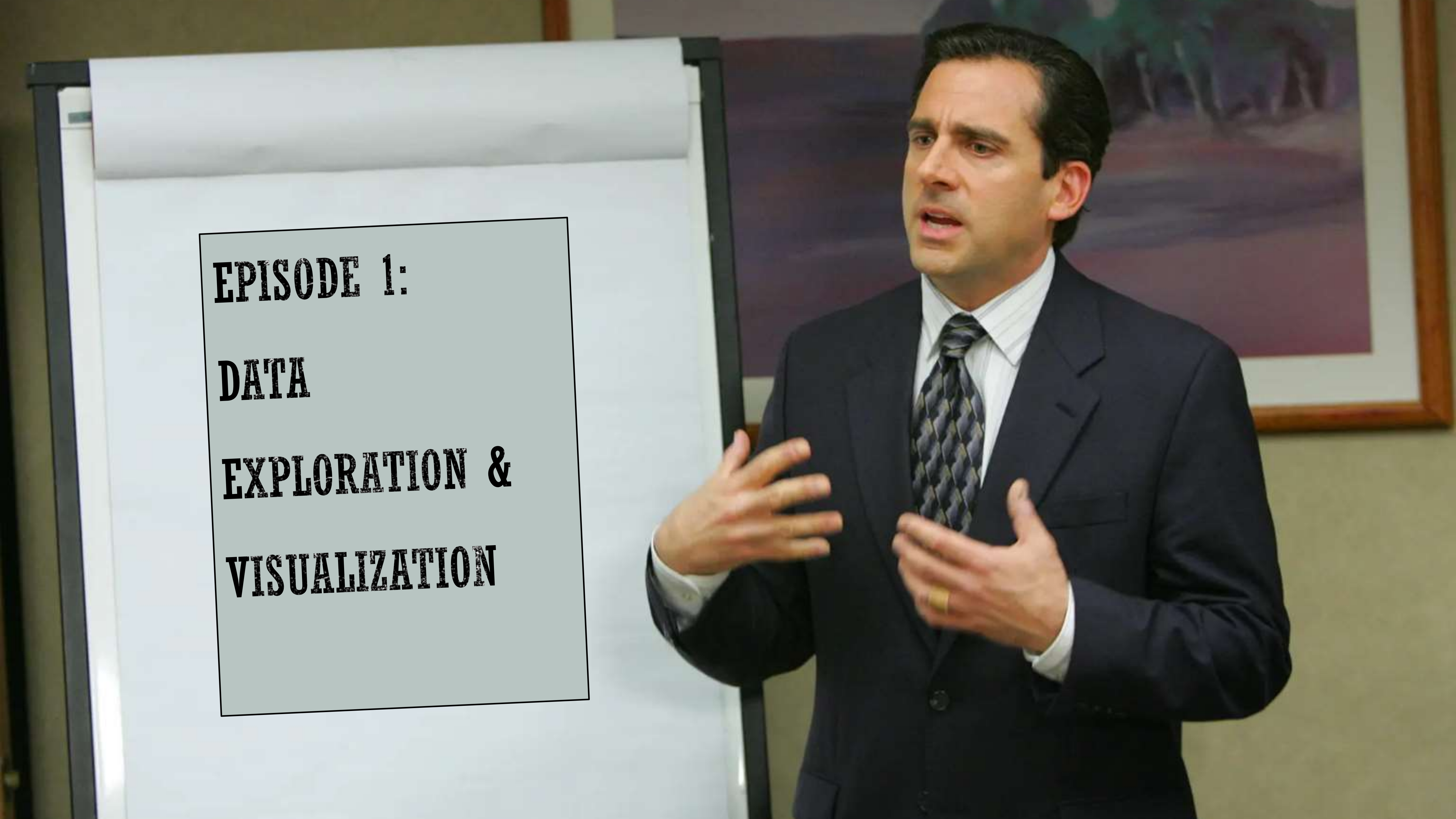
In association with

"Business Analytics with R"

Group-E Presents…

EPISODES:

- Data Exploration & Visualization
- [Classification] Decision Tree
- [Classification] Logistic Regression


EPISODE 1:
DATA
EXPLORATION &
VISUALIZATION

1.1 EACH VARIABLES STATISTICS AND SPECIFICATION

Structure of the dataset:

```
> str(data)
'data.frame':	1543 obs. of  17 variables:
 $ EmployeeID             : int  9 59 61 74 79 88 94 105 111 155 ...
 $ Branch                 : Factor w/ 13 levels "","Atlanta","Boston",..: 10 8 2 9 7 5 7 6 6 3 ...
 $ Tenure                 : num  10 5 8 4 7 2 10 5 6 6 ...
 $ Salary                 : num  73500 63500 59000 72000 83000 62500 73500 57000 65500 56000 ...
 $ Department             : Factor w/ 15 levels "Accounting","Administration",..: 4 3 3 5 8 7 3 3 7 14 ...
 $ JobSatisfaction        : num  4 4 3 4 3 3 3 3 3 3 ...
 $ WorkLifeBalance        : num  4 3 3.5 3.5 3.5 4.5 4.5 2.5 4 4 ...
 $ CommuteDistance        : Factor w/ 3 levels "Long","Medium",..: 3 2 2 1 2 2 1 3 3 3 ...
 $ MaritalStatus          : Factor w/ 3 levels "Divorced","Married",..: 3 3 3 3 2 3 1 3 3 2 ...
 $ Education              : Factor w/ 3 levels "Bachelor","High School",..: 1 1 3 1 3 1 1 1 1 1 ...
 $ PerformanceRating      : num  3.33 2.67 3.33 2.67 3 ...
 $ TrainingHours          : num  18 12 24 24 36 12 24 12 6 30 ...
 $ OverTime               : int  1 1 1 1 1 1 1 1 1 1 ...
 $ NumProjects            : num  2 3.62 3 2 4 2 2 2 3 3 ...
 $ YearsSincePromotion    : num  2 0 0 0 1 0 4 0 0 0 ...
 $ EnvironmentSatisfaction: num  1 1 1 1 1 1 1 1 1 1 ...
 $ ChurnLikelihood        : Factor w/ 2 levels "Highly Likely to Churn",..: 1 1 1 1 1 1 1 1 1 1 ...
```

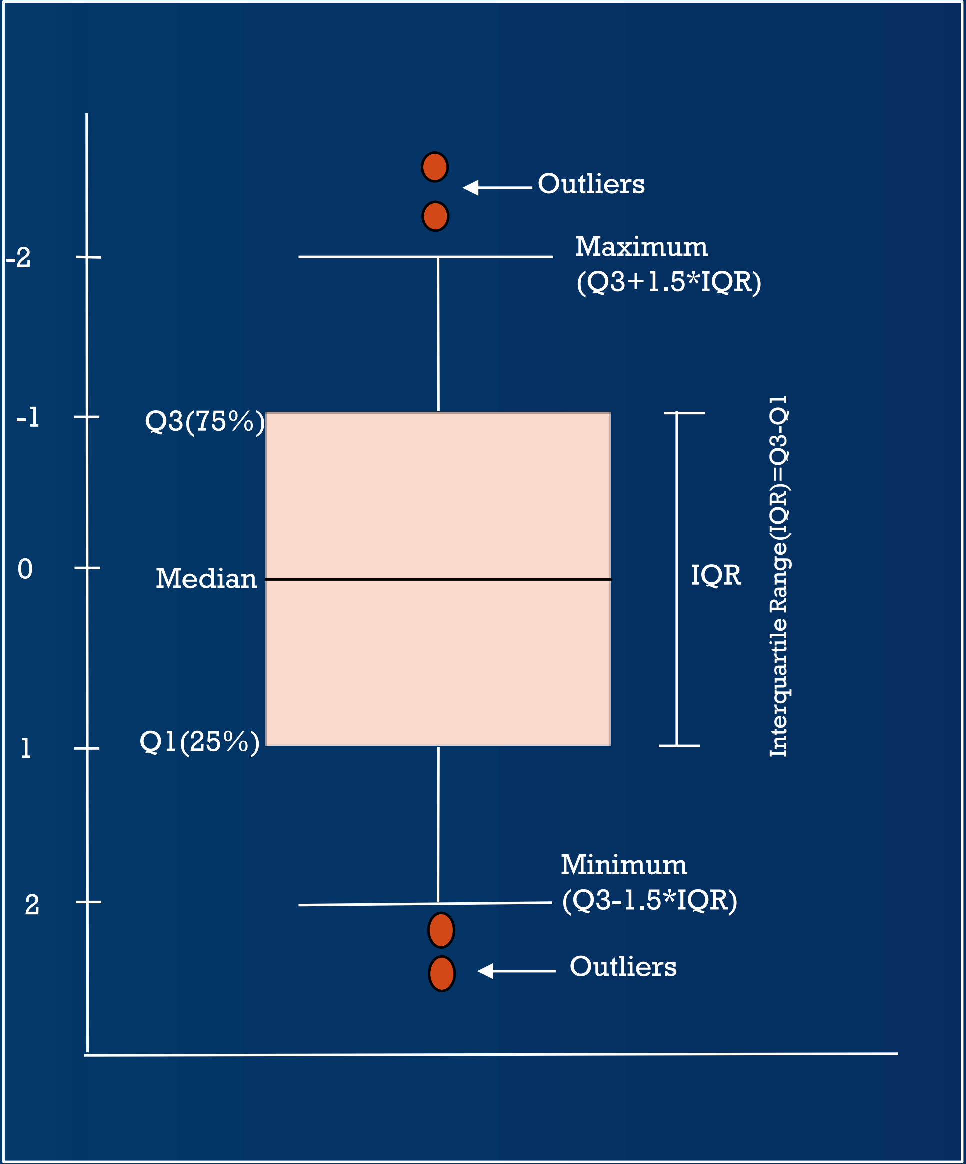
- 1543 observations (i.e., rows) and 17 variables (i.e. attributes or columns)
- There are 11 variables are numeric and 6 variables are categoric variables

No.	Each Variables Statistics and Specification	Mean	Std.	Min.	Max.
1)	**Employee ID**: A unique identifier for each employee.	772.0	445.6	1.0	1543.0
2)	**Tenure**: The number of years the employee has been with the company.	7.6	4.1	0.0	27.0
3)	**Salary**: The employee's annual salary.	66653.6	8448.9	40000	98000
4)	**Job Satisfaction**: The employee's self-reported job satisfaction level	3.4	1.1	1.0	5.0
5)	**WorkLife Balance**: Employee's work-life balance rating	3.8	0.5	2.0	5.0
6)	**Years Since Promotion**: Number of years since the employee's last promotion.	1.2	2.1	0.0	16.0
7)	**Training Hours**: The number of hours of training the employee has received.	34.1	21.6	5.2	96.0
8)	**Over Time**: Whether the employee works overtime or not.	0.9	0.2	0.0	1.0
9)	**Num Projects**: Number of projects the employee is currently working on.	3.5	0.7	1.5	6.4
10)	**Performance Rating**: The employee's performance rating	3.5	0.4	1.6	5.0
11)	**Environment Satisfaction**: Employee's environment satisfaction	NA	NA	NA	NA
12)	**Churn Likelihood:** Employee's likelihood to leave the company	NA	NA	NA	NA
13)	**Department**: The department in which the employee works	NA	NA	NA	NA
14)	**Commute Distance**: The distance the employee commutes to work	NA	NA	NA	NA
15)	**Marital Status**: The marital status of the employee	NA	NA	NA	NA
16)	**Education**: The highest level of education attained by the employee	NA	NA	NA	NA
17)	**Branch**: The "Branch" feature represents the geographic location of each employee	NA	NA	NA	NA

Independent Variables

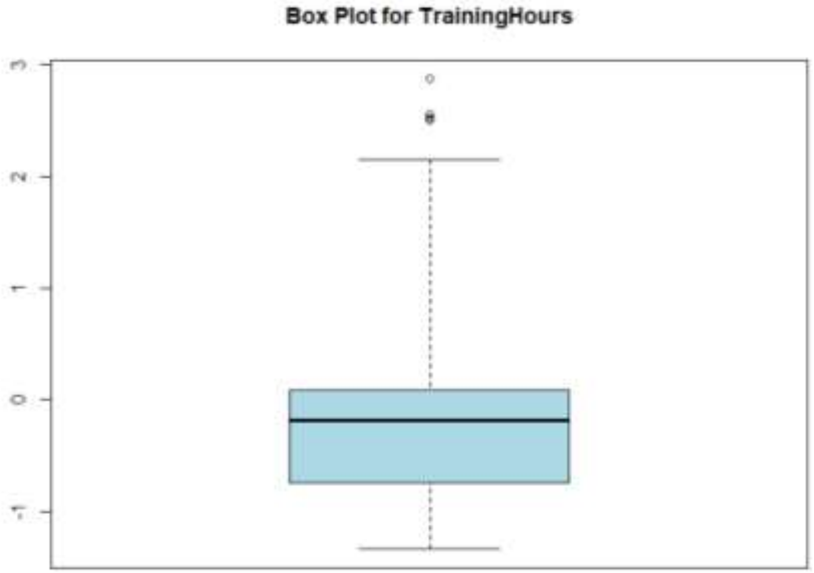
Dependent Variable

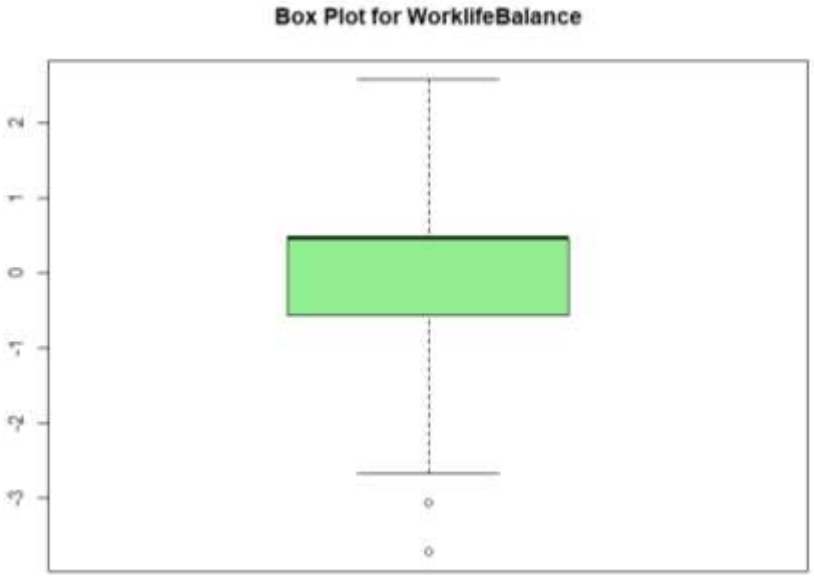
1.2 Outlier Detection by using Box Plot and Pre-processing

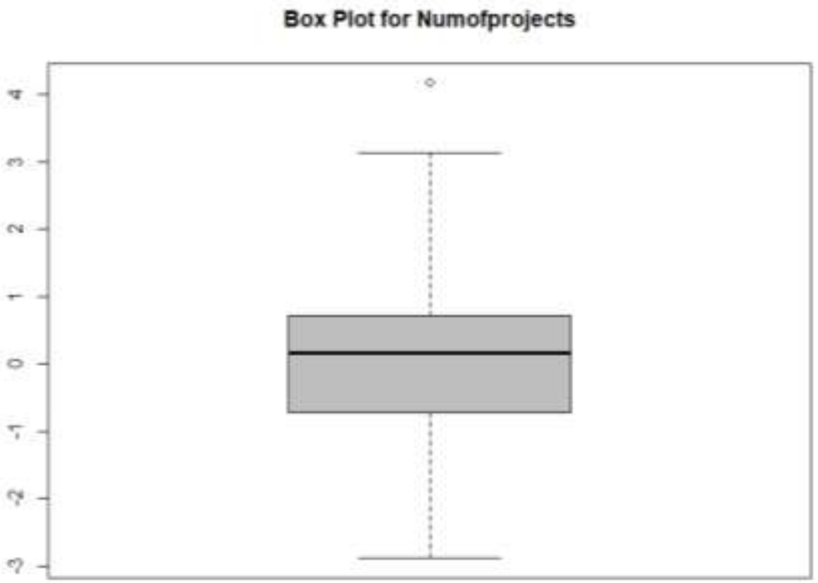


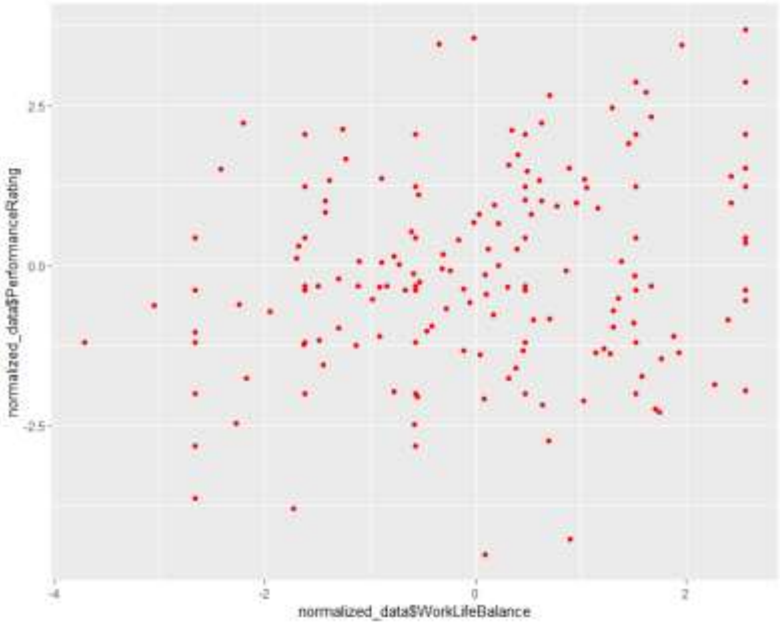

```
# Load data from the CSV file into a data frame named 'data'
data <- read.csv("output_file (2).csv", header=TRUE, stringsAsFactors=TRUE)
# Display the structure of the data frame
str(data)
# Identify numeric columns in the data frame using sapply and is.numeric
numeric_cols <- sapply(data, is.numeric)
# Subset the data frame to include only numeric columns
numeric_data <- data[, numeric_cols]
# Subset the data frame to include only non-numeric columns
non_numeric_data <- data[, !numeric_cols]
# Normalize the numeric data using z-score normalization
normalized_data <- as.data.frame(scale(numeric_data))
# Create a box plot for the 'TrainingHours' column
boxplot.default(normalized_data$TrainingHours,range= 2.5,col = "lightblue",main="Box Plot for TrainingHours")
# Create a box plot for the 'WorklifeBalance' column
boxplot.default(normalized_data$WorkLifeBalance,range= 2.0 ,col = "lightgreen",main="Box Plot for
WorklifeBalance")
# Create a box plot for the 'Numofprojects' column
boxplot.default(normalized_data$NumProjects,range= 2.0 ,col = "grey",main="Box Plot for Numofprojects")
# Create a scatter plot for 'TrainingHours', ' WorklifeBalance', 'NumProjects'.
ggplot(normalized_data)+geom_point(mapping
=aes(normalized_data$WorkLifeBalance,normalized_data$PerformanceRating),col="red")
ggplot(normalized_data)+geom_point(mapping =
aes(normalized_data$NumProjects,normalized_data$PerformanceRating),col="blue")
ggplot(normalized_data)+geom_point(mapping =
```

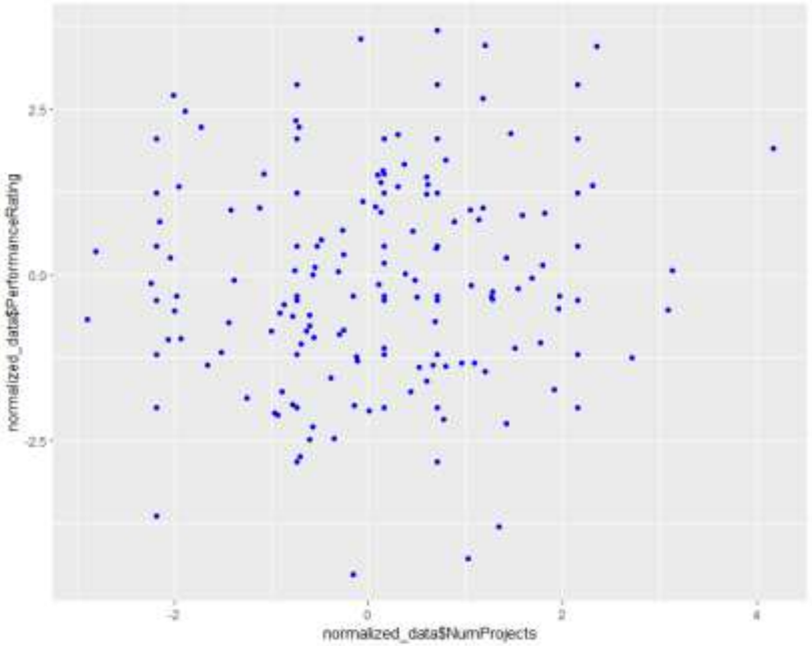
1.2 Outlier Detection by using Box Plot and Pre-processing

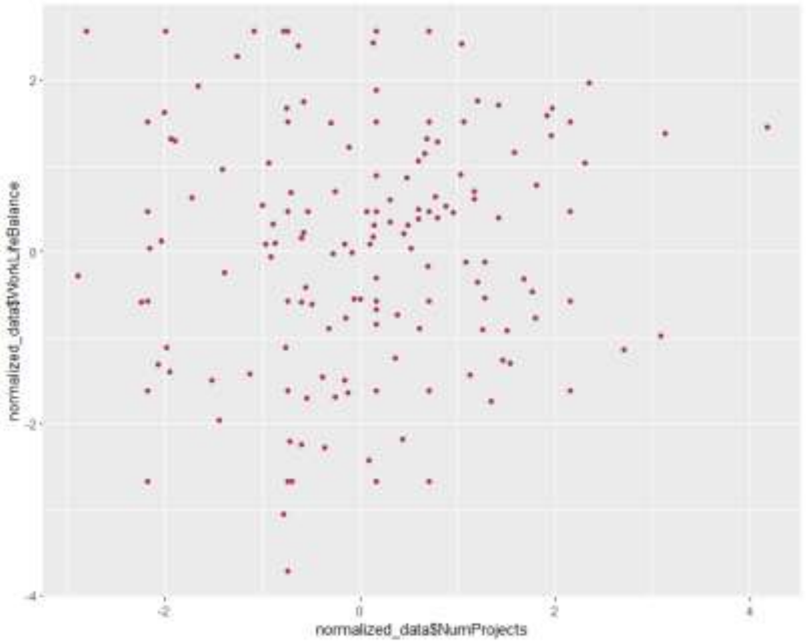


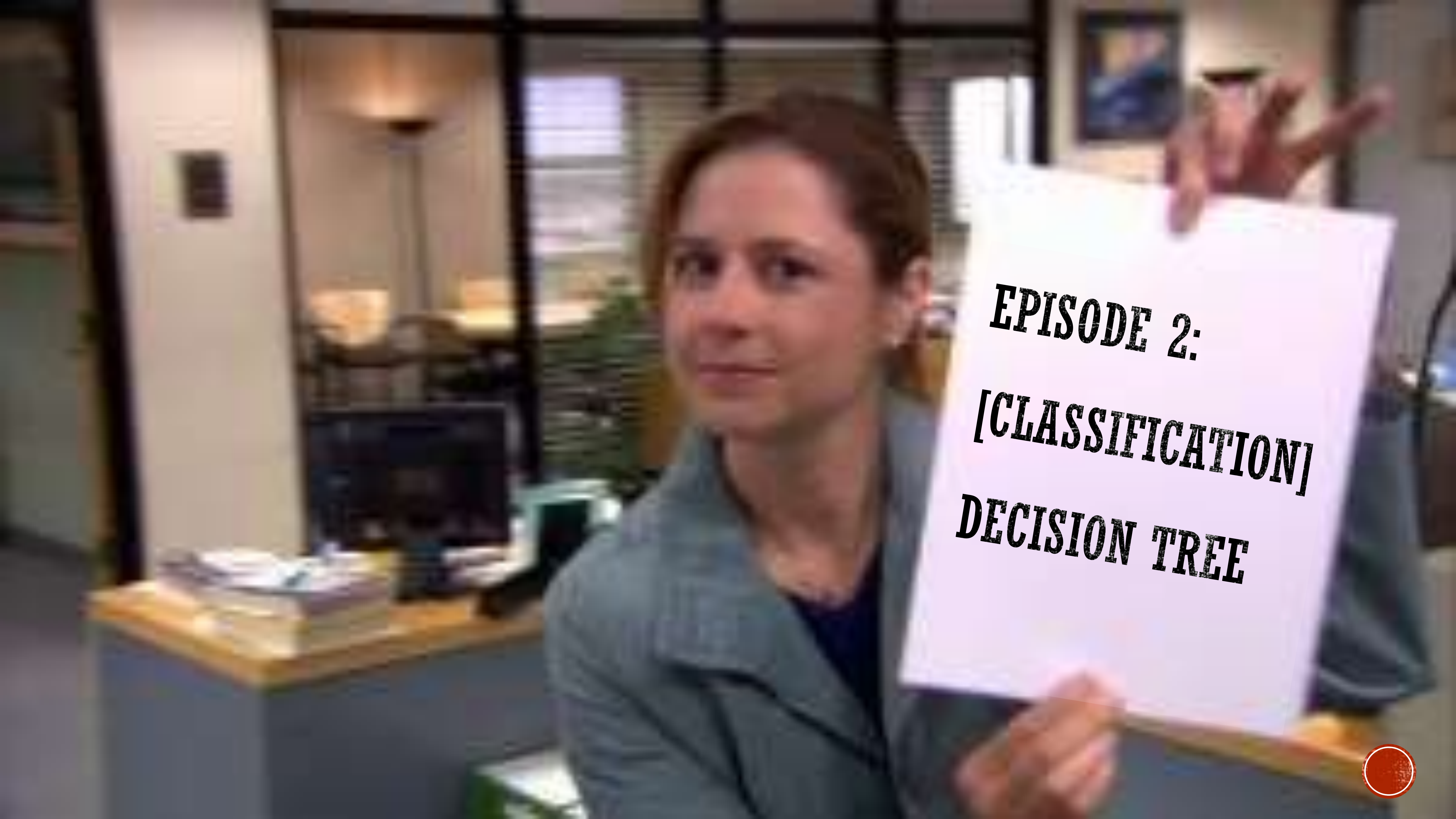











EPISODE 2:
[CLASSIFICATION]
DECISION TREE

2.1 Splitting the data into training and test data

```
getwd()
install.packages("rpart.plot")
install.packages("rpart")
library("rpart","rpart.plot")
# split the data into training and testing data sets
# Randomly select 2/3 of the rows
#set.seed(400) # for reproducible results
#set.seed(300) # for reproducible results
set.seed(345) # for reproducible results
train = sample(1:nrow(B_Project), nrow(B_Project)*(2/3))
train
# Using the train index set to split the dataset
# The split is 2/3 and 1/3 each for train and test set
B_Project.train = B_Project[train,]
B_Project.test = B_Project[-train,]
```

2.2 Growing a tree and display basic results

INPUT:

```
#EnvironmentSatisfaction+NumProjects+PerformanceRating
# The making of Decision Tree using rpart
fit = rpart(ChurnLikelihood ~., #Setting dependent variable. Rest are independent variables.
            data=B_Project.train, # dataframe used as the training data
            method="class", # treat survived as a categorical variable, default
            control=rpart.control(xval=0, minsplit=1),
            # xval: num of cross validation
            # minsplit=1: This will stop splitting if node has 1 or fewer observations
            parms=list(split="gini"))
fit
```

OUTPUT:

```
> fit
n= 1028

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 1028 439 Highly Likely (0.572957198 0.427042802)
   2) EnvironmentSatisfaction< 2.464 497   3 Highly Likely (0.993963783 0.006036217) *
   3) EnvironmentSatisfaction>=2.464 531  95 Less Likely (0.178907721 0.821092279)
     6) EnvironmentSatisfaction< 2.9315 36  18 Highly Likely (0.500000000 0.500000000)
      12) NumProjects< 3.89 22   5 Highly Likely (0.772727273 0.227272727) *
      13) NumProjects>=3.89 14   1 Less Likely (0.071428571 0.928571429) *
     7) EnvironmentSatisfaction>=2.9315 495  77 Less Likely (0.155555556 0.844444444) *
> |
```

2.3 Plotting a tree and make an interpretation

R code: rpart.plot(fit, type = 1, extra = 1,main ="Decision Tree For Employee Churn Prediction")

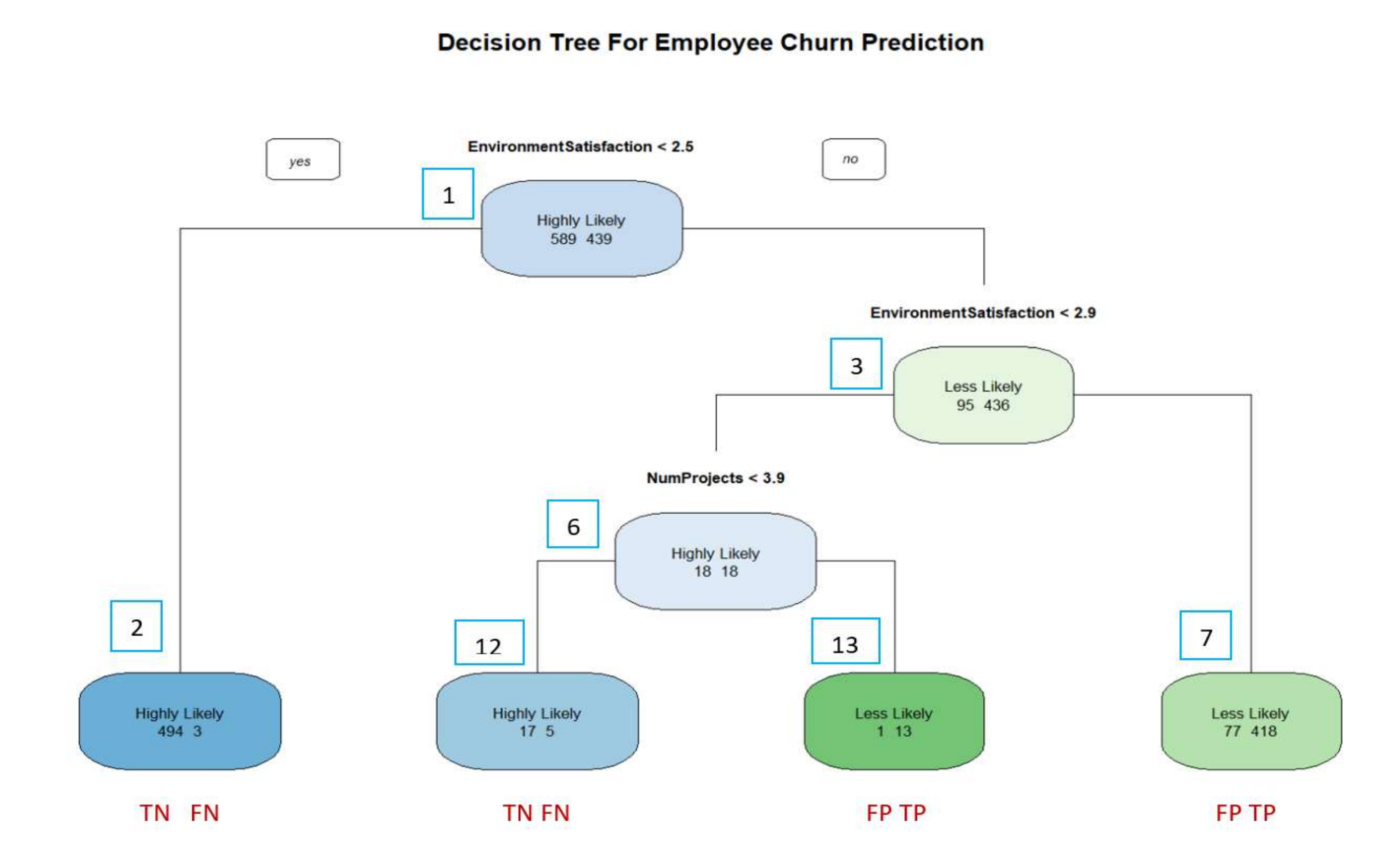
<u>Tree Interpretation Definitions</u>

True Positive (TP):
Pred Pos & Actual Pos
TP = 418+13 = 431

False Positive (FP):
Pred Pos & Actual Neg
FP = 1+77 = 78

True Negative (TN):
Pred Neg & Actual Neg
TN = 494 +17 = 511

False Negative (FN):
Pred Neg & Actual Pos
FN = 3+5 = 8



2.4 ACCURACY ON THE TRAINING & TEST DATA

INPUT:

```r
# extract the vector of predicted class for each observation in B_Project.train
B_Project.pred <- predict(fit, B_Project.train, type="class")
# extract the actual class of each observation in B_Project.train
B_Project.actual <- B_Project.train$ChurnLikelihood
# now building the confusion matrix
confusion.matrix <- table(B_Project.pred, B_Project.actual)
confusion.matrix
```

```r
# Accuracy on the Training Data
B_Project.pred<-predict(fit, B_Project.train, type="class")
B_Project.actual<-B_Project.train$ChurnLikelihood
confusion.matrix<-table(B_Project.pred, B_Project.actual)
pt<-prop.table(confusion.matrix)
#accuracy
pt[1,1] + pt[2,2]

# Accuracy on the Testing data
B_Project.pred <- predict(fit, B_Project.test, type="class")
B_Project.actual <- B_Project.test$ChurnLikelihood
confusion.matrix <- table(B_Project.pred, B_Project.actual)
addmargins(confusion.matrix)
pt <- prop.table(confusion.matrix)
#accuracy
pt[1,1] + pt[2,2]
```

OUTPUT:

```
> confusion.matrix <- table(B_Project.pred, B_Project.actual)
> confusion.matrix
                B_Project.actual
B_Project.pred  Highly Likely Less Likely
  Highly Likely           511           8
  Less Likely              78         431
```

```
> # Accuracy on the Training Data
> B_Project.pred<-predict(fit, B_Project.train, type="class")
> B_Project.actual<-B_Project.train$ChurnLikelihood
> confusion.matrix<-table(B_Project.pred, B_Project.actual)
> pt<-prop.table(confusion.matrix)
> #Accuracy on Training
> pt[1,1] + pt[2,2]
[1] 0.9163424
>
> # Accuracy on the Testing data
> B_Project.pred <- predict(fit, B_Project.test, type="class")
> B_Project.actual <- B_Project.test$ChurnLikelihood
> confusion.matrix <- table(B_Project.pred, B_Project.actual)
> addmargins(confusion.matrix)
                B_Project.actual
B_Project.pred  Highly Likely Less Likely Sum
  Highly Likely           253           5 258
  Less Likely              42         215 257
  Sum                     295         220 515
> pt <- prop.table(confusion.matrix)
> #Accuracy on Test
> pt[1,1] + pt[2,2]
[1] 0.9087379
```

*Overall Performance Accuracy (0.916)

Total Correct / Total Number of Obs.

(TP + TN)/ (TP+FP+TN+FN)= (431+511)/

(431+78+511+8) = 0.916

Error Rate = Total Incorrect / Total Number of Obs. = 1 – Accuracy = 1- 0.737 = 0.084

TPR = Recall = (Sensitivity) = TP/P = 431/(431+78) = 0.847

TNR = (Specificity) = TN/N = 511/(511+8) = 0.985

FPR = FP/N = Type 1 Error Rate (α)= 78/(519) = 0.151

FNR = FN/P = Type 2 Error Rate (β)= 8/(509) = 0.0157


IS THAT ENOUGH?
NO!
SHOULD I KEEP GOING?

EPISODE 3:

[CLASSIFICATION]

LOGISTIC

REGRESSION

No one's leaving the room until u watch the...

3.1 SPLITTING THE DATA INTO TRAINING AND TEST DATA

```
#Training and Test Data
churn.df<-read.csv("Employee Churn Dataset 2.csv") ; #Loading the data
churn.df$ChurnLikelihood<-as.factor(churn.df$ChurnLikelihood) ; #Converting output as factor
churn.df$Department<-factor(churn.df$Department) ; #Treating Department as categorical

set.seed(2) ; #Splitting the data into training and test data sets
train<-sample(1:nrow(churn.df), (0.6)*nrow(churn.df))
train.df<-churn.df[train,]
test.df<-churn.df[-train,]
```

```
> #Training and Test Data
> churn.df<-read.csv("Employee Churn Dataset 2.csv") ; #Loading the data
> churn.df$ChurnLikelihood<-as.factor(churn.df$ChurnLikelihood) ; #Converting output as factor
> churn.df$Department<-factor(churn.df$Department) ; #Treating Department as categorical
> set.seed(2) ; #Splitting the data into training and test data sets
> train<-sample(1:nrow(churn.df), (0.6)*nrow(churn.df))
> train.df<-churn.df[train,]
> test.df<-churn.df[-train,]
```

3.2 LOGISTIC REGRESSION AND DISPLAY THE RESULTS

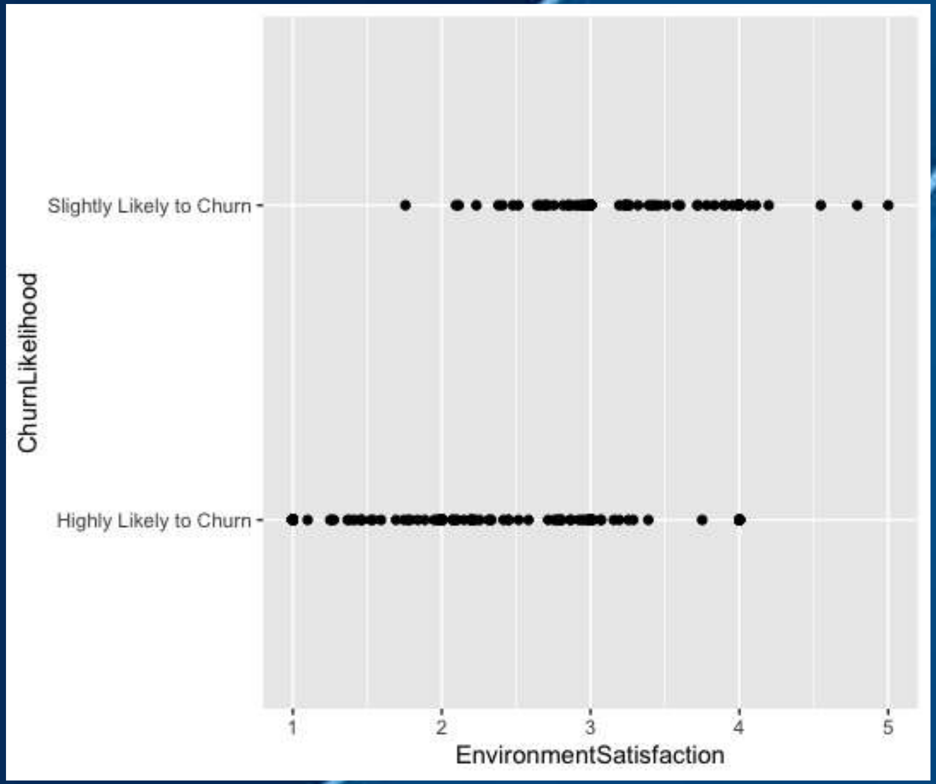
INPUT:

```
#Logistic regression and display the results
logit.reg <-glm(ChurnLikelihood ~ Salary + JobSatisfaction + WorkLifeBalance
                + CommuteDistance + MaritalStatus + PerformanceRating + TrainingHours
                + OverTime + NumProjects + YearsSincePromotion + EnvironmentSatisfaction,
                data = train.df, family = "binomial")

```

OUTPUT:

```
> #Logistic regression and display the results
> logit.reg <-glm(ChurnLikelihood ~ Salary + JobSatisfaction + WorkLifeBalance
+                 + CommuteDistance + MaritalStatus + PerformanceRating + TrainingHours
+                 + OverTime + NumProjects + YearsSincePromotion + EnvironmentSatisfaction,
+                 data = train.df, family = "binomial")
```



$$y = \frac{e^{(b_0 + b_1X)}}{1 + e^{(b_0 + b_1X)}}$$

3.3 INTERPRETATION OF SIGNIFICANCY AND COEFFICIENT

INPUT:

```
#Interpretation of Significancy and Coefficient
summary(logit.reg) ; #Results of logistic regression

#Accuracy on the training & test data
logitPredict<-predict(logit.reg, test.df, type = "response") ;#Computing predicted probabilities
logitPredictClass<-ifelse(logitPredict> 0.5, 1, 0) ; #Converting probability to a classification
logitPredictClass
```

OUTPUT:

```
> #Interpretation of Significancy and Coefficient
> summary(logit.reg) ; #Results of logistic regression

Call:
glm(formula = ChurnLikelihood ~ Salary + JobSatisfaction + WorkLifeBalance
    CommuteDistance + MaritalStatus + PerformanceRating + TrainingHours +
    OverTime + NumProjects + YearsSincePromotion + EnvironmentSatisfaction,
    family = "binomial", data = train.df)

Coefficients: (1 not defined because of singularities)
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              -1.68e+01   2.49e+00   -6.75  1.5e-11 ***
Salary                    3.12e-05   1.84e-05    1.69    0.091 .
JobSatisfaction          -8.01e-02   1.30e-01   -0.61    0.539
WorkLifeBalance          -2.86e-01   3.33e-01   -0.86    0.391
CommuteDistanceMedium    -7.60e-02   3.64e-01   -0.21    0.835
CommuteDistanceShort     -2.86e-01   3.55e-01   -0.81    0.420
MaritalStatusMarried     -3.44e-02   3.72e-01   -0.09    0.926
MaritalStatusSingle      -9.97e-02   4.59e-01   -0.22    0.828
PerformanceRating         4.58e-01   3.96e-01    1.16    0.247
TrainingHours             9.40e-03   6.85e-03    1.37    0.170
OverTimeTRUE                    NA         NA      NA       NA
NumProjects               2.58e-01   2.40e-01    1.08    0.282
YearsSincePromotion      -7.01e-03   8.51e-02   -0.08    0.934
EnvironmentSatisfaction   4.99e+00   4.09e-01   12.19  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpretation:

- $b_1 = 4.99$
- Therefore, Odds Ratio= $e^{b1} = e^{4.99} = 146.94$

Which means,

- An increase of 1 unit of Environment Satisfaction multiplies the odds of acceptance of a Likelihood by 146.94
- An increase of 1 unit of Environment Satisfaction with an increase of _____ in the odds of acceptance of a Likelihood

3.4 ACCURACY ON THE TRAINING & TEST DATA

INPUT:

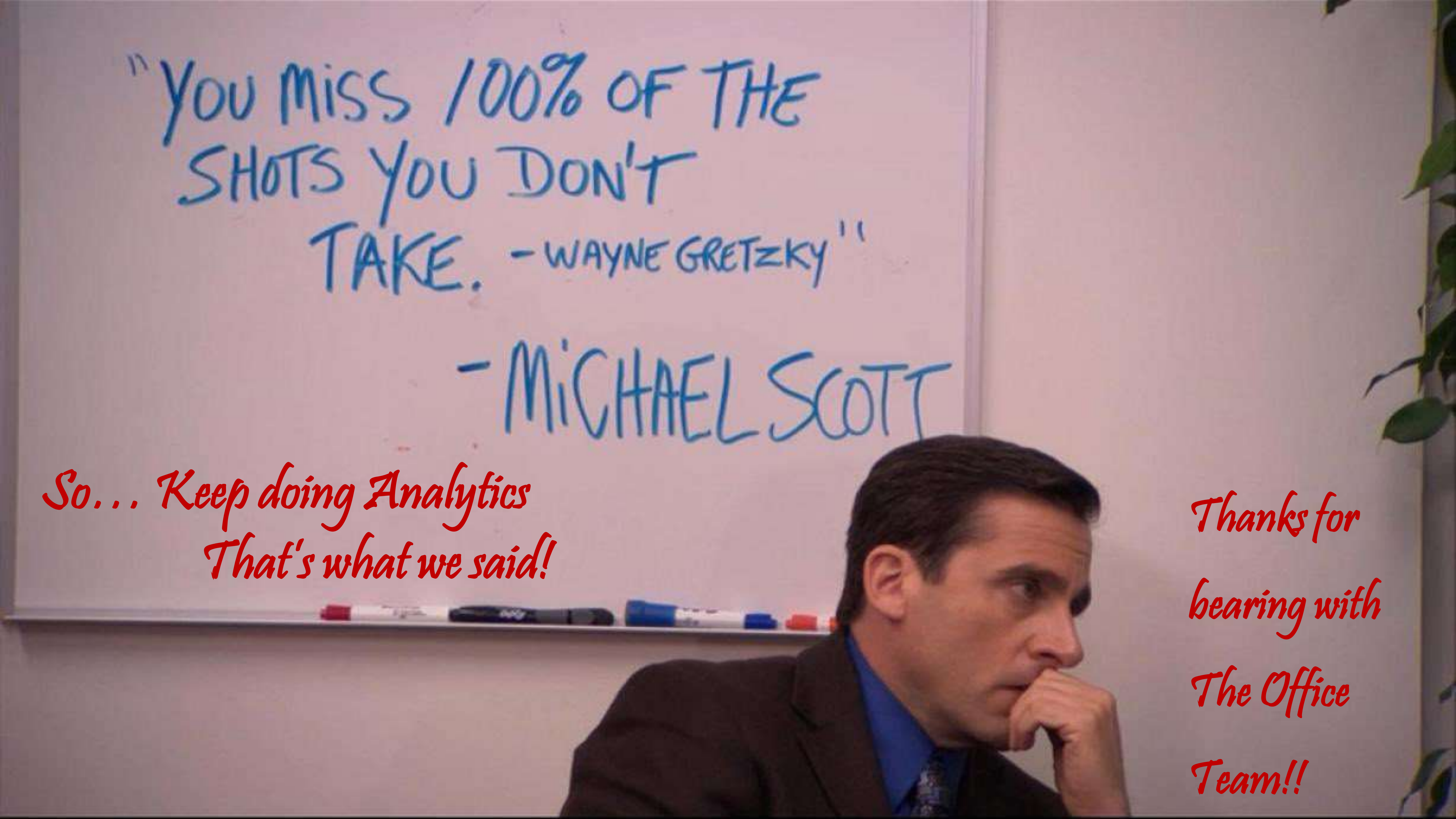
```
#Accuracy on the training & test data
logitPredict<-predict(logit.reg, test.df, type = "response") ;#Computing predicted probabilities
logitPredictClass<-ifelse(logitPredict> 0.5, 1, 0) ; #Converting probability to a classification
logitPredictClass
```

OUTPUT:

```
> #Accuracy on the training & test data
> logitPredict<-predict(logit.reg, test.df, type = "response") ;#Computing predicted probabilities
> logitPredictClass<-ifelse(logitPredict> 0.5, 1, 0) ; #Converting probability to a classification
> logitPredictClass
   2    4    5    6   10   12   14   17   18   19   22   30   33   40   41   51   62 
  NA    0    0    0    0    0    0   NA    0    0    0    0    0    0    0    0    0 
  63   64   66   67   69   72   73   74   76   78   81   86   87   91   92   94   95 
   0   NA    0    0    0    0   NA    0    0    0    0    0    0   NA    0    0    0 
  96   97   98  101  102  105  106  107  112  115  117  118  120  122  123  125  126 
   0    0   NA    0    0    0    0    0    0    0    0    0   NA   NA   NA    0    0 
 127  128  133  135  136  139  150  152  153  154  155  159  163  171  174  175  177 
   0    0    0    0    0    0    0    0    0   NA    0    0    0    0    0    0    0 
 178  183  189  195  197  198  202  208  209  210  211  216  217  219  220  224  225 
   0    0   NA    0    0    0   NA    0   NA    0    0    0    0    0    0    0   NA 
 229  230  232  236  237  240  242  244  245  246  249  250  256  257  261  265  271 
  NA   NA   NA   NA    0    0    0    0    0   NA    0    0    0    0    0    0    0 
 274  276  277  279  282  289  290  298  301  302  307  308  309  312  314  316  319 
  NA    0    0    0   NA    0    0    0    0    0    0    0    0   NA    0    0    0 
 321  322  324  330  332  336  338  339  345  346  351  354  355  356  359  360  363 
   0    0    0    0   NA    0    0    0    0   NA    0    0    0   NA    0    0    0 
 364  366  367  372  375  378  379  380  382  384  386  387  389  393  394  396  400 
   0   NA   NA   NA    0    0    0   NA    0    0    0    0   NA    0   NA    0    0 
 402  403  406  407  409  411  412  414  417  418  420  421  422  423  424  425  426 
   0    0    0   NA    0   NA   NA    0    0    0    0    0    0    0    0   NA   NA 
 433  436  438  439  441  448  449  450  451  453  458  459  463  467  470  477  481 
  NA    0    0   NA    0    0    0    0    0    0    0    0    0   NA    0    0   NA 
 485  487  489  490  491  494  495  496  497  501  504  506  507  511  513  515  517 
   0    0    0    0    0    0    0   NA   NA    0    0    0    0   NA    0    0   NA 
 521  527  530  531  532  535  536  537  540  541  542  544  545  546  548  551  553 
   0    0    0    0    0    0    0   NA    0    0    0    0    0    0    0   NA    0 
 555  557  559  560  561  562  565  566  569  570  572  573  574  575  576  578  579 
  NA    0   NA    0    0   NA    0    0    0    0    0    0    0    0    0    0    0 
 580  587  589  593  595  597  598  601  602  605  609  610  611  612  616  617  625 
   0   NA    0    0    0    0   NA    0    0    0   NA    0   NA   NA    0    0    0 
 626  631  633  641  642  646  650  651  653  656  657  664  667  669  673  674  682 
   0   NA    0    0    0    0    0    0    0    0    0    0    0    0   NA    0    0 
 686  688  692  694  695  702  703  707  712  715  718  721  722  723  724  726  731 
   0    0    0    0    0    0    0    0    0   NA   NA    0    0   NA    0    0    0 
```

```
 733  737  738  744  745  746  748  752  755  756  757  758  759  760  761  762  763 
   0    0    0    0    0    0   NA    1    1    1   NA   NA    0    1    0    1    1 
 766  767  769  770  771  778  782  785  790  791  793  795  798  800  805  806  808 
   1   NA    1    1    1    1    1    1    1    1   NA   NA    1   NA    1    1    1 
 812  813  819  820  821  825  831  832  833  835  844  847  852  853  858  863  864 
   1   NA    1    1    1    1    1    1   NA    1    1    1   NA    1    1   NA    1 
 865  867  871  877  879  883  884  885  888  892  895  896  898  900  903  904  905 
   1    1    1    1    1    1   NA    1    1   NA    1    1    1    1   NA    1    1 
 911  913  914  915  921  924  925  926  930  931  935  939  948  949  951  952  953 
   1    1   NA    1    1    1    1    1    1    1   NA   NA    1   NA    1    1   NA 
 954  955  956  957  958  961  964  973  974  977  983  984  985  988  990  991  992 
  NA    1    1    1    1    1    1    1   NA    1    1   NA    1    1    1    1   NA 
 995  999 1001 1004 1009 1012 1014 1015 1017 1019 1021 1026 1027 1029 1031 1035 1036 
   1    1    1    1    1    1    1    1    1    1    1   NA    1    1    1    1   NA 
1040 1043 1047 1048 1050 1052 1053 1054 1056 1061 1066 1067 1068 1070 1071 1075 1079 
   1    1    1    1   NA   NA    1    1   NA    1    1   NA    1    1    1    1   NA 
1082 1085 1088 1096 1097 1101 1103 1104 1105 1106 1108 1109 1114 1123 1124 1125 1126 
   1   NA   NA   NA   NA    1    1    1    1    1   NA   NA    1    1    1    1    1 
1127 1128 1136 1137 1139 1140 1141 1142 1144 1145 1147 1148 1149 1155 1156 1158 1159 
   1    1    1    1    1    1    1    1    1    1   NA    1    1    1   NA    1    1 
1160 1162 1164 1166 1167 1169 1172 1176 1178 1181 1182 1185 1190 1192 1193 1194 1195 
   1    1    1   NA    1    1    1    1    1    1    1    1    1   NA    1    1    1 
1196 1199 1201 1202 1205 1206 1209 1213 1217 1218 1221 1222 1224 1229 1230 1231 1235 
   1    1   NA    1    1    1    1    1    1   NA    1    1   NA    1   NA   NA    1 
1237 1238 1239 1243 1245 1248 1250 1252 1254 1256 1258 1259 1260 1261 1265 1270 1274 
   1   NA    1    1   NA    1    1    1   NA    1    1    1    1   NA   NA   NA    1 
1275 1277 1278 1279 1284 1285 1289 1293 1300 1303 1304 1305 1311 1312 1316 1319 1324 
   1   NA    1    1    1    1    1    1    1    1    1    1    1   NA    1    1    1 
1328 1329 1331 1332 1333 1334 1335 1336 1338 1343 1344 1348 1349 1350 1351 1353 1356 
   1    1   NA   NA    1    1   NA    1    1    1   NA    1    1    1    1   NA    1 
1358 1360 1364 1365 1366 1370 1378 1379 1381 1383 1385 1388 1390 1393 1394 1397 1398 
   1    1   NA    1   NA    1   NA    1    1   NA    1    1    1    1    1   NA    1 
1401 1406 1409 1411 1412 1414 1417 1420 1422 1424 1427 1428 1429 1433 1435 1436 1437 
  NA    1    1   NA   NA    1    1    1    1    1    1    1    1    1   NA    1    1 
1439 1440 1441 1442 1445 1446 1458 1461 1464 1468 1469 1472 1476 1482 1489 1491 1493 
   1    1    1    1    1   NA    1    1   NA    1    1   NA    1   NA   NA    1    1 
1494 1496 1498 1504 1507 1510 1512 1516 1518 1519 1521 1525 1526 1531 1532 1533 1535 
  NA    1    1    1   NA    1    1   NA   NA   NA   NA   NA   NA   NA   NA   NA   NA 
```

```
1538 1539 1540 1541 1542 1543 
  NA   NA   NA   NA   NA   NA 
```


"YOU MISS 100% OF THE SHOTS YOU DON'T TAKE. -WAYNE GRETZKY"
-MICHAEL SCOTT
So… Keep doing Analytics
That's what we said!
Thanks for bearing with The Office Team!!