PRAHLAD

2019

Management And Field Engineers Are Looking For Answers From Analytics!

- Q1 –> Analyze the given dataset and create a model that can predict the response variable
- Q2 –> What are some ways to make sure the model that you create more robust to outliers?
- Q3 –> What could be some issues if the distribution of the test data is significantly different than the distribution of the training data?
- Q4 –> Regex Test

Q1 –> Analyze the given dataset and create a model that can predict the response variable

Read Data

```
library(readxl)

## Warning: package 'readxl' was built under R version 3.5.3

#Load DataSet

df <-
read.csv("C:/Users/TOSHIBA/Desktop/DataSet.csv",sep=";",stringsAsFactors =
FALSE,header=FALSE,na.strings="?")

# Load RegexTask

regex_data <- read_excel("C:/Users/TOSHIBA/Desktop/regex.xlsx")
```

Understanding Data

```
# View sie and columns

str(df)

## 'data.frame':    690 obs. of  16 variables:
##  $ V1 : chr  "b" "a" "a" "b" ...
##  $ V2 : chr  "30,83" "58,67" "24,5" "27,83" ...
##  $ V3 : chr  "0" "4,46" "0,5" "1,54" ...
##  $ V4 : chr  "u" "u" "u" "u" ...
##  $ V5 : chr  "g" "g" "g" "g" ...
##  $ V6 : chr  "w" "q" "q" "w" ...
##  $ V7 : chr  "v" "h" "h" "v" ...
```

```
##  $ V8 : chr  "1,25" "3,04" "1,5" "3,75" ...
##  $ V9 : chr  "t" "t" "t" "t" ...
##  $ V10: chr  "t" "t" "f" "t" ...
##  $ V11: int  1 6 0 5 0 0 0 0 0 0 ...
##  $ V12: chr  "f" "f" "f" "t" ...
##  $ V13: chr  "g" "g" "g" "g" ...
##  $ V14: int  202 43 280 100 120 360 164 80 180 52 ...
##  $ V15: int  0 560 824 3 0 0 31285 1349 314 1442 ...
##  $ V16: chr  "+" "+" "+" "+" ...
```

```
# View 10 rows

head(df,10)
```

```
##    V1    V2     V3 V4 V5 V6 V7    V8 V9 V10 V11 V12 V13 V14   V15 V16
## 1   b 30,83      0  u  g  w  v  1,25  t   t   1   f   g 202     0   +
## 2   a 58,67   4,46  u  g  q  h  3,04  t   t   6   f   g  43   560   +
## 3   a  24,5    0,5  u  g  q  h   1,5  t   f   0   f   g 280   824   +
## 4   b 27,83   1,54  u  g  w  v  3,75  t   t   5   t   g 100     3   +
## 5   b 20,17  5,625  u  g  w  v  1,71  t   f   0   f   s 120     0   +
## 6   b 32,08      4  u  g  m  v   2,5  t   f   0   t   g 360     0   +
## 7   b 33,17   1,04  u  g  r  h   6,5  t   f   0   t   g 164 31285   +
## 8   a 22,92 11,585  u  g cc  v  0,04  t   f   0   f   g  80  1349   +
## 9   b 54,42    0,5  y  p  k  h  3,96  t   f   0   f   g 180   314   +
## 10  b  42,5  4,915  y  p  w  v 3,165  t   f   0   t   g  52  1442   +
```

```
#Converting to numeric

df$V2 <- as.numeric(gsub(",","",df$V2))

df$V3 <- as.numeric(gsub(",","",df$V3))

df$V8 <- as.numeric(gsub(",","",df$V8))

df$V11 <-as.numeric(df$V11)

df$V14 <-as.numeric(df$V14)

df$V15 <-as.numeric(df$V15)


# Converting Dependent Var to factor

df$V16 <- as.factor(ifelse(df$V16=="+","POSITIVE","NEGATIVE"))
```

Are There Any Missing Value In Data?

```
sapply(df,function(x) sum(is.na(x)))
```

```
##  V1  V2  V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 
##  12  12   0   6   6   9   9   0   0   0   0   0   0  13   0   0
```

Yes there are missing values 37/690 = 5%

Exploratory Analysis to Identify Set Of Important Features

Summary Of Data

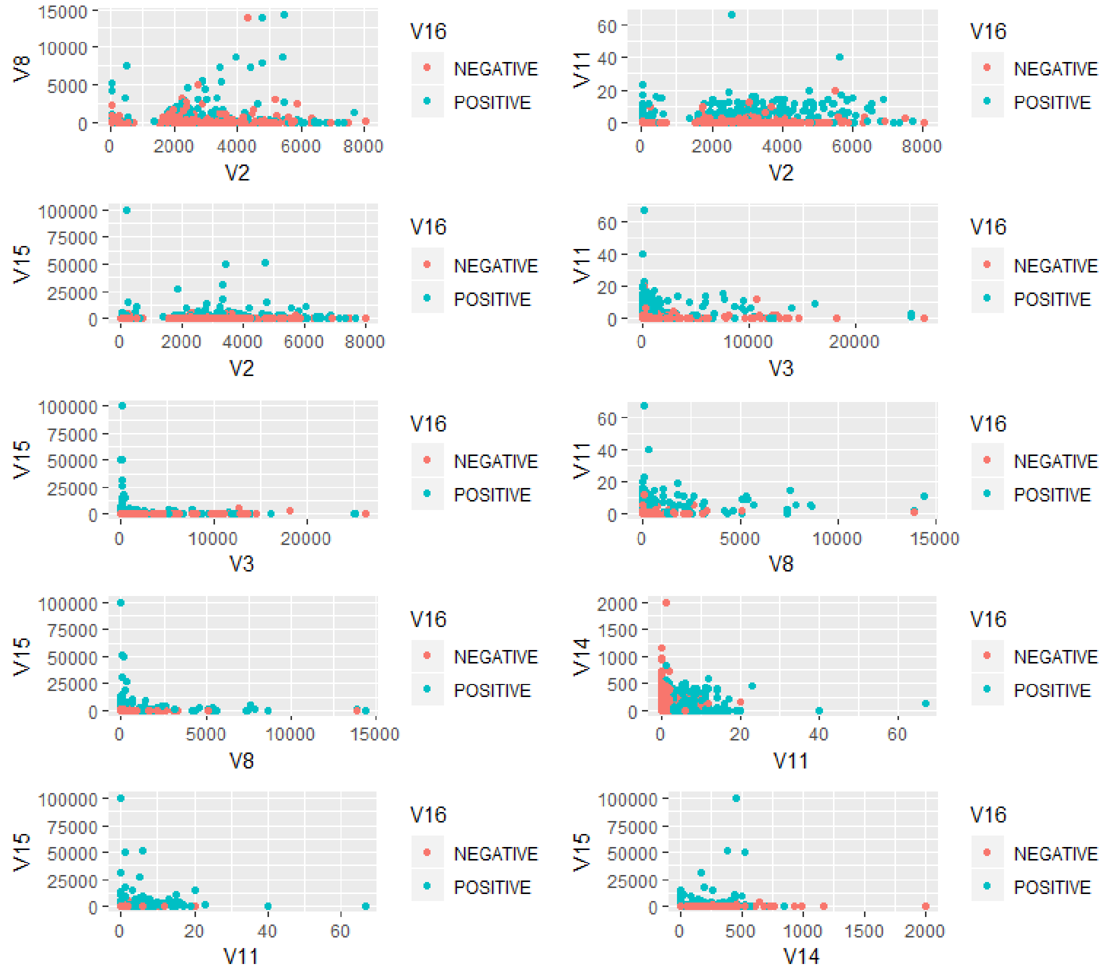
```
summary(df)

##       V1                   V2             V3              V4           
##  Length:690         Min.   :  16   Min.   :    0   Length:690        
##  Class :character   1st Qu.:1921   1st Qu.:   15   Class :character  
##  Mode  :character   Median :2600   Median :  125   Mode  :character  
##                     Mean   :2689   Mean   : 1187                     
##                     3rd Qu.:3571   3rd Qu.:  665                     
##                     Max.   :8025   Max.   :26335                     
##                     NA's   :12                                       
##       V5                 V6                 V7           
##  Length:690         Length:690         Length:690        
##  Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character  
##                                                          
##                                                          
##                                                          
##                                                          
##        V8                V9                 V10                 V11      
##  Min.   :    0.0   Length:690         Length:690         Min.   : 0.0  
##  1st Qu.:    5.0   Class :character   Class :character   1st Qu.: 0.0  
##  Median :   35.0   Mode  :character   Mode  :character   Median : 0.0  
##  Mean   :  453.4                                         Mean   : 2.4  
##  3rd Qu.:  219.8                                         3rd Qu.: 3.0  
##  Max.   :14415.0                                         Max.   :67.0  
##                                                                        
##      V12                V13                 V14             V15          
##  Length:690         Length:690         Min.   :   0   Min.   :     0.0  
##  Class :character   Class :character   1st Qu.:  75   1st Qu.:     0.0  
##  Mode  :character   Mode  :character   Median : 160   Median :     5.0  
##                                        Mean   : 184   Mean   :  1017.4  
##                                        3rd Qu.: 276   3rd Qu.:   395.5  
##                                        Max.   :2000   Max.   :100000.0  
##                                        NA's   :13                       
##        V16     
##  NEGATIVE:383  
##  POSITIVE:307  
##                
##                
## 
```

```
##
##
```

Can Numeric Variables Help In Seggrgeating The Class? Visually Checking..

```
# NUmeric vars: V2,V3,V8,V11,V14,V15

# For every variable plot with other variables and so on...There will be 15 combinations

#Load Packages

suppressMessages(library(ggplot2))

suppressMessages(library(gridExtra))

suppressMessages(library(purrr))

suppressMessages(library(tidyr))

suppressMessages(library(dplyr))

scatter_fun = function(x, y) {
     ggplot(df, aes_string(x = x, y = y,colour="V16") ) +
          geom_point()
}

# Plotting significant vars identified from above exercise

p1 <- scatter_fun("V2","V8")

p2 <- scatter_fun("V2","V11")

p3<- scatter_fun("V2","V15")

p4 <-scatter_fun("V3","V11")

p5 <-scatter_fun("V3","V15")

p6 <-scatter_fun("V8","V11")

p7 <-scatter_fun("V8","V15")
```

```
p8 <-scatter_fun("V11","V14")

p9<-scatter_fun("V11","V15")

p10<-scatter_fun("V14","V15")


grid.arrange(p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,ncol=2)
```

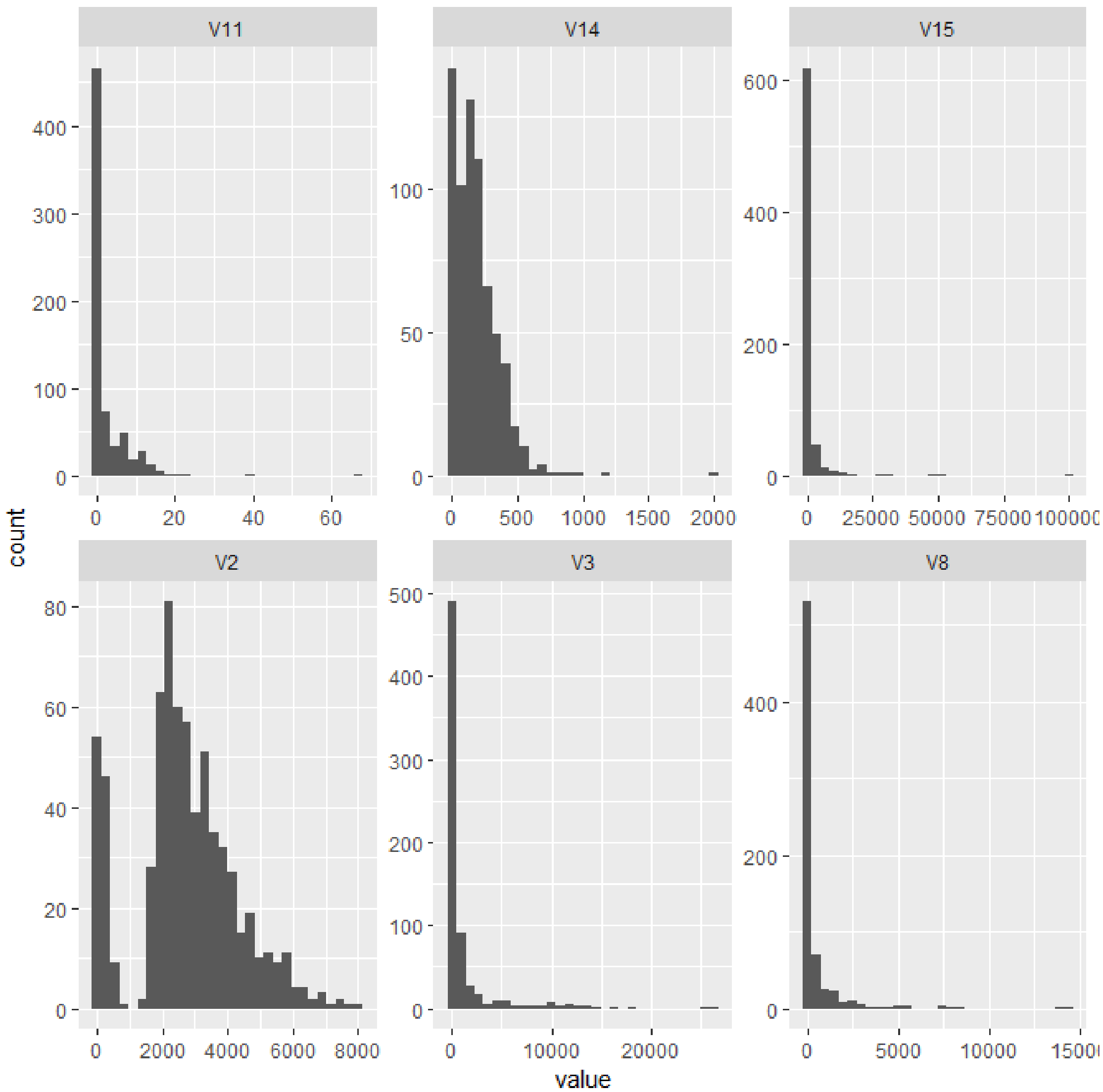



```
rm(p1,p2,p3,p4,p5,p6,p7,p8,p9,p10)
```

The interaction between two variables in each plot are able to split the class variable.for example,plot between v2 and v8 we can split and it creates two regions which are positive and negative class..This is also how a decision tree splits:)

Histogram Of Numerical Variables

```
df %>%keep(is.numeric)%>%gather() %>% ggplot(aes(value)) +facet_wrap(~ key,
scales = "free") +geom_histogram()
```



Other than V2 all the other numerical variables is skewed to the right

Check For Multicollinearity Between Numerical Variables

```
suppressMessages(library(pander))

## Warning: package 'pander' was built under R version 3.5.3

p <- sapply(df,function(x) is.numeric(x))

t <- df[,p]

#Remove Na's

q1 <- na.omit(t)
```

```r
c <-as.data.frame(cor(q1))

panderOptions('table.split.table', Inf)

pander(c)
```

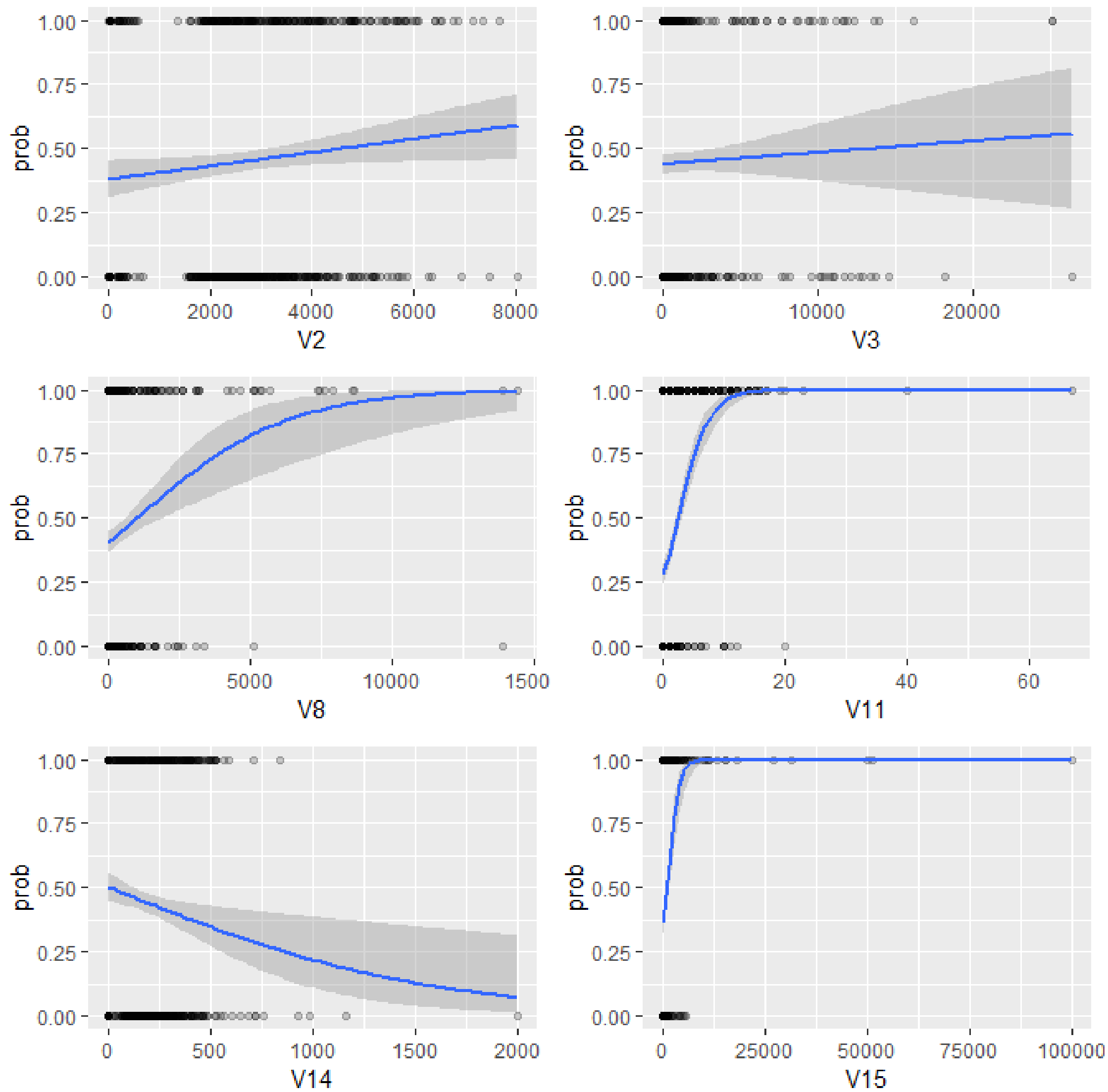
	V2	V3	V8	V11	V14	V15
V2	1	0.02141	0.07683	0.1237	-0.0292	0.007418
V3	0.02141	1	0.1496	-0.0119	-0.01216	-0.0343
V8	0.07683	0.1496	1	0.102	0.003972	-0.01808
V11	0.1237	-0.0119	0.102	1	-0.118	0.0596
V14	-0.0292	-0.01216	0.003972	-0.118	1	0.0694
V15	0.007418	-0.0343	-0.01808	0.0596	0.0694	1

```r
rm(t,q1,c)
```

No correlation exists between numerical variables

Checking Which Variable Is Able To Estimate The Logistic Curve

```r
df <- df %>% mutate(prob = ifelse(V16 == "POSITIVE", 1, 0))

q <- names(df[,p])

plots <- list()

for(i in q){

  s <-ggplot(data=df,aes_string(x=i,y="prob")) +geom_point(alpha = 0.2) +
  geom_smooth(method = "glm", method.args = list(family = "binomial"))

  plots[[i]] <- s
}

do.call("grid.arrange",plots)
```



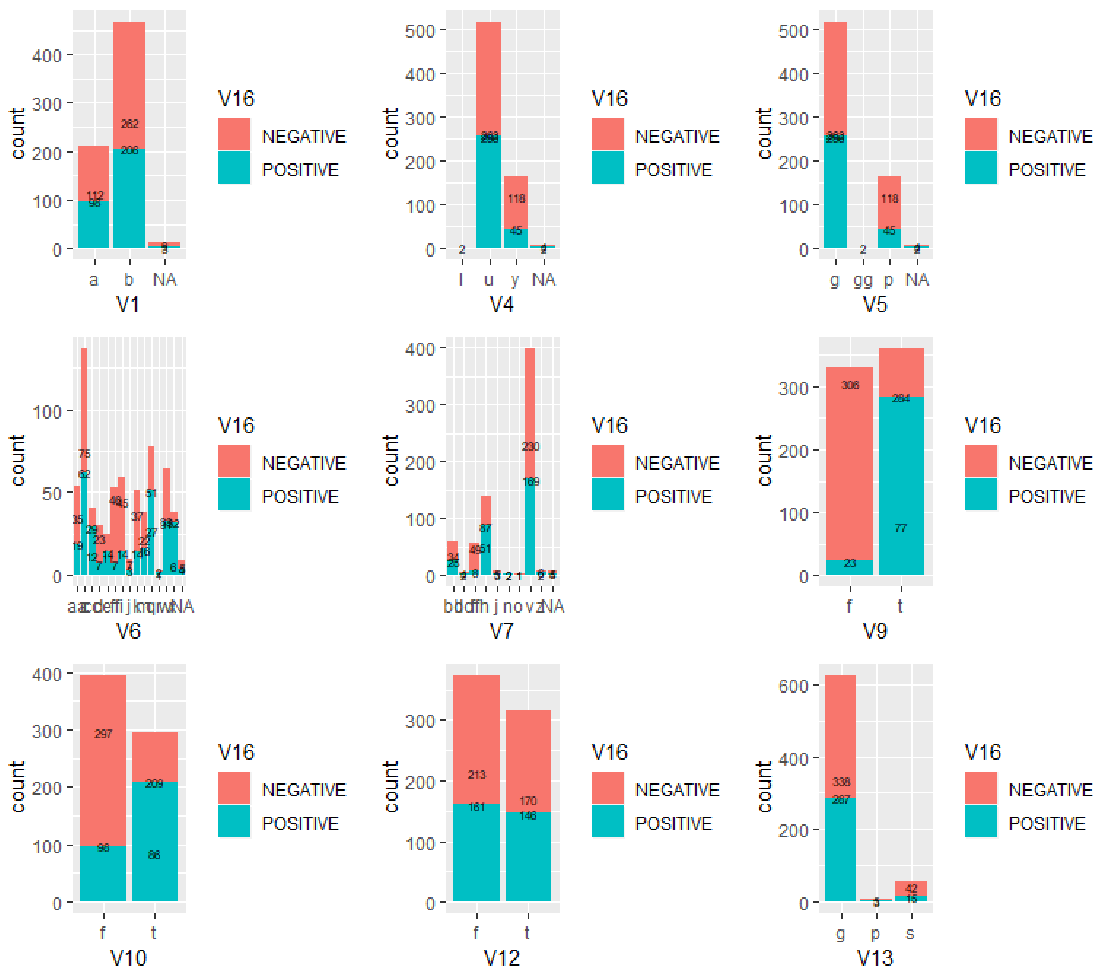

```
rm(d,q,plots)
```

V8,V11,V15 are able to estimate the logistic curve

Count Of Character Variables w.r.t Class

```
p <- sapply(df,function(x) is.character(x))

q <- names(df[,p])

plots <- list()

for(i in q){

  s <- ggplot(df,aes_string(x=i,fill="V16"))+geom_bar()
```

```
  d <- s+geom_text(stat='count', aes(label=..count..),size=2,vjust=0.5)

  plots[[i]] <- d
}

do.call("grid.arrange",plots)
```




```
rm(d,p,plots)
```

Variables like V9 and V10 are clearly able to distinguish class compared to other variables.Attributes in other variables are also able to distinguish class..For example,attribute “p” in V5.Likewise there could be individual attributes giving info..One more interesting thing is V4 and v5 have the same informarion so we can keep one of them

Check Balance Of Class

```
table(df$V16)
```

```
##
## NEGATIVE POSITIVE
##      383      307
```

Balance of class looks o.k

Derived Variables For Model Building

Converting to Dummy Vars

```r
# Remove NA's- Since NA's is 5% we can remove

library(dummies)

## dummies-1.5.6 provided by Decision Patterns

df_completed <- df[complete.cases(df),]

df_completed$V4 <- NULL

#Check balance after NA removal

table(df_completed$V16)

##
## NEGATIVE POSITIVE
##      357      296

# Create dummies

df_dummies <- dummy.data.frame(df_completed, names = q , sep = "_")

df_dummies <- data.frame(df_dummies)

rm(q)
```

After removal of NA's there is not much reduction of negative cases

Building a Logistic Model and Doing Feature Selection Also Checking The Vars We Found Important In Visualisation Are Present Or Not

```r
#MOdel-1

mod1 = suppressWarnings(glm(V16 ~.,family=binomial,data=df_dummies))

summary(mod1)

##
## Call:
## glm(formula = V16 ~ ., family = binomial, data = df_dummies)
```

```
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -2.409e-06  -2.409e-06  -2.409e-06   2.409e-06   2.409e-06
##
## Coefficients: (8 not defined because of singularities)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.657e+01  2.069e+05   0.000    1.000
## V1_a        -2.164e-08  3.267e+04   0.000    1.000
## V1_b                NA         NA      NA       NA
## V2           2.081e-13  9.775e+00   0.000    1.000
## V3          -7.670e-13  4.593e+00   0.000    1.000
## V5_g         2.494e-08  3.459e+04   0.000    1.000
## V5_gg       -4.435e-06  3.536e+05   0.000    1.000
## V5_p                NA         NA      NA       NA
## V6_aa       -4.535e-08  8.198e+04   0.000    1.000
## V6_c        -4.465e-08  7.040e+04   0.000    1.000
## V6_cc       -3.721e-08  8.326e+04   0.000    1.000
## V6_d        -5.563e-08  9.607e+04   0.000    1.000
## V6_e        -3.158e-08  1.301e+05   0.000    1.000
## V6_ff       -2.755e-08  2.041e+05   0.000    1.000
## V6_i        -5.559e-08  8.646e+04   0.000    1.000
## V6_j        -5.119e-08  2.179e+05   0.000    1.000
## V6_k        -4.860e-08  8.249e+04   0.000    1.000
## V6_m        -3.978e-08  8.804e+04   0.000    1.000
## V6_q        -3.430e-08  7.447e+04   0.000    1.000
## V6_r         2.996e-08  2.676e+05   0.000    1.000
## V6_w        -3.590e-08  7.743e+04   0.000    1.000
## V6_x                NA         NA      NA       NA
## V7_bb        2.615e-06  1.812e+05   0.000    1.000
## V7_dd        2.605e-06  1.970e+05   0.000    1.000
## V7_ff        2.591e-06  2.539e+05   0.000    1.000
## V7_h         2.628e-06  1.771e+05   0.000    1.000
## V7_j         2.640e-06  2.820e+05   0.000    1.000
## V7_n         2.590e-06  2.838e+05   0.000    1.000
## V7_o         2.632e-06  3.776e+05   0.000    1.000
## V7_v         2.615e-06  1.761e+05   0.000    1.000
## V7_z                NA         NA      NA       NA
## V8           2.633e-12  1.032e+01   0.000    1.000
## V9_f        -1.598e-08  4.389e+04   0.000    1.000
## V9_t                NA         NA      NA       NA
## V10_f        3.625e-08  3.807e+04   0.000    1.000
## V10_t               NA         NA      NA       NA
## V11          1.004e-08  3.584e+03   0.000    1.000
## V12_f        2.885e-08  2.925e+04   0.000    1.000
## V12_t               NA         NA      NA       NA
## V13_g        1.355e-08  5.561e+04   0.000    1.000
## V13_p       -3.870e-09  3.346e+05   0.000    1.000
## V13_s               NA         NA      NA       NA
## V14          2.650e-11  8.854e+01   0.000    1.000
```

```
## V15         -2.645e-12  3.824e+00   0.000    1.000
## prob         5.313e+01  4.557e+04   0.001    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8.9954e+02  on 652  degrees of freedom
## Residual deviance: 3.7884e-09  on 616  degrees of freedom
## AIC: 74
##
## Number of Fisher Scoring iterations: 25
```

```
#Do stepwise regression and select best parameters based on AIC

#backwards = step(mod1,trace=0)

#summary(backwards)

#Equation and Best Set of vars

#formula(backwards)

#rm(backwards,mod1)
```

We see from mod-1 that V9 is considered most significant variable as we had seen from exploration as well.The significant variables are set of variables which are able to distinguish the class.The AIC is reduced from 438.4 to 418.86.The set of important vars are V5,V6,V7,V8,V9,V11,V14,V15

Metrics to Determine How Good The Logistic model Is

```
#Load Packages

suppressMessages(library(caret))

## Warning: package 'caret' was built under R version 3.5.3

suppressMessages(library (caTools))

## Warning: package 'caTools' was built under R version 3.5.3

suppressMessages(library(e1071))

## Warning: package 'e1071' was built under R version 3.5.3

#Split the data into train and validation--To see how good the model performs
on new data#

set.seed(88)

split <- sample.split(df_dummies$V16, SplitRatio = 0.60)
```

```
df_dummies_train <-  subset(df_dummies, split == TRUE)

df_dummies_test <- subset(df_dummies, split == FALSE)

mod2 <- glm(V16~V5_g + V6_aa + V6_c + V6_d + V6_ff + V6_i + V6_j + V6_k +
    V6_m + V6_q + V6_w + V7_bb + V7_ff + V7_h + V7_j + V7_n +
    V7_v + V8 + V9_f + V11 + V14 +
V15,data=df_dummies_train,family="binomial")

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

predictions <- predict(mod2, type = 'response',df_dummies_test)

## Threshold as 0.5

predictions <- as.factor(ifelse(predictions>0.5,"POSITIVE","NEGATIVE"))

confusionMatrix(predictions,df_dummies_test$V16,positive="POSITIVE")

## Confusion Matrix and Statistics
##
##             Reference
## Prediction NEGATIVE POSITIVE
##   NEGATIVE      123        9
##   POSITIVE       20      109
##
##                Accuracy : 0.8889
##                  95% CI : (0.8443, 0.9243)
##     No Information Rate : 0.5479
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.7775
##  Mcnemar's Test P-Value : 0.06332
##
##             Sensitivity : 0.9237
##             Specificity : 0.8601
##          Pos Pred Value : 0.8450
##          Neg Pred Value : 0.9318
##              Prevalence : 0.4521
##          Detection Rate : 0.4176
##    Detection Prevalence : 0.4943
##       Balanced Accuracy : 0.8919
##
##        'Positive' Class : POSITIVE
##

rm(predicitons,df_dummies_train,df_dummies_test)

## Warning in rm(predicitons, df_dummies_train, df_dummies_test): object
## 'predicitons' not found
```

```r
rm(mod2)
```

** The accuracy is 0.89 with recall being 0.92and precision being 0.85 on validation data.The model has good metrics**

Buidling a RF Model And Also Checking The Aars We Found Important In Visualisation Are Present Or Not

```r
library(randomForest)

## Warning: package 'randomForest' was built under R version 3.5.3

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin

## Picking up number of trees to be small as the number of samples are less

model_rf1 <- randomForest(V16~.,data=df_dummies,ntree=50)

importance(model_rf1)

##         MeanDecreaseGini
## V1_a        5.328133e-01
## V1_b        4.965187e-01
## V2          4.469426e+00
## V3          5.125055e+00
## V5_g        1.200996e+00
## V5_gg       1.037143e+00
## V5_p        1.318279e+00
## V6_aa       2.365266e-01
## V6_c        6.647107e-01
## V6_cc       9.075319e-01
## V6_d        3.307829e-01
## V6_e        3.416397e-01
## V6_ff       1.229295e+00
## V6_i        8.509027e-01
```

```
## V6_j      1.308658e-02
## V6_k      1.208239e+00
## V6_m      3.056877e-01
## V6_q      7.948826e-01
## V6_r      1.155101e-02
## V6_w      9.145702e-01
## V6_x      1.339099e+00
## V7_bb     4.070587e-01
## V7_dd     5.642744e-05
## V7_ff     1.429837e+00
## V7_h      1.300750e+00
## V7_j      3.875018e-01
## V7_n      1.584635e-01
## V7_o      1.609052e-01
## V7_v      5.704764e-01
## V7_z      9.824082e-02
## V8        4.919615e+00
## V9_f      4.762881e+01
## V9_t      3.778370e+01
## V10_f     1.284609e+01
## V10_t     8.173960e+00
## V11       1.853811e+01
## V12_f     5.705682e-01
## V12_t     7.722931e-01
## V13_g     9.332619e-01
## V13_p     1.812552e-01
## V13_s     5.038225e-01
## V14       6.166060e+00
## V15       9.625197e+00
## prob      1.421033e+02
```

we could see from model_rf1 that variables V8,V9,V10,V11,V14,V15 have high importance.V2 and V3 have also come out to be important as we have seen in first plot how decision tree segments

Metrics to Determine How Good The RF model Is

```
predictions <- predict(model_rf1, type = 'response',df_dummies)

confusionMatrix(predictions,df_dummies$V16,positive="POSITIVE")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction NEGATIVE POSITIVE
##   NEGATIVE      357        0
##   POSITIVE        0      296
##
##                Accuracy : 1
##                  95% CI : (0.9944, 1)
##     No Information Rate : 0.5467
```

```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.4533
##          Detection Rate : 0.4533
##    Detection Prevalence : 0.4533
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : POSITIVE
##

rm(model_rf1,predictions)
```

RF provides an OOB estimate which is sample of data kept aside for validation.Hence we arent splitting into train and test..The metrics are better than logistic model

Q2 –> What are some ways to make sure the model that you create more robust to outliers?

- Outlier should be identified first through a boxplot approach or any clustering approach
- Once identified classifying and cleaning the datset becomes very important to remove an outlier without any pattern loss.
- You can create more samples of a good outlier using any sampling methodology in a training data set
- An outlier can have an impact on statistcal model like linear regression where it changes the slope of line so we can go for any tree based model like gbm which reduces the error

Q3 –> What could be some issues if the distribution of the test data is significantly different than the distribution of the training data?

- Both the data sets are not coming from same population
- Now if we test the model the model will not be able to understand the pattern as the observations in test data are new and it can cause overfitting or underfitting.
- More variance causes overfitting and less bias causes underfitting.So,It is important to give enough sample size which could be a representation of population size for training of the model.