049 |      0.000 |      0.033 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              5 |          0 |          0 |          1 |          0 |          1 |          1 |          0 |
##                |      0.000 |      0.000 |      0.333 |      0.000 |      0.333 |      0.333 |      0.000 |
##                |      0.000 |      0.000 |      0.200 |      0.000 |      0.500 |      0.032 |      0.000 |
##                |      0.000 |      0.000 |      0.016 |      0.000 |      0.016 |      0.016 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              6 |          0 |          1 |          0 |          0 |          0 |          2 |          0 |
##                |      0.000 |      0.167 |      0.000 |      0.000 |      0.000 |      0.333 |      0.000 |
##                |      0.000 |      1.000 |      0.000 |      0.000 |      0.000 |      0.065 |      0.000 |
##                |      0.000 |      0.016 |      0.000 |      0.000 |      0.000 |      0.033 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              7 |          0 |          0 |          1 |          1 |          0 |          4 |          0 |
##                |      0.000 |      0.000 |      0.125 |      0.125 |      0.000 |      0.500 |      0.000 |
##                |      0.000 |      0.000 |      0.200 |      0.125 |      0.000 |      0.129 |      0.000 |
##                |      0.000 |      0.000 |      0.016 |      0.016 |      0.000 |      0.066 |      0.000 |
## ---------------|------------|------------|------------|------------|------------|------------|------------|
##              8 |          0 |          0 |          0 |          0 |          0 |          9 |          0 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.900 |      0.000 |
##                |      0.000 |      0.000 |      0.000 |      0.000 |      0.000 |      0.290 |      0.000 |
```

```
##                 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.148 |     0.000 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##               9 |         0 |         0 |         0 |         1 |         1 |         3 |         0 |
##                 |     0.000 |     0.000 |     0.000 |     0.167 |     0.167 |     0.500 |     0.000 |
##                 |     0.000 |     0.000 |     0.000 |     0.125 |     0.500 |     0.097 |     0.000 |
##                 |     0.000 |     0.000 |     0.000 |     0.016 |     0.016 |     0.049 |     0.000 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##              10 |         0 |         0 |         1 |         1 |         0 |         2 |         1 |
##                 |     0.000 |     0.000 |     0.200 |     0.200 |     0.000 |     0.400 |     0.200 |
##                 |     0.000 |     0.000 |     0.200 |     0.125 |     0.000 |     0.065 |     0.500 |
##                 |     0.000 |     0.000 |     0.016 |     0.016 |     0.000 |     0.033 |     0.016 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##              11 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |
##                 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |
##                 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |
##                 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##              12 |         1 |         0 |         0 |         1 |         0 |         1 |         1 |
##                 |     0.250 |     0.000 |     0.000 |     0.250 |     0.000 |     0.250 |     0.250 |
##                 |     0.500 |     0.000 |     0.000 |     0.125 |     0.000 |     0.032 |     0.500 |
##                 |     0.016 |     0.000 |     0.000 |     0.016 |     0.000 |     0.016 |     0.016 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##    Column Total |         2 |         1 |         5 |         8 |         2 |        31 |         2 |
##                 |     0.033 |     0.016 |     0.082 |     0.131 |     0.033 |     0.508 |     0.033 |
## ----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
##
##
```

```
set.seed(123)
data <- data.frame(FinalProjectx = sample(c("True","False"), 250, replace = TRUE),
Predicted_Test_labels = sample(c("True","False"), 250, replace = TRUE)
)
library(caret)
confusionMatrix(as.factor(data$Predicted_Test_labels), as.factor(data$FinalProjectx), positive = "True")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False    58   54
##      True     69   69
##
##                Accuracy : 0.508
##                  95% CI : (0.4443, 0.5716)
##     No Information Rate : 0.508
##     P-Value [Acc > NIR] : 0.5253
##
##                   Kappa : 0.0176
##
##  Mcnemar's Test P-Value : 0.2068
##
##             Sensitivity : 0.5610
##             Specificity : 0.4567
##          Pos Pred Value : 0.5000
```

```
##          Neg Pred Value : 0.5179
##              Prevalence : 0.4920
##          Detection Rate : 0.2760
##    Detection Prevalence : 0.5520
##       Balanced Accuracy : 0.5088
##
##        'Positive' Class : True
##
```