Prediction of Diabetes in Women

By Aparajita Mukherjee

Table of Content

1. Project Objective

The Objective of this project is to –

1. Plot graphs between Dependent and Independent Variables.
2. Split the data into in-sample(train) and out-of-sample(test) – 70%-30%.
3. Build models using – Naïve Bayes, K-NN, Logistic Regression on train dataset.
4. Validate the model on the test dataset(to check for overfitting).
5. Calculate accuracy using Confusion Matrix.
6. Compare the results and draw inferences.

2. Details of the Project

A total of 9 variables were used for segregating the cases where there were occurrence of diabetes as compared to cases where there were none.

Variable Name	Data Type	Variable Description
NoPreg	integer	Number of time pregnant
PlaGluConc	integer	Plasma glucose concentration a 2 hours in an oral glucose tolerance test
DiastolicBP	integer	Diastolic blood pressure (mm Hg)
TSkinThick	integer	Triceps skin fold thickness (mm)
Test	integer	2-Hour serum insulin (mu U/ml)
BMI	numeric	Body mass index (weight in kg/(height in m)^2)
DiabPediFunc	numeric	Diabetes pedigree function
Age	integer	Age (years)
Class	integer	Class variable (0 or 1)

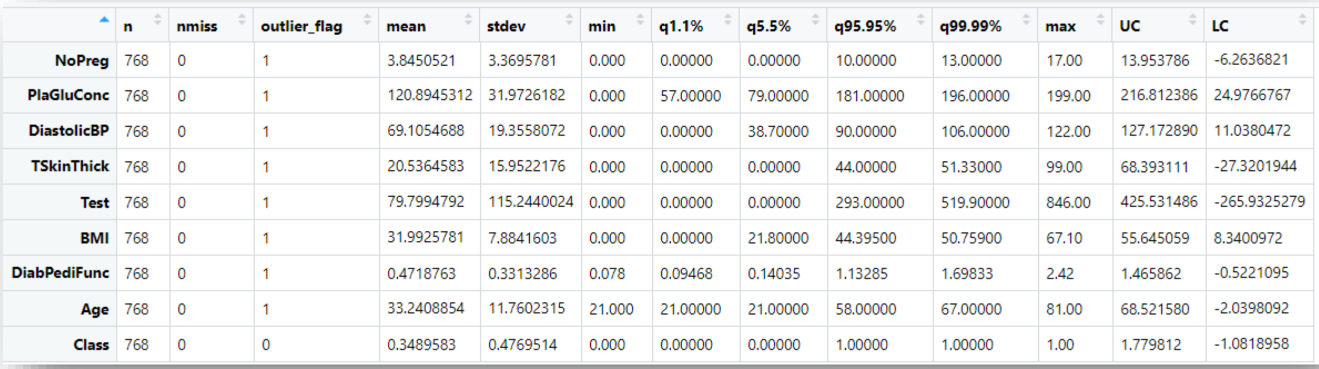
The datatype of the Class variable(target variable) requires to be changed to factor before using it as input to the logistic regression model.

Also, the dataset needs to be normalized before using it to build the KNN(K-Nearest Neighbour model and Naïve Bayes.

3. Required Packages

Library	Description
library(ggplot2)	Create Data Visualisations
library(car)	Companion to Applied Regression
library(caret)	Classification and Regression Tree
library(class)	Functions for Classification
library(devtools)	Tools to make developing R packages easier
library(e1071)	Misc Functions of Statistics(Naïve Bayes)
library(lmtest)	Testing Linear Regression Models
library(Hmisc)	Get the detailed summary of the dataset
library(ROCR)	Visualizing the Performance of Scoring Classifiers
library(plyr)	Tools for Splitting, Applying & Combining Data
library(pROC)	Display & Analyse ROC curve
library(psych)	Procedures for Psychometric and Personality Research
library(dplyr)	A Grammar for Data Manipulation
library(corrplot)	Visualization of Correlation Matrix
library(caTools)	Splitting the Dataset
library(DataExplorer)	Automate Data Exploration and Treatment

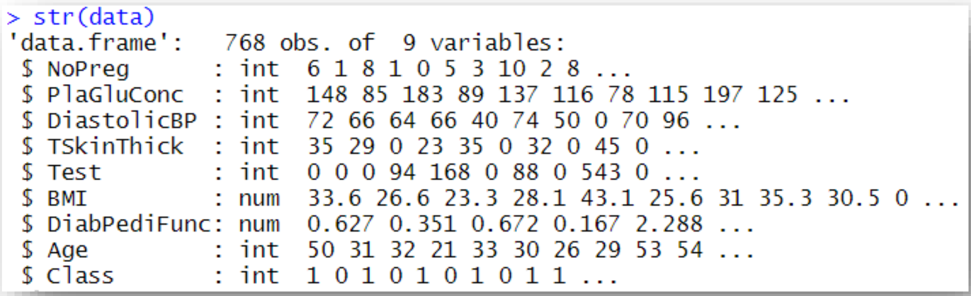
4. Basic EDA(Exploratory Data Analysis)



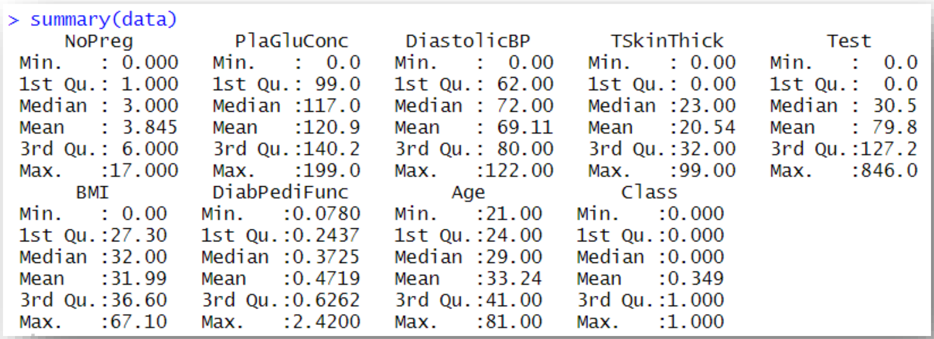
	n	nmiss	outlier_flag	mean	stdev	min	q1.1%	q5.5%	q95.95%	q99.99%	max	UC	LC
NoPreg	768	0	1	3.8450521	3.3695781	0.000	0.00000	0.00000	10.00000	13.00000	17.00	13.953786	-6.2636821
PlaGluConc	768	0	1	120.8945312	31.9726182	0.000	57.00000	79.00000	181.00000	196.00000	199.00	216.812386	24.9766767
DiastolicBP	768	0	1	69.1054688	19.3558072	0.000	0.00000	38.70000	90.00000	106.00000	122.00	127.172890	11.0380472
TSkinThick	768	0	1	20.5364583	15.9522176	0.000	0.00000	0.00000	44.00000	51.33000	99.00	68.393111	-27.3201944
Test	768	0	1	79.7994792	115.2440024	0.000	0.00000	0.00000	293.00000	519.90000	846.00	425.531486	-265.9325279
BMI	768	0	1	31.9925781	7.8841603	0.000	0.00000	21.80000	44.39500	50.75900	67.10	55.645059	8.3400972
DiabPediFunc	768	0	1	0.4718763	0.3313286	0.078	0.09468	0.14035	1.13285	1.69833	2.42	1.465862	-0.5221095
Age	768	0	1	33.2408854	11.7602315	21.000	21.00000	21.00000	58.00000	67.00000	81.00	68.521580	-2.0398092
Class	768	0	0	0.3489583	0.4769514	0.000	0.00000	0.00000	1.00000	1.00000	1.00	1.779812	-1.0818958

As is clear from the above table that –

- There are 9 variables having 768 observations.
- There are no missing values.
- All the 8 variables except for the target variable, have outliers.



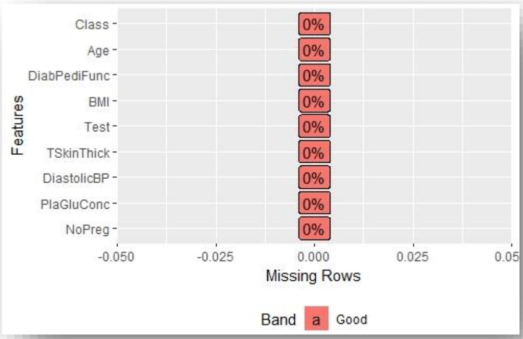
```
> str(data)
'data.frame':   768 obs. of  9 variables:
 $ NoPreg      : int  6 1 8 1 0 5 3 10 2 8 ...
 $ PlaGluConc  : int  148 85 183 89 137 116 78 115 197 125 ...
 $ DiastolicBP : int  72 66 64 66 40 74 50 0 70 96 ...
 $ TSkinThick  : int  35 29 0 23 35 0 32 0 45 0 ...
 $ Test        : int  0 0 0 94 168 0 88 0 543 0 ...
 $ BMI         : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ DiabPediFunc: num  0.627 0.351 0.672 0.167 2.288 ...
 $ Age         : int  50 31 32 21 33 30 26 29 53 54 ...
 $ Class       : int  1 0 1 0 1 0 1 0 1 1 ...
```



```
> summary(data)
     NoPreg        PlaGluConc      DiastolicBP      TSkinThick         Test      
 Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00   Min.   :  0.0  
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00   1st Qu.:  0.0  
 Median : 3.000   Median :117.0   Median : 72.00   Median :23.00   Median : 30.5  
 Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54   Mean   : 79.8  
 3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00   3rd Qu.:127.2  
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00   Max.   :846.0  
      BMI        DiabPediFunc         Age            Class      
 Min.   : 0.00   Min.   :0.0780   Min.   :21.00   Min.   :0.000  
 1st Qu.:27.30   1st Qu.:0.2437   1st Qu.:24.00   1st Qu.:0.000  
 Median :32.00   Median :0.3725   Median :29.00   Median :0.000  
 Mean   :31.99   Mean   :0.4719   Mean   :33.24   Mean   :0.349  
 3rd Qu.:36.60   3rd Qu.:0.6262   3rd Qu.:41.00   3rd Qu.:1.000  
 Max.   :67.10   Max.   :2.4200   Max.   :81.00   Max.   :1.000  
```

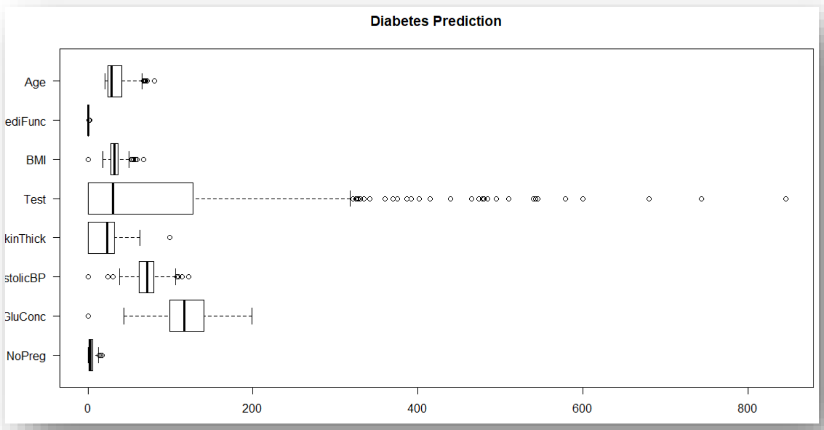
- There are 9 variables having 768 observations.
- There are evidences of outliers in the dataset which will be clear after plotting the same through a boxplot.

4.1 Missing Values



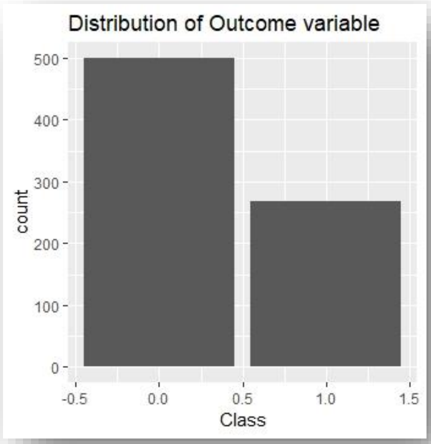
Missing values in the dataset can lower the accuracy rate of the model. It is therefore necessary to either remove the missing values(if the percentage of missing values is very small) or replace them with the Mean, Median or Mode of the particular variable. From the above plot it is evident that there are no missing values in the dataset and it is good to start working with.

4.2 Outliers



The plot indicates that there are outliers in all the independent variables of the dataset. But the algorithms that will be used will not be significantly affected by the presence of outliers. In case, necessary, we will cap the outlier values to the 99th quantile of that particular variable.

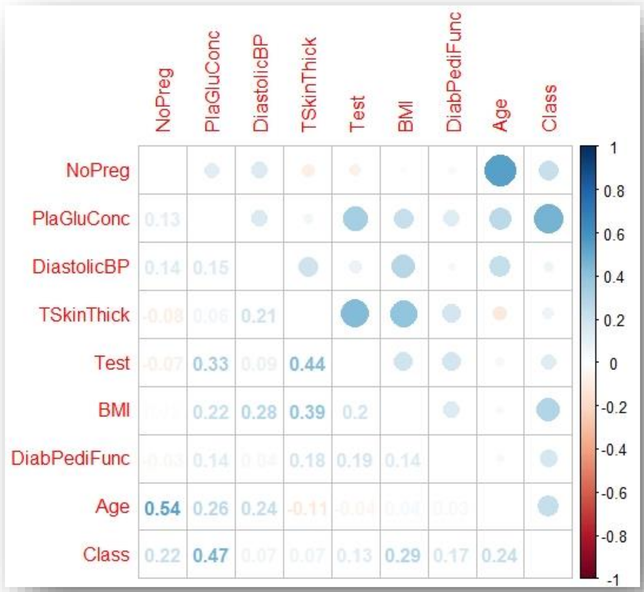
Diabetes	Count	Percentage
0	500	65%
1	268	35%
Total	768	100%



The number of women detected to be suffering from diabetes is very less as compared to women who were not suffering from it.

4.3 Correlation

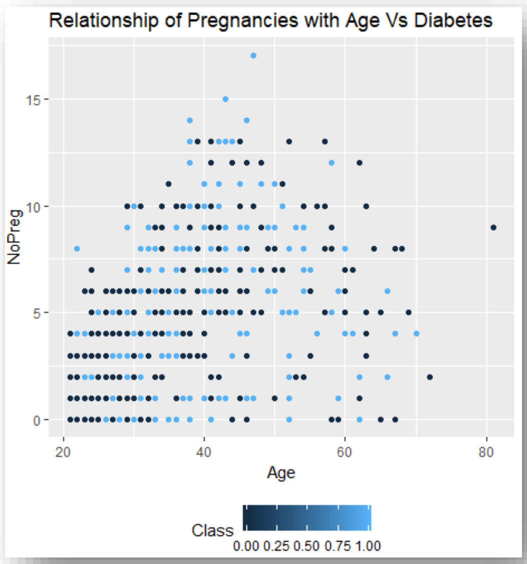
It is evident that there is not much correlation between the variables. This wouldn't affect the accuracy of the model.

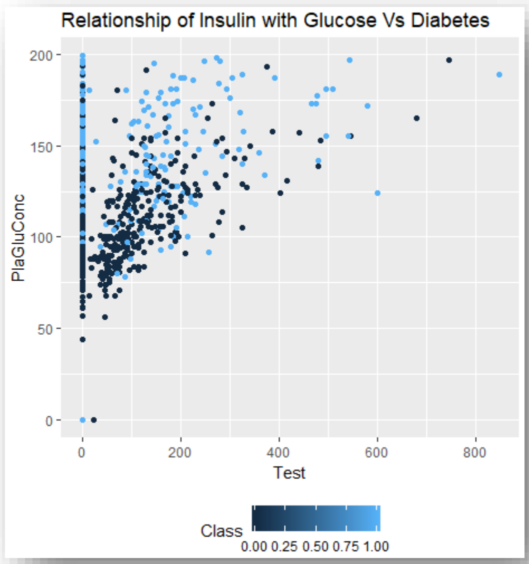


- There is no obvious relationship between age and onset of diabetes.
- There is no obvious relationship between pedi function and onset of diabetes.
- This may suggest that diabetes is not hereditary, or that the Diabetes Pedigree Function needs work.
- Variables like PleGluConc, NoPreg, BMI, Age, DiabPediFunc and Test have a positive relation with the target variable(Class). This means whenever the value for these variables increases, the minority class also increases.
- Most of the variables are positively related.
-
- Larger values of plas combined with larger values for age, pedi, mass, insu, skin, pres, and preg tends to show greater likelihood of testing positive for diabetes.

4.4 Visualization(Independent vs Dependent)

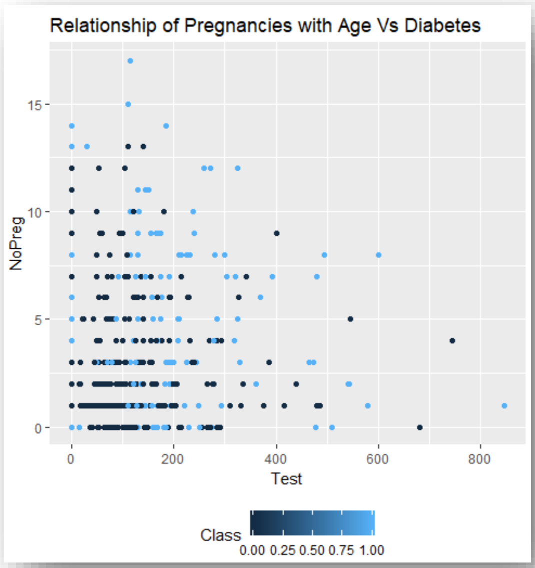
4.4.1 Scatter Plot

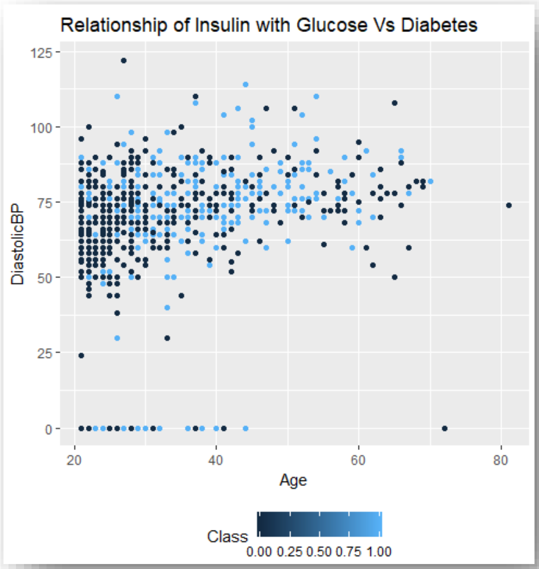




No clear boundary can be drawn that separates Non-diabetic and Diabetic women based on Number of Pregnancies vs Age

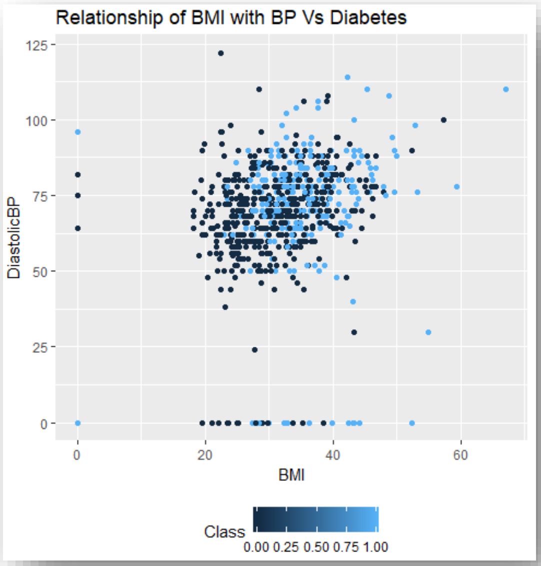
Non-diabetic women seemed to have lower levels of Insulin and Glucose as opposed to Diabetic women who recorded low to high levels of Insulin and high levels of Glucose

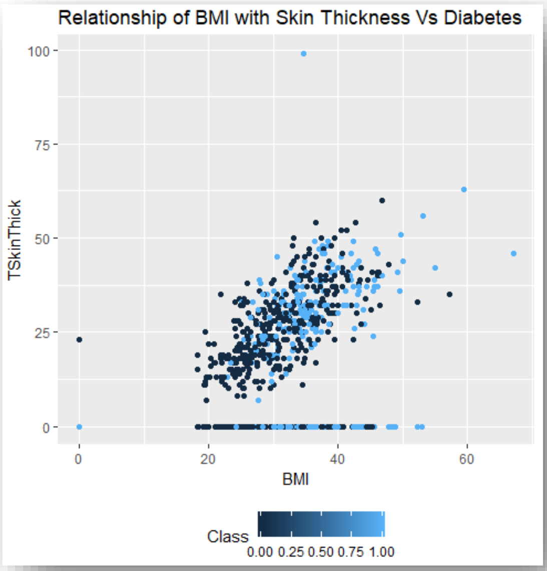




There is no significant distinction based on Insulin level and Pregnancies, but diabetic women seem to have a slightly increased insulin and the Pregnancies.

Non-diabetic women seemed to have lower BP levels and Age as opposed to Diabetic women who recorded low to high levels of BP and at a later Age.

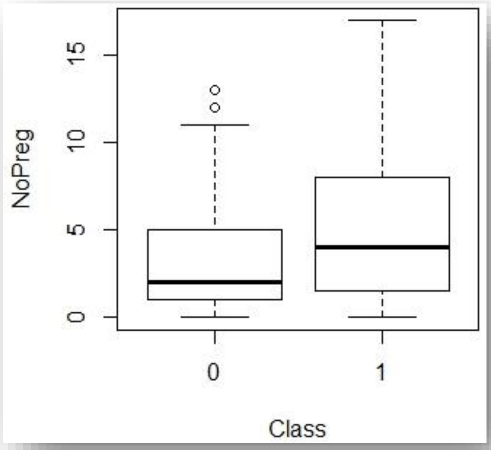


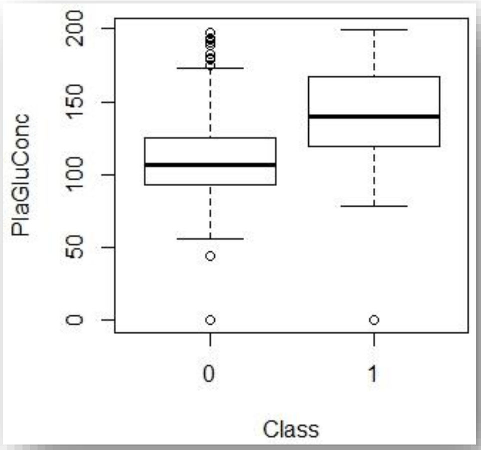


Women who have Diabetes can be differentiated from those who don't have based on BMI and BP values

Women with low values of BMI and Skin Thickness did not have Diabetes

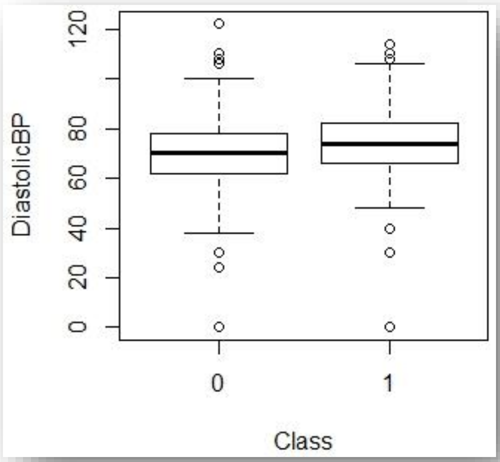
4.4.2 Boxplot

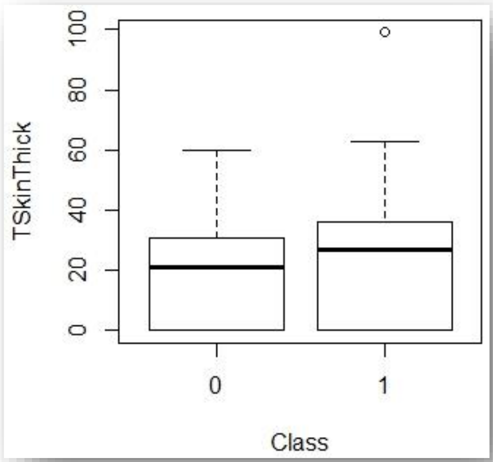




Higher the number of times a women was pregnant, she will have more probability to be diagnosed with Diabetes. This has a lot of data to the right side of the median as the 1st quartile is more than the 3rd quartile.

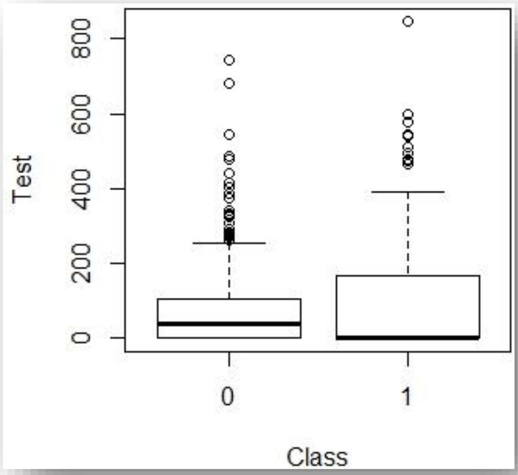
Higher the number of times a wo

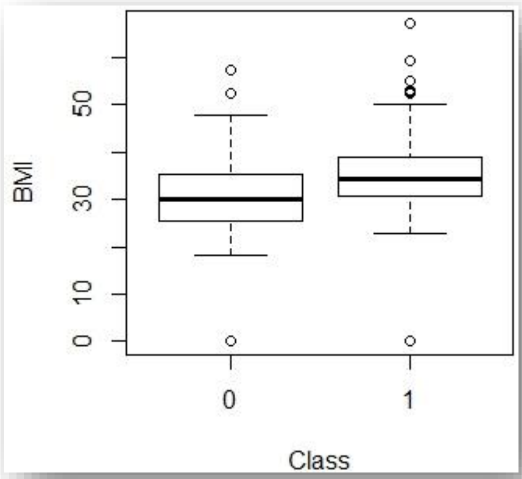




The boxplot looks almost symmetric and relatively uniform. But a woman with marginally higher Diastolic Blood Pressure will be suffering from diabetes.

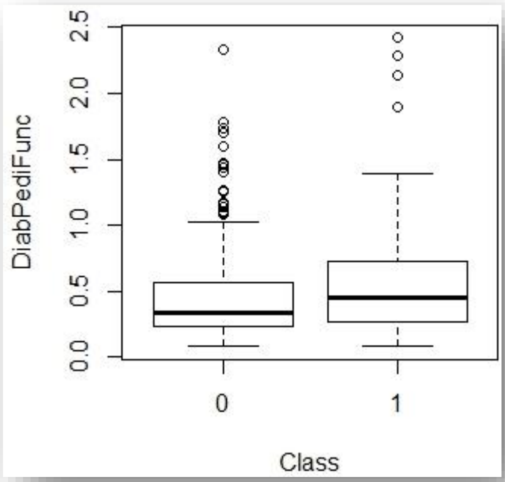
The value of the 1st quartile for the variable – TskinThick is "0" and therefore, the boxplot starts at the same point. It also indicates that women with marginally higher Triceps Skin Fold Thickness tend to suffer from diabetes.

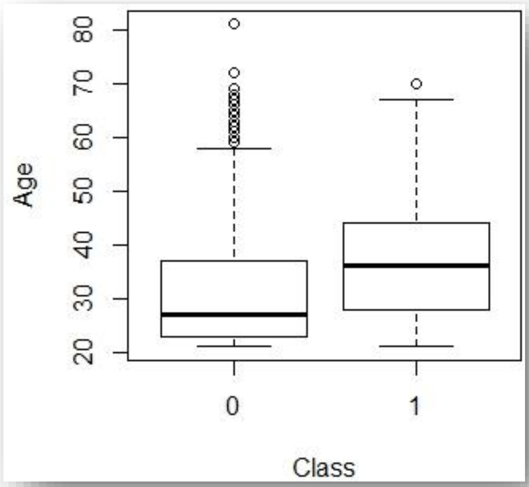




As was seen in the summary, the minimum and the value for the first quartile for the variable – Test starts from "0", especially for the women who were detected with the diabetes.

Shows that all the women who had Diabetes had a BMI greater than 25, which is above the normal levels. On the other hand, women who did not have Diabetes had a BMI ranging from 18 to 60.





Women with marginally higher amount of Diabetes Pedigree Function are more susceptible to having diabetes.

This boxplot indicates that higher age is a major factor contributing to the cause of diabetes in women.

5. Model Building

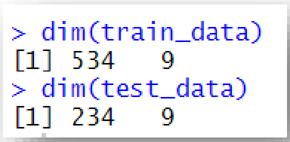
Algorithms used –

- Logistic Regression
- K-Nearest Neighbour
- Naïve Bayes

These algorithms are relevant because they perform classification on a dataset, deal appropriately with missing or erroneous data.

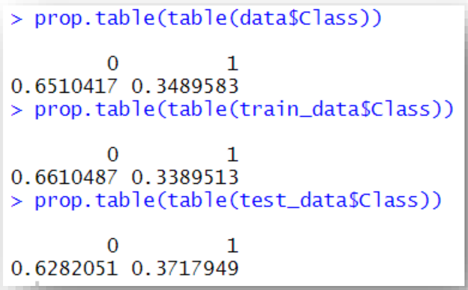
5.1 Split the Data

Before using the dataset for building and evaluating the models, it is advisable to split the data. The default ratio to split is usually a 70-30 spit.



```
> dim(train_data)
[1] 534   9
> dim(test_data)
[1] 234   9
```

We want to ensure that the proportion of overall class variable is as relative as it was in the original dataset and the proportion gets retained in the train and test dataset.

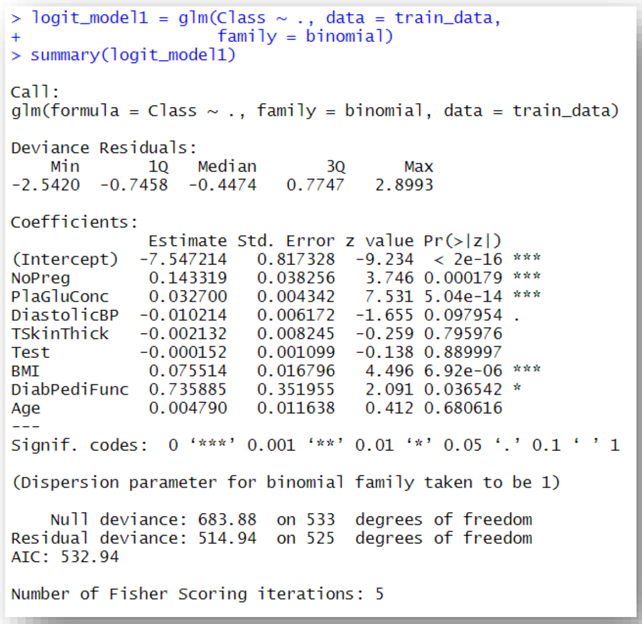


```
> prop.table(table(data$Class))

        0         1 
0.6510417 0.3489583 
> prop.table(table(train_data$Class))

        0         1 
0.6610487 0.3389513 
> prop.table(table(test_data$Class))

        0         1 
0.6282051 0.3717949 
```

5.2 Logistic Regression

5.2.1 Model Building

A full model was built with Class as the response variable with the rest of the 8 predictor variables. Step-wise variable selection method was used to identify the most important variables. The final model chosen with AIC as the criterion for selection generated a logistic regression model with the lowest AIC value of 532.94 as below.



```
> logit_model1 = glm(Class ~ ., data = train_data,
+                    family = binomial)
> summary(logit_model1)

Call:
glm(formula = Class ~ ., family = binomial, data = train_data)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.5420  -0.7458  -0.4474   0.7747   2.8993  

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept)  -7.547214   0.817328  -9.234  < 2e-16 ***
NoPreg        0.143319   0.038256   3.746 0.000179 ***
PlaGluConc    0.032700   0.004342   7.531 5.04e-14 ***
DiastolicBP  -0.010214   0.006172  -1.655 0.097954 .  
TSkinThick   -0.002132   0.008245  -0.259 0.795976    
Test         -0.000152   0.001099  -0.138 0.889997    
BMI           0.075514   0.016796   4.496 6.92e-06 ***
DiabPediFunc  0.735885   0.351955   2.091 0.036542 *  
Age           0.004790   0.011638   0.412 0.680616    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 683.88  on 533  degrees of freedom
Residual deviance: 514.94  on 525  degrees of freedom
AIC: 532.94

Number of Fisher Scoring iterations: 5
```

5.2.2 Interpretation

Factors to be checked –

- Variable Significance/Insignificance
- AIC(Akaike information criterion)
- Fisher Scoring Iterations

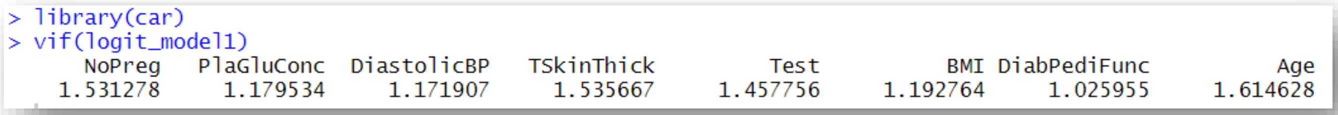
The variables – TskinThick, Test and Age seem to be statistically insignificant.

AIC(Akaike information criterion) - is an estimator of the relative quality of statistical models for a given set of data. It is another measure of goodness of fit that takes into account the ability of the model to fit the data. Thus, AIC provides a means for model selection. The lower the AIC, the better is the model.

Fisher scoring iterations uses an iterative approach (the Newton-Raphson algorithm by default) that looks for the best model. The algorithm stops when it doesn't perceive that moving again would yield much additional improvement. This line tells you how many iterations there were before the process stopped and output the results.

5.2.3 Multicollinearity Check

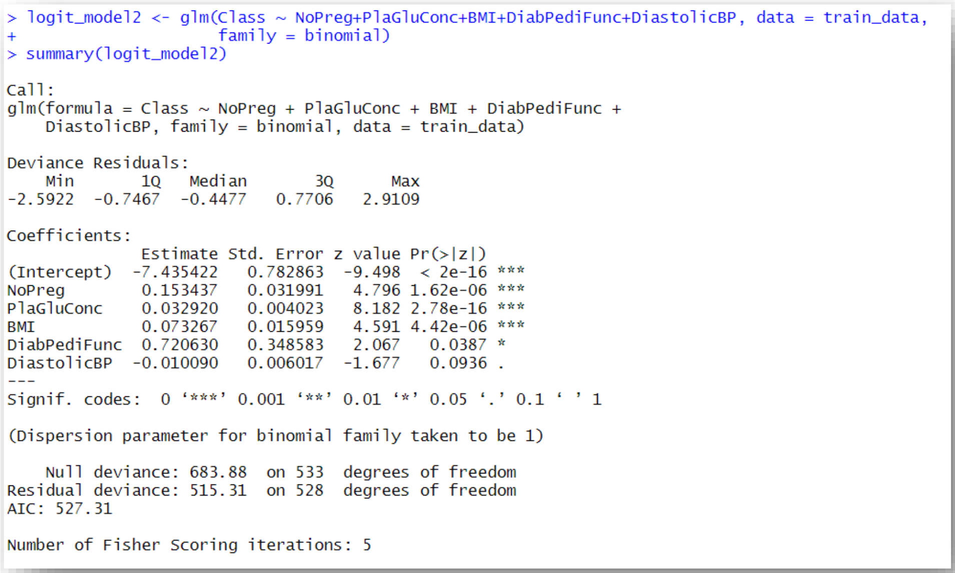
VIF(Variable Inflation Factor) – measures how much the variance of a regression coefficient is inflated due to multicollinearity in the model. It helps solve the problem of Multicollinearity(implies that the information that this variable provides about the response is redundant in the presence of the other variables).



```
> library(car)
> vif(logit_model1)
    NoPreg   PlaGluConc   DiastolicBP   TSkinThick          Test           BMI  DiabPediFunc           Age
  1.531278     1.179534      1.171907     1.535667      1.457756      1.192764      1.025955      1.614628
```

The result indicated that the variables aren't correlated to each other. This will help in building a model with good accuracy.

5.2.4 Improving the Model(AIC)



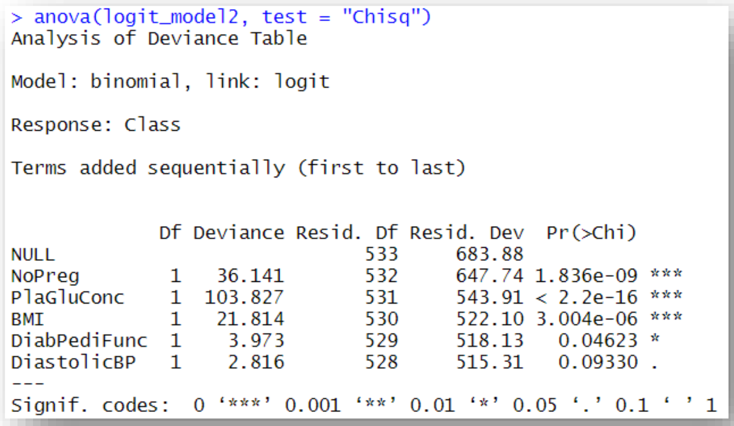
```
> logit_model2 <- glm(Class ~ NoPreg+PlaGluConc+BMI+DiabPediFunc+DiastolicBP, data = train_data,
+                      family = binomial)
> summary(logit_model2)

Call:
glm(formula = Class ~ NoPreg + PlaGluConc + BMI + DiabPediFunc + 
    DiastolicBP, family = binomial, data = train_data)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.5922  -0.7467  -0.4477   0.7706   2.9109  

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept)  -7.435422   0.782863  -9.498  < 2e-16 ***
NoPreg        0.153437   0.031991   4.796 1.62e-06 ***
PlaGluConc    0.032920   0.004023   8.182 2.78e-16 ***
BMI           0.073267   0.015959   4.591 4.42e-06 ***
DiabPediFunc  0.720630   0.348583   2.067   0.0387 *  
DiastolicBP  -0.010090   0.006017  -1.677   0.0936 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 683.88  on 533  degrees of freedom
Residual deviance: 515.31  on 528  degrees of freedom
AIC: 527.31

Number of Fisher Scoring iterations: 5
```

The AIC after removing the statistically insignificant variables is 527.31 which is better than the previous 532.94.

```
> step_model <- step(logit_model1)
Start:  AIC=532.94
Class ~ NoPreg + PlaGluConc + DiastolicBP + TSkinThick + Test +
    BMI + DiabPediFunc + Age

               Df Deviance    AIC
- Test          1   514.96 530.96
- TSkinThick    1   515.00 531.00
- Age           1   515.11 531.11
<none>              514.94 532.94
- DiastolicBP   1   517.69 533.69
- DiabPediFunc  1   519.38 535.38
- NoPreg        1   529.65 545.65
- BMI           1   537.30 553.30
- PlaGluConc    1   584.52 600.52

Step:  AIC=530.96
Class ~ NoPreg + PlaGluConc + DiastolicBP + TSkinThick + BMI +
    DiabPediFunc + Age

               Df Deviance    AIC
- TSkinThick    1   515.09 529.09
- Age           1   515.13 529.13
<none>              514.96 530.96
- DiastolicBP   1   517.70 531.70
- DiabPediFunc  1   519.38 533.38
- NoPreg        1   529.73 543.73
- BMI           1   537.43 551.43
- PlaGluConc    1   593.54 607.54

Step:  AIC=529.09
Class ~ NoPreg + PlaGluConc + DiastolicBP + BMI + DiabPediFunc +
    Age

               Df Deviance    AIC
- Age           1   515.31 527.31
<none>              515.09 529.09
- DiastolicBP   1   518.08 530.08
- DiabPediFunc  1   519.40 531.40
- NoPreg        1   529.86 541.86
- BMI           1   538.70 550.70
- PlaGluConc    1   594.14 606.14
```

```
Step:  AIC=527.31
Class ~ NoPreg + PlaGluConc + DiastolicBP + BMI + DiabPediFunc

               Df Deviance    AIC
<none>              515.31 527.31
- DiastolicBP   1   518.13 528.13
- DiabPediFunc  1   519.64 529.64
- BMI           1   538.71 548.71
- NoPreg        1   539.36 549.36
- PlaGluConc    1   598.86 608.86
```

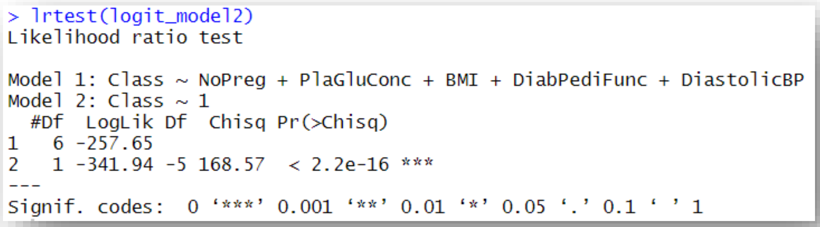
We can use the step function in order to verify the selection of the best model with the lowest AIC score.



```
> anova(logit_model2, test = "Chisq")
Analysis of Deviance Table

Model: binomial, link: logit

Response: Class

Terms added sequentially (first to last)


             Df Deviance Resid. Df Resid. Dev  Pr(>Chi)    
NULL                           533     683.88              
NoPreg        1   36.141       532     647.74 1.836e-09 ***
PlaGluConc    1  103.827       531     543.91 < 2.2e-16 ***
BMI           1   21.814       530     522.10 3.004e-06 ***
DiabPediFunc  1    3.973       529     518.13   0.04623 *  
DiastolicBP   1    2.816       528     515.31   0.09330 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

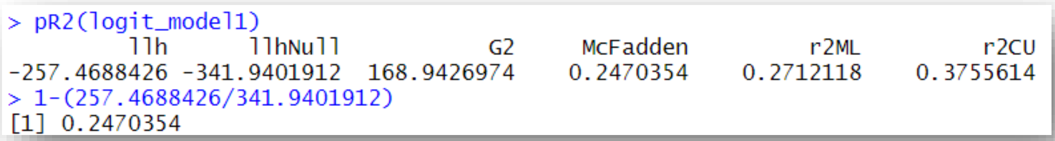
Likelihood Ratio – assesses the goodness of fit of two competing statistical models based on the ratio of their likelihoods. It compares the intercept-only model with the model with the predictor variables.

The p-value and Chisq value is significant, stating that the model built is significant enough for this model.



```
> lrtest(logit_model2)
Likelihood ratio test

Model 1: Class ~ NoPreg + PlaGluConc + BMI + DiabPediFunc + DiastolicBP
Model 2: Class ~ 1
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   6 -257.65
2   1 -341.94 -5 168.57  < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

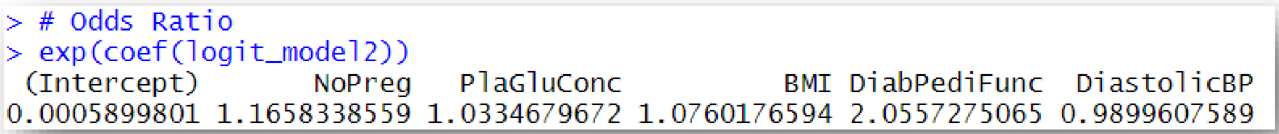
This will show the robustness of the model. McFadden score, tells – how much of the variance is actually explained by this particular model over the intercept-only model.



```
> pR2(logit_model1)
          llh       llhNull            G2      McFadden          r2ML          r2CU
-257.4688426 -341.9401912  168.9426974     0.2470354     0.2712118     0.3755614
> 1-(257.4688426/341.9401912)
[1] 0.2470354
```

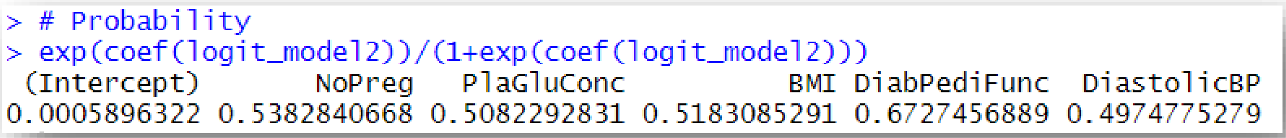
Coefficients of the variables and how much they are contributing to the overall model.

DiabPediFunc, NoPreg, BMI and PlaGluConc are contributing the most to the model.



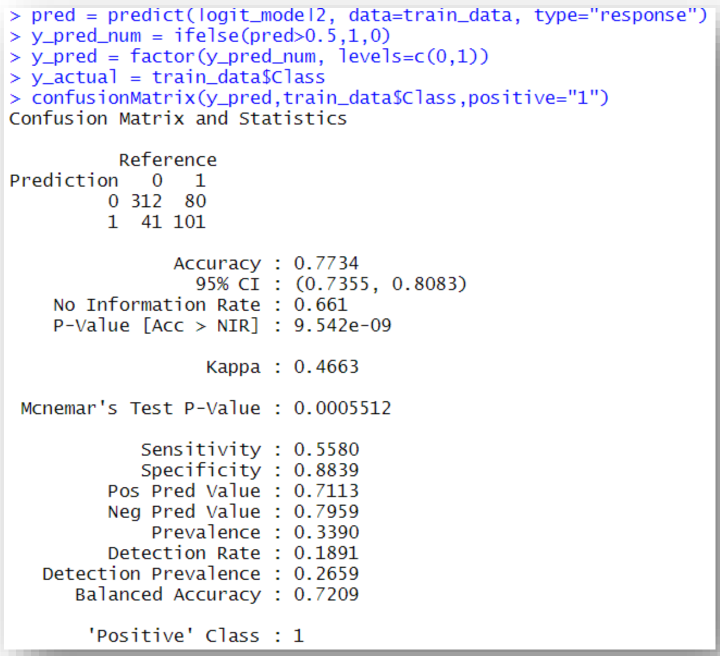
```
> # Odds Ratio
> exp(coef(logit_model2))
 (Intercept)       NoPreg    PlaGluConc          BMI DiabPediFunc  DiastolicBP
0.0005899801 1.1658338559 1.0334679672 1.0760176594 2.0557275065 0.9899607589
```

Probabilities for all the variables –



```
> # Probability
> exp(coef(logit_model2))/(1+exp(coef(logit_model2)))
 (Intercept)       NoPreg    PlaGluConc          BMI DiabPediFunc  DiastolicBP
0.0005896322 0.5382840668 0.5082292831 0.5183085291 0.6727456889 0.4974775279
```

5.2.5 Tuning the Model(to improve Sensitivity)

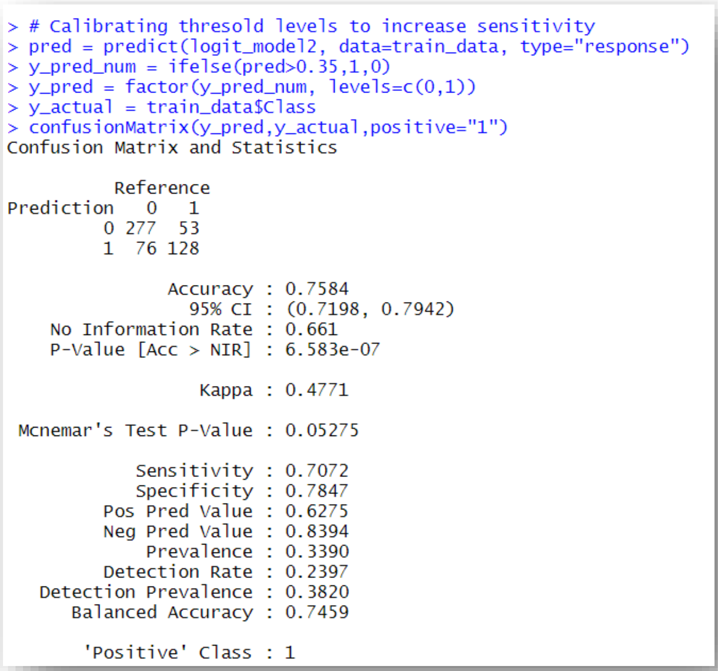


```
> pred = predict(logit_model2, data=train_data, type="response")
> y_pred_num = ifelse(pred>0.5,1,0)
> y_pred = factor(y_pred_num, levels=c(0,1))
> y_actual = train_data$Class
> confusionMatrix(y_pred,train_data$Class,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 312  80
         1  41 101

               Accuracy : 0.7734
                 95% CI : (0.7355, 0.8083)
    No Information Rate : 0.661
    P-Value [Acc > NIR] : 9.542e-09

                  Kappa : 0.4663

 Mcnemar's Test P-Value : 0.0005512

            Sensitivity : 0.5580
            Specificity : 0.8839
         Pos Pred Value : 0.7113
         Neg Pred Value : 0.7959
             Prevalence : 0.3390
         Detection Rate : 0.1891
   Detection Prevalence : 0.2659
      Balanced Accuracy : 0.7209

       'Positive' Class : 1
```

Sensitivity/True Positive Rate – These are cases in which we predicted yes (they have the disease), and they do have the disease.

In the above model the Accuracy is 77.34% but the Sensitivity is just 55.8%. In order to increase the value for sensitivity, a threshold of 0.35 was taken instead of the previous 0.50.

In the new model though the accuracy has decreased by a small amount, the Sensitivity has drastically improved by 14.9%. We will therefore consider the revised threshold to determine the model True Positive Rate.



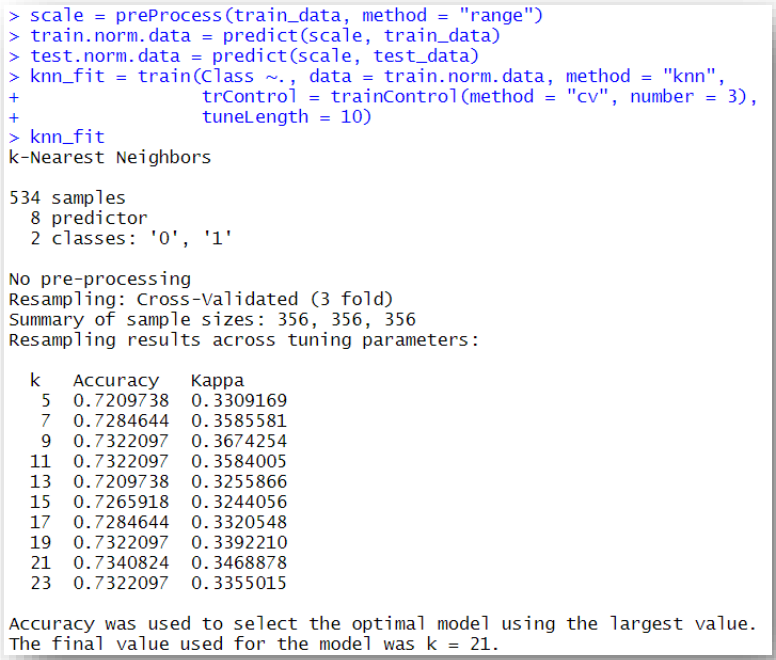
```
> # Calibrating thresold levels to increase sensitivity
> pred = predict(logit_model2, data=train_data, type="response")
> y_pred_num = ifelse(pred>0.35,1,0)
> y_pred = factor(y_pred_num, levels=c(0,1))
> y_actual = train_data$Class
> confusionMatrix(y_pred,y_actual,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 277  53
         1  76 128

               Accuracy : 0.7584
                 95% CI : (0.7198, 0.7942)
    No Information Rate : 0.661
    P-Value [Acc > NIR] : 6.583e-07

                  Kappa : 0.4771

 Mcnemar's Test P-Value : 0.05275

            Sensitivity : 0.7072
            Specificity : 0.7847
         Pos Pred Value : 0.6275
         Neg Pred Value : 0.8394
             Prevalence : 0.3390
         Detection Rate : 0.2397
   Detection Prevalence : 0.3820
      Balanced Accuracy : 0.7459

       'Positive' Class : 1
```

5.3 K-NN(K-Nearest Neighbour)

5.3.1 Normalizing continuous variables

Distance metric is highly influenced by the scale of the variable. Hence, it is important to standardize variables before utilizing them in model building. We will use min-max standardization method to bring all variables in same scale.

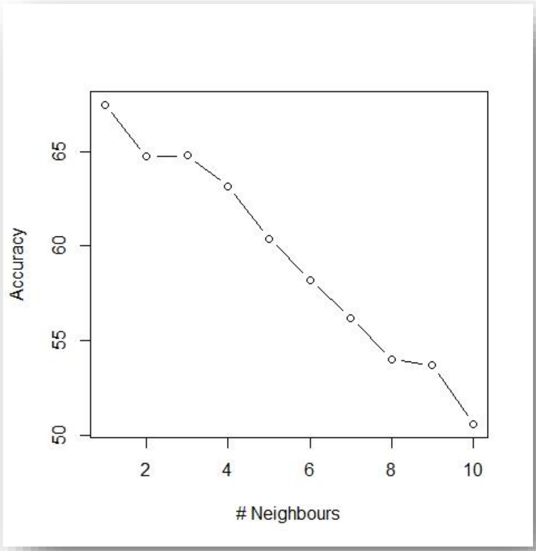
5.3.2 Model Building



```
> scale = preProcess(train_data, method = "range")
> train.norm.data = predict(scale, train_data)
> test.norm.data = predict(scale, test_data)
> knn_fit = train(Class ~., data = train.norm.data, method = "knn",
+                 trControl = trainControl(method = "cv", number = 3),
+                 tuneLength = 10)
> knn_fit
k-Nearest Neighbors

534 samples
  8 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 356, 356, 356
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   5  0.7209738  0.3309169
   7  0.7284644  0.3585581
   9  0.7322097  0.3674254
  11  0.7322097  0.3584005
  13  0.7209738  0.3255866
  15  0.7265918  0.3244056
  17  0.7284644  0.3320548
  19  0.7322097  0.3392210
  21  0.7340824  0.3468878
  23  0.7322097  0.3355015

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 21.
```

We use the "train" function to get the best k value from the model. It gives the best k value that was selected for this particular model.

This indicates that if we select 21 neighbours to make a classification decision, then this model will fit.



5.4 Naïve Bayes

5.4.1 Model Building

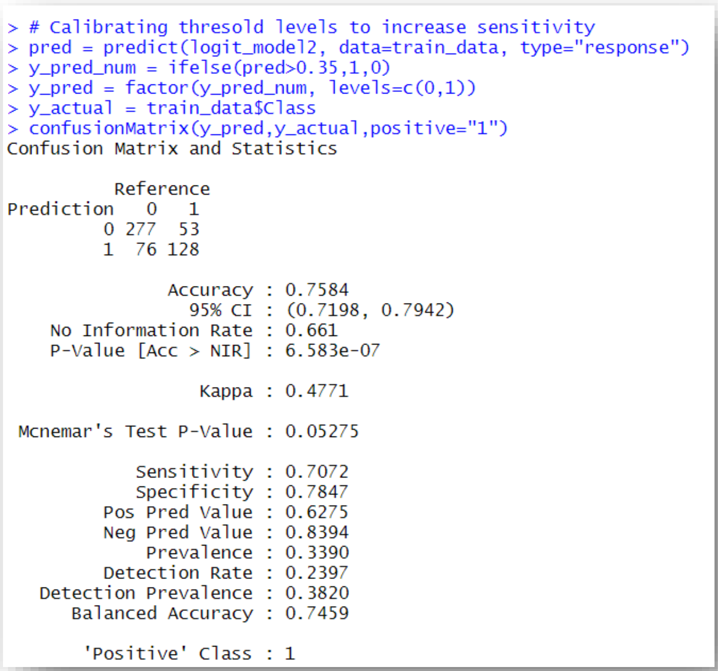
```
> NB

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = train.norm.data[-c(9)], y = train.norm.data$Class)

A-priori probabilities:
train.norm.data$Class
        0         1 
0.6610487 0.3389513 


Conditional probabilities:
                     NoPreg
train.norm.data$Class      [,1]      [,2]
                    0 0.1881353 0.1816886
                    1 0.3012675 0.2251628

                     PlaGluConc
train.norm.data$Class      [,1]      [,2]
                    0 0.5567363 0.1371966
                    1 0.7115854 0.1578604

                     DiastolicBP
train.norm.data$Class      [,1]      [,2]
                    0 0.5567965 0.1505474
                    1 0.5858618 0.1682860

                     TSkinThick
train.norm.data$Class      [,1]      [,2]
                    0 0.3104006 0.2376306
                    1 0.3359642 0.2711273

                     Test
train.norm.data$Class       [,1]      [,2]
                    0 0.08789104 0.1299195
                    1 0.13477990 0.1839854

                     BMI
train.norm.data$Class      [,1]      [,2]
                    0 0.4540388 0.1161715
                    1 0.5200863 0.1149361

                     DiabPediFunc
train.norm.data$Class      [,1]      [,2]
                    0 0.1541309 0.1343762
                    1 0.1955829 0.1527359

                     Age
train.norm.data$Class      [,1]      [,2]
                    0 0.1709160 0.1948376
                    1 0.2642726 0.1762445
```

Probabilities identified across all the variables

6. Model Comparison

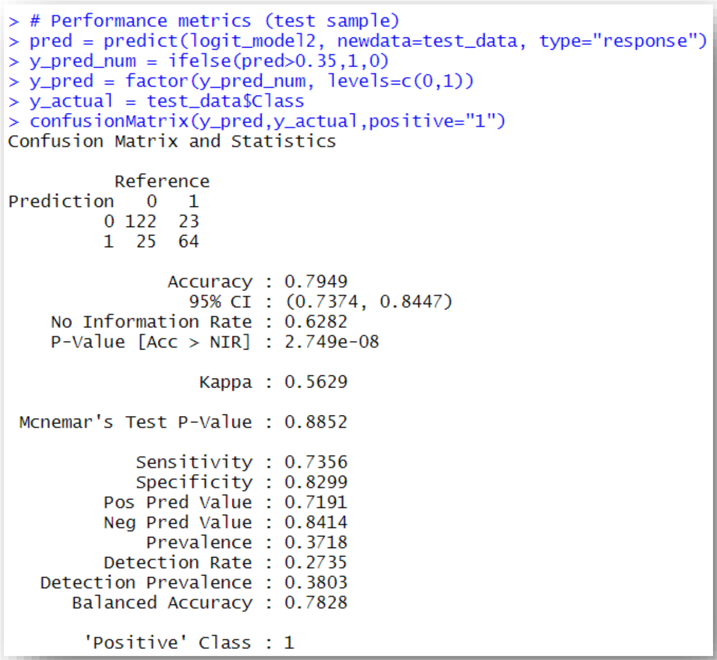
6.1 Logistic Regression

6.1.1 Train Dataset



```
> # Calibrating thresold levels to increase sensitivity
> pred = predict(logit_model2, data=train_data, type="response")
> y_pred_num = ifelse(pred>0.35,1,0)
> y_pred = factor(y_pred_num, levels=c(0,1))
> y_actual = train_data$Class
> confusionMatrix(y_pred,y_actual,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 277  53
         1  76 128

               Accuracy : 0.7584
                 95% CI : (0.7198, 0.7942)
    No Information Rate : 0.661
    P-Value [Acc > NIR] : 6.583e-07

                  Kappa : 0.4771

 Mcnemar's Test P-Value : 0.05275

            Sensitivity : 0.7072
            Specificity : 0.7847
         Pos Pred Value : 0.6275
         Neg Pred Value : 0.8394
             Prevalence : 0.3390
         Detection Rate : 0.2397
   Detection Prevalence : 0.3820
      Balanced Accuracy : 0.7459

       'Positive' Class : 1
```

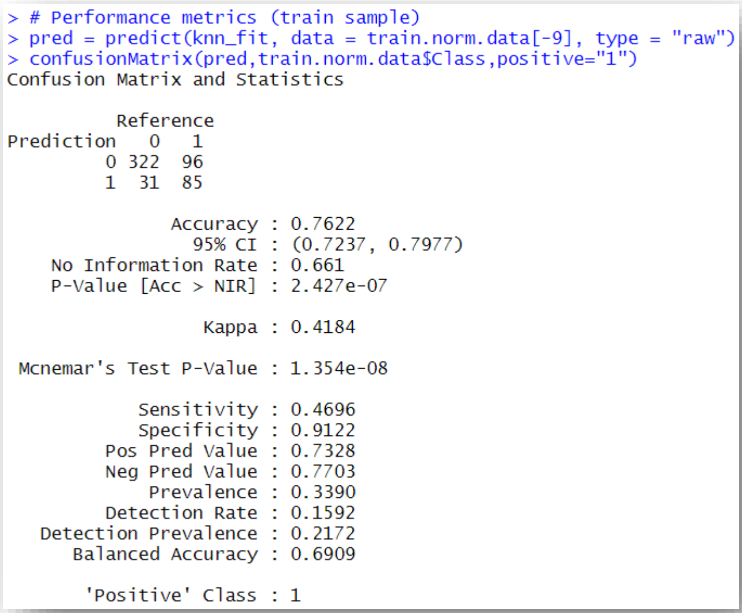
6.1.2 Test Data



```
> # Performance metrics (test sample)
> pred = predict(logit_model2, newdata=test_data, type="response")
> y_pred_num = ifelse(pred>0.35,1,0)
> y_pred = factor(y_pred_num, levels=c(0,1))
> y_actual = test_data$Class
> confusionMatrix(y_pred,y_actual,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 122  23
         1  25  64

               Accuracy : 0.7949
                 95% CI : (0.7374, 0.8447)
    No Information Rate : 0.6282
    P-Value [Acc > NIR] : 2.749e-08

                  Kappa : 0.5629

 Mcnemar's Test P-Value : 0.8852

            Sensitivity : 0.7356
            Specificity : 0.8299
         Pos Pred Value : 0.7191
         Neg Pred Value : 0.8414
             Prevalence : 0.3718
         Detection Rate : 0.2735
   Detection Prevalence : 0.3803
      Balanced Accuracy : 0.7828

       'Positive' Class : 1
```

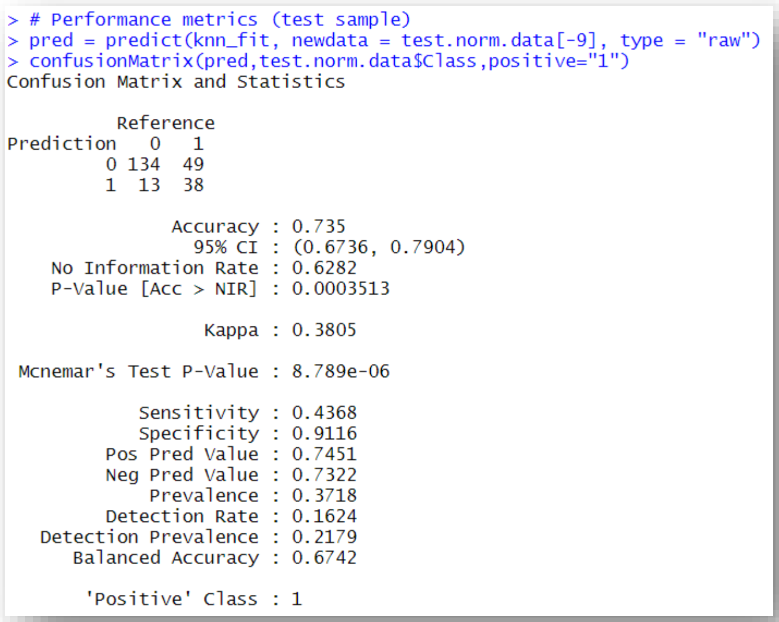
6.2 K Nearest Neighbour

6.2.1 Train Dataset



```
> # Performance metrics (train sample)
> pred = predict(knn_fit, data = train.norm.data[-9], type = "raw")
> confusionMatrix(pred,train.norm.data$Class,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 322  96
         1  31  85

               Accuracy : 0.7622
                 95% CI : (0.7237, 0.7977)
    No Information Rate : 0.661
    P-Value [Acc > NIR] : 2.427e-07

                  Kappa : 0.4184

 Mcnemar's Test P-Value : 1.354e-08

            Sensitivity : 0.4696
            Specificity : 0.9122
         Pos Pred Value : 0.7328
         Neg Pred Value : 0.7703
             Prevalence : 0.3390
         Detection Rate : 0.1592
   Detection Prevalence : 0.2172
      Balanced Accuracy : 0.6909

       'Positive' Class : 1
```

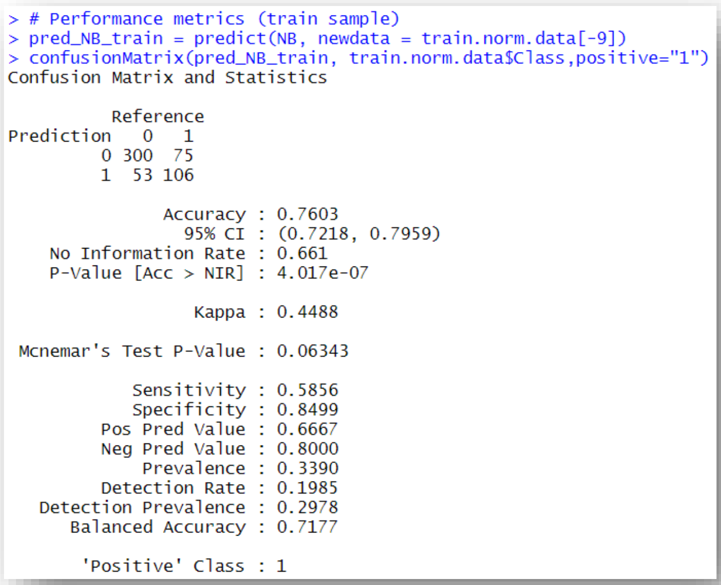
6.2.2 Test Data



```
> # Performance metrics (test sample)
> pred = predict(knn_fit, newdata = test.norm.data[-9], type = "raw")
> confusionMatrix(pred,test.norm.data$Class,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 134  49
         1  13  38

               Accuracy : 0.735
                 95% CI : (0.6736, 0.7904)
    No Information Rate : 0.6282
    P-Value [Acc > NIR] : 0.0003513

                  Kappa : 0.3805

 Mcnemar's Test P-Value : 8.789e-06

            Sensitivity : 0.4368
            Specificity : 0.9116
         Pos Pred Value : 0.7451
         Neg Pred Value : 0.7322
             Prevalence : 0.3718
         Detection Rate : 0.1624
   Detection Prevalence : 0.2179
      Balanced Accuracy : 0.6742

       'Positive' Class : 1
```

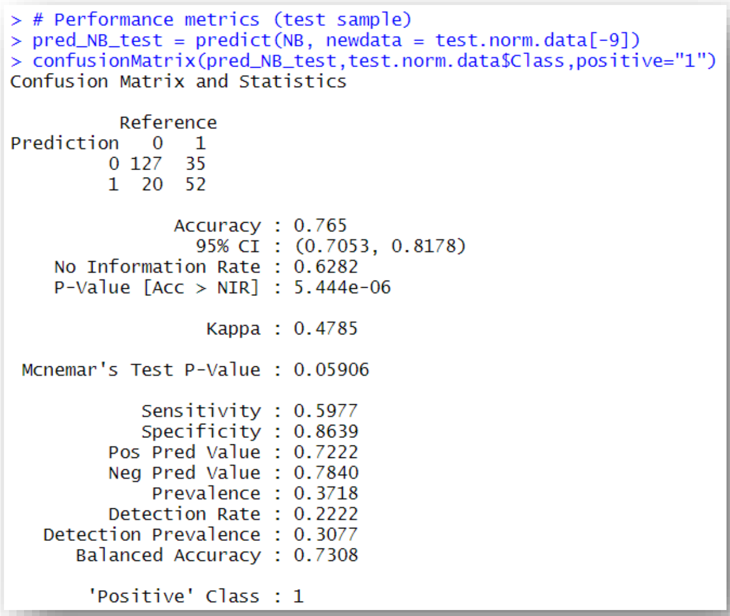
6.3 Naïve Bayes

6.3.1 Train Dataset



```
> # Performance metrics (train sample)
> pred_NB_train = predict(NB, newdata = train.norm.data[-9])
> confusionMatrix(pred_NB_train, train.norm.data$Class,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 300  75
         1  53 106

               Accuracy : 0.7603
                 95% CI : (0.7218, 0.7959)
    No Information Rate : 0.661
    P-Value [Acc > NIR] : 4.017e-07

                  Kappa : 0.4488

 Mcnemar's Test P-Value : 0.06343

            Sensitivity : 0.5856
            Specificity : 0.8499
         Pos Pred Value : 0.6667
         Neg Pred Value : 0.8000
             Prevalence : 0.3390
         Detection Rate : 0.1985
   Detection Prevalence : 0.2978
      Balanced Accuracy : 0.7177

       'Positive' Class : 1
```

6.3.2 Test Data



```
> # Performance metrics (test sample)
> pred_NB_test = predict(NB, newdata = test.norm.data[-9])
> confusionMatrix(pred_NB_test,test.norm.data$Class,positive="1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 127  35
         1  20  52

               Accuracy : 0.765
                 95% CI : (0.7053, 0.8178)
    No Information Rate : 0.6282
    P-Value [Acc > NIR] : 5.444e-06

                  Kappa : 0.4785

 Mcnemar's Test P-Value : 0.05906

            Sensitivity : 0.5977
            Specificity : 0.8639
         Pos Pred Value : 0.7222
         Neg Pred Value : 0.7840
             Prevalence : 0.3718
         Detection Rate : 0.2222
   Detection Prevalence : 0.3077
      Balanced Accuracy : 0.7308

       'Positive' Class : 1
```

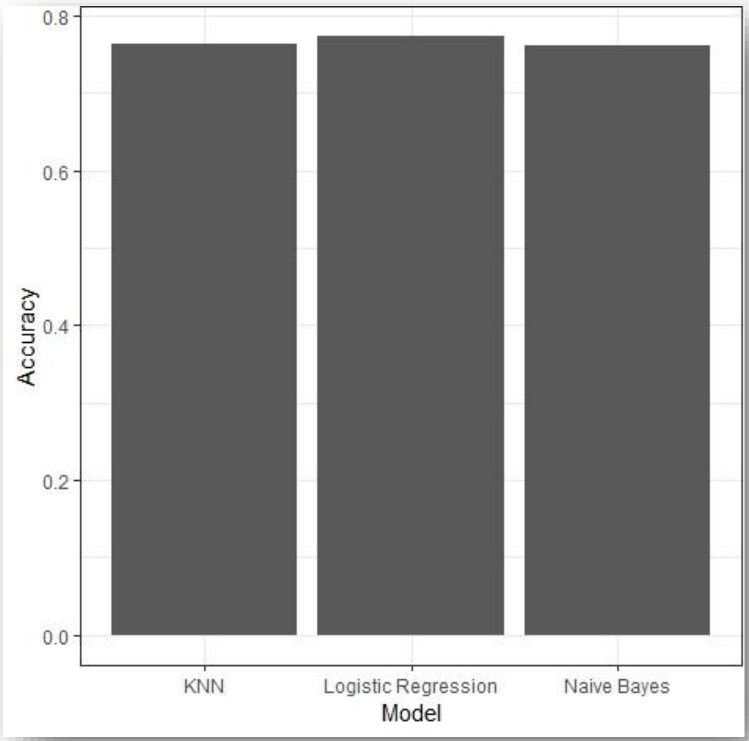
For Logistic Regression, KNN and Naïve Bayes, the model when validated on the test/validation dataset gives a result similar to that of the train dataset. We can therefore say that the model is not an over-fit model.

7. Cross Validation & Insights

7.1 Cross Validation

In this problem, decision makers will be keen to identify the positives very accurately. Therefore, we will not just evaluate the model based on the Accuracy but also use Sensitivity as a measuring factor to compare the models.

	Accuracy			**Sensitivity**			**Specificity**		
	Logistic Regression	**KNN**	**Naïve Bayes**	**Logistic Regression**	**KNN**	**Naïve Bayes**	**Logistic Regression**	**KNN**	**Naïve Bayes**
Train	**77%**	76%	76%	**71%**	47%	59%	79%	**91%**	85%
Test	**79%**	74%	77%	**75%**	44%	60%	82%	**91%**	86%



From the above results it is clear that although there is not much difference in all the three algorithms, **Logistic Regression** gives a slightly better results at predicting the true positives as compared to the others.

The Diabetes Database was analysed and explored in detail. The patterns identified using Data Exploration methods were validated using the modelling techniques employed. Classification models such as Logistic Regression, K-Nearest Neighbour and Naïve Bayes were built and evaluated to identify best model to predict the occurrence of Diabetes in women. From the cross-validated performance measure of sensitivity, the Logistic

Regression model was concluded as the best performing model while Naïve Bayes seems to have performed the worst.

7.2 Insights

It is clear from the analysis that a marginal increase in most of the factors can increase the probability of a woman being diagnosed with diabetes. It is therefore, important to focus on correctly classifying the results over the misclassifications.

Therefore, the occurrence of the disease is significantly affected by the following factors -

- **NoPreg –** nbm,
- **PlaGluConc –**
- **DiastolicBP –**
- **TSkinThick –**
- **Test –**
- **BMI –**
- **DiabPediFunc –**
- **Age –**

8. Source of Data

- Great Learning Mentored Learning Session and Recorded Sessions
- Google
- R Blogger
- stats.stackexchange.com
- uc-r.github.io
- www.kaggle.com
- www.rpubs.com
- www.datacamp.com