```
setwd("C:\\Users\\Swapnil bandekar\\Downloads\\Swapnil\\Data Analytics\\My Work\\R\\Datasets")

getwd()



### Multi Co-linearity

# Co-relation between independent variables

# If value of VIF is high that means co-linearity is high

# Most commonly used value of VIF is 2

# If VIF > 2 , high multi co-linearity exist


## 3 conditions to finalize the model


# 1. The variables should be significant ( p value should be low )

# 2. Beta signs ( + or - ) should be significant ( in terms of business context )

# 3. VIF should be less than 2


## Linear Regression : Target variable is continuous , model will predict continuous values

## Logistics Regression : Target variable is binary , model will predict the probability


## Always use the variables which will useful later on ( when we apply on the population )

## Loan amount or loan duration can't be used in the model as will not be knowing the values whenever we will use this
#  model on population


## Linear Regression Equation


# S = B0 + B1X1 + B2X2 + B3X3


## Logistics Regression Equation
```

```
# P = exp(S) {N} / [ 1 + exp(S) ] {D}  ( initial equation )


# N => 0 to infinity

# D => 1 to infinity

# D >= N Always

# N/D => 0 to 1

## Solving the Initial Equation

# P = exp(S) / [ 1 + exp(S) ]


# 1/P = [ 1 + exp(S) ] / exp(S)

# 1/P = 1/exp(S) + 1

# 1/P - 1 = 1/exp(S)

# (1-P)/P = 1/exp(S)

# exp(S) = P/(1-P)

## Taking Log

# log [ exp(S) ] = log(P/1-P)

# log(P/1-P) = S


### Log(P/1-P) = S =  B0 + B1X1 + B2X2 + B3X3  ( final equation )


# If B is positive => Probability is higher
# If B is negative => Probability is lower

# 1 unit change in X1 will change the log odds of P by B1.



# Future Information can't be used in the model e.g. EMI amount paid in 1st 6
months

# Performance window variables can't be used in the model
```

```
# Only observation window variables can be used in the model


## Observation window : Variables available at the time of application of the model



### In logistics Regression model , model will always predict the probability of 1
( Always )


## Divide the data into 2 parts ( Train and Test )

# Train : 70% ( to build the model )

# Test : 30% ( to test the model )


## 2 types of validation


## 1. In time validation

# Train and Test datasets are chosen randomly

# Suppose we have 1000 data points from (2005 -2008) ,  we will split the data as
700 data points in train dataset and
# 300 data points in test dataset

## 2. Out Time Validation

# Train and Test datasets are chosen randomly

# Suppose we have 1000 data points from (2005 -2008) ,  we will split the data as
700 data points in train dataset and
# 300 data points in test dataset

# And Out of Time validation is done of data points from 2009



## Concept of Train and Test is also done in case of Linear Regression

# It is done to check if the model is good or the model is overfitting or
underfitting with the help of Ginny values


# Model Type       Train    Test      Bias      Variance
```

```
# Overfitting       50       30         Low         High

# Underfitting      28       27         High        Low

# Good              38       38         Right       Right


## Overfitting

# Model in running smoothly on Train Data but failing big time on the Test Data
then it is called as overfitting

# All variables are used in the model blindly



## Underfitting

# Model in not running smoothly on Train Data but doing the same on the Test Data
then it is called as Underfitting
# ( Model performacne not at par )

# Only 1 or 2 variables are used in the model



## Good Model

# Model is running smoothly on Train as well as Test Data

# All relevant variables are used in the model



GoodBad <- read.csv("GOODBAD.CSV")

View(GoodBad)

str(GoodBad)


# 1 : Good , 2 : Bad

table( GoodBad$Good.Bad )


# Logistics Regression Model Predicts the Probability of 1 ( always )
# Here , we want to predict Probability of Default . Hence , converting the
Good.Bad variable
```

```
GoodBad$Good.Bad <- 1 - GoodBad$Good.Bad

table( GoodBad$Good.Bad )


plot( GoodBad$Good.Bad )

dim(GoodBad)


## Checking for missing values

colSums( is.na( GoodBad ))


Sampling <- sort( sample( nrow( GoodBad ) , nrow( GoodBad )*0.7 ))

length( Sampling )


## Select Training Sample

Train <- GoodBad [ Sampling , ]
Test <- GoodBad [ -Sampling , ]

nrow( Train )
nrow( Test )


table( Train$Good.Bad ) / 700

table( Test$Good.Bad ) / 300


table( Train$Good.Bad , Train$Check_Account_Status ) / 700


#---------------------------------------Logistics Regression
Modelling-----------------------------------------------


## Iteration 1
```

```
attach(Train)

MyResult <- glm( data = Train , formula = Good.Bad~Check_Account_Status , family =
binomial )

summary( MyResult )



## Iteration 2

MyResult <- glm( formula = Good.Bad ~ CreditHistory + Check_Account_Status , family
= binomial , data = Train )

summary( MyResult )



## Iteration 3


MyResult <- glm( formula = Good.Bad ~ CreditHistory + Check_Account_Status +
Duration , family = binomial , data = Train )

summary( MyResult )



## Iteration 4


MyResult <- glm( formula = Good.Bad ~ CreditHistory + Check_Account_Status +
Duration + Purpose , family = binomial , data = Train )

summary( MyResult )


# Choosing a cut-off of 1% and checking which variable is significant enough to be
used in the model

# CreditHistoryA34 , Check_Account_StatusA14 , Duration , PurposeA41 , PurposeA43
are significant variables

# creating dummy variables for significant variables


Train$CreditHistoryA34 <- ifelse( Train$CreditHistory == "A34" , 1 , 0 )

Train$PurposeA41 <- ifelse( Train$Purpose == "A41" , 1 , 0 )
```

```
Train$PurposeA43 <- ifelse( Train$Purpose == "A43" , 1 , 0 )

Train$Check_Account_StatusA14 <- ifelse( Train$Check_Account_Status == "A14" , 1 ,
0 )


MyResult1 <- glm( formula = Good.Bad ~ Check_Account_StatusA14 + CreditHistoryA34 +
Duration + PurposeA41 + PurposeA43 , family = binomial , data = Train )

summary(MyResult1)

View(Train)
View(Test)

Test1 <- Test

View(Test1)
View(Test)

Test1$CreditHistoryA34 <- ifelse( Test1$CreditHistory == "A34" , 1 , 0 )

Test1$PurposeA41 <- ifelse( Test1$Purpose == "A41" , 1 , 0 )

Test1$PurposeA43 <- ifelse( Test1$Purpose == "A43" , 1 , 0 )

Test1$Check_Account_StatusA14 <- ifelse( Test1$Check_Account_Status == "A14" , 1 ,
0 )


## Finding the predicted values


Predicted <- MyResult1$fitted.values

head(Predicted)



Predicted1 <- predict( MyResult1 , data = Test , type = "response" )


Predicted2 <- predict( MyResult1 , data = Test1 , type = "response" )


# predict command is used to predict the result

# type = "response" => to get the probability


## Confusion Matrix
```

```
Predbkt1 <- ifelse ( Predicted1 > 0.50 , 'D' , 'A' )


table( Predbkt1 , Train$Good.Bad )



## Plotting the ROC Curve

library( ROCR )


# The prediction function of the ROCR library basically creates a structure to
validate
# Our predictions with actual values


Pred <- prediction( Predicted1 , Train$Good.Bad )


# "tpr" and "fpr" are arguments of the performance function indicating that the
plot is between True Positive Rate and
#  False Positive Rate

Perf <- performance( Pred , "tpr" , "fpr" )

plot( Perf , col = "red" )

abline( 0 , 1 , lty = 8 , col = "grey" )


# True Positive Rate = tpr = True Positive / No of Positives = 72 / ( 72 + 142 )
# We are predicting porbability of default ; hence True Positive = 72 ( No of
defaulters predicted by the model who have
#                                                                        actually
defaulted in the past )
# No of Positive = 72 + 142 ( total no of defaulters from the past)

# False Positive Rate = fpr = False Positive / No of Negatives = 40 / ( 40 + 454 )
# False Positive = 40 ( No of defaulters predicted by the model who are actually
good customers )
# No of Negative = 40 + 454 ( total no of good customers )

# Performance command is used to find out tpr and fpr

## Bigger the area between Diagonal line and curve better is the model

## AUC : Area Under Curve
```

```
## How to choose Cutoff's ??

# Use @ to access the slots


Cutoffs <- data.frame( cut = Perf@alpha.values[[1]] , fpr = Perf@x.values[[1]] ,
tpr = Perf@y.values[[1]] )

head(Cutoffs)


Cutoffs <- Cutoffs[ order( Cutoffs$tpr , decreasing = TRUE ) , ]

head(Cutoffs)




AUC <- performance( Pred , "auc" )

AUC <- unlist( slot( AUC , "y.values"))

AUC


## Ginny = 2*AUC - 1


## Model Acceptance Crieteria

# Generally people use cutoff of 40% ; if Ginny > 40% , model is good

# Model which is having the higher Ginny value should be approved , but Ginny value
should hold for both Train and Test

# Domain                  Ginny

# Application Models      >35% ( Loan )

# Marketing               >45%

# Behavioural             >50% ( People scorecard of existing customers )


## To choose a Good Model
```

```
Reduced<-step( MyResult1 ,direction="backward" )

# based on the above code, PurposeA43 had low AIC, so run a model with only tht
variable

MyResult3 <- glm( data = Train , Good.Bad ~ PurposeA43 , family=binomial )

summary( MyResult3 )



#####-------------------Summary-------------------


# More frequently used Regerssion

# Target variable is Categorical / Binary

# Gives the probability of Target Variable

# Gives the probability of 1

# Only 1 Target variable is present

# Hypothesis on independent variable which can affect the Target Variable

# Percentile of independent variable should be considered

# EDA ( Exploratory Data Analysis )

# Univariate Analysis on independent variables

# Dummy variable creation for categorical variables

# Never Impute values of Target Variable ( No missing value Treatment on Target
Variable )

# Bivariate Analysis ( Independent vs Target Variable )

# Transformation of independent variable if required

# Model Iterations

# Remove the variable with high P values

# Variables Shortlisting

# As long as VIF is higher than the Cut off ; remove 1 variable at a time and run
the model again until the VIF is lower
```

```
# than the cut off

# Model should contain Max 10 variables

# All variables should be unique in nature

# 70 : 30 Rule Train : Test


### Suppose that Amazon wants to do the online campaign for the customers who have not bought from amazon for last 1 year
#   from the entire pool of customers and want to identify the possible customers who can buy with Amazon in near future.
#   This is case logistics regression problem but we do not have Target variable at on today (Nov'19).
#   So we will extract the data of customers who have not bought from Amazon between Jun'18 to Jun'19 and map the same with
#   latest data ; we will find that most of them are present in both the data we will mark them as 0 (who have not Bought )
#   and there must be some customers who have bought from Jun'19 to Nov'19 we will mark as them as 1.
#   We will target customers who have whistlisted the products , who are visting the app , who have not raised customer
#   complaints....
```