Diabetes.markdown

nirman

2023-11-29

Importing the packages and calling it by library function

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.3.2
```

```
library(imputeTS)  ##for mean imputation
```

```
## Warning: package 'imputeTS' was built under R version 4.3.2
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(caTools)    ##package for train test split
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

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

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

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

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.3.2
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: lattice
```

```
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.3.2
```

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

```
##
## Attaching package: 'pROC'
```

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

```
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.3.2
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

Loaded the dataset from local system

```
df=read.csv("F:/Desktop items/datasets/diabetes.csv")
head(df)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35       0 33.6
## 2           1      85            66            29       0 26.6
## 3           8     183            64             0       0 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74             0       0 25.6
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

```
tail(df)
```

```
##     Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 763           9      89            62             0       0 22.5
## 764          10     101            76            48     180 32.9
## 765           2     122            70            27       0 36.8
## 766           5     121            72            23     112 26.2
## 767           1     126            60             0       0 30.1
## 768           1      93            70            31       0 30.4
##     DiabetesPedigreeFunction Age Outcome
## 763                    0.142  33       0
## 764                    0.171  63       0
## 765                    0.340  27       0
## 766                    0.245  30       0
## 767                    0.349  47       1
## 768                    0.315  23       0
```

Checking data types for all variables

```
str(df)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ Pregnancies             : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : int  72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : int  35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : 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 ...
##  $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : int  1 0 1 0 1 0 1 0 1 1 ...
```

```
sum(is.na(df))
```

```
## [1] 0
```

No missing values

Checking for Duplicate Values

```
sum(duplicated(df))
```

```
## [1] 0
```

No duplicate values

Checking for inappropriate or zero values in variables

```
for (col in names(df)) {
  zero_values <- sum(df[[col]] <= 0)
  cat(paste('Zero values in column', col, '=', zero_values, '\n'))
}
```

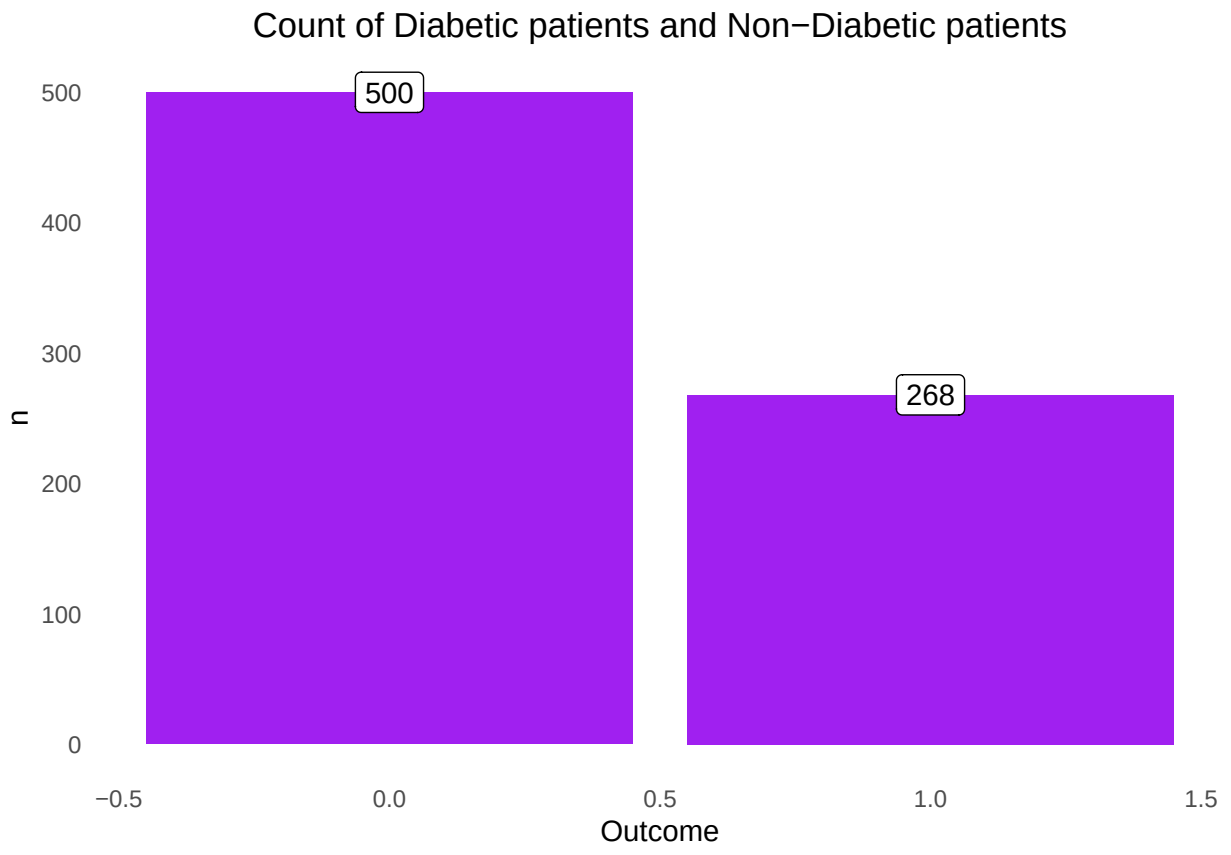
```
## Zero values in column Pregnancies = 111
## Zero values in column Glucose = 5
## Zero values in column BloodPressure = 35
## Zero values in column SkinThickness = 227
## Zero values in column Insulin = 374
## Zero values in column BMI = 11
## Zero values in column DiabetesPedigreeFunction = 0
## Zero values in column Age = 0
## Zero values in column Outcome = 500
```

Here Outcome has 500 zero values which is obvious as outcome is in 0's and 1's and Pregnancies has 111 zeroes because there might be women without pregnant and has diabetes.

EXPLORATORY DATA ANALYSIS \

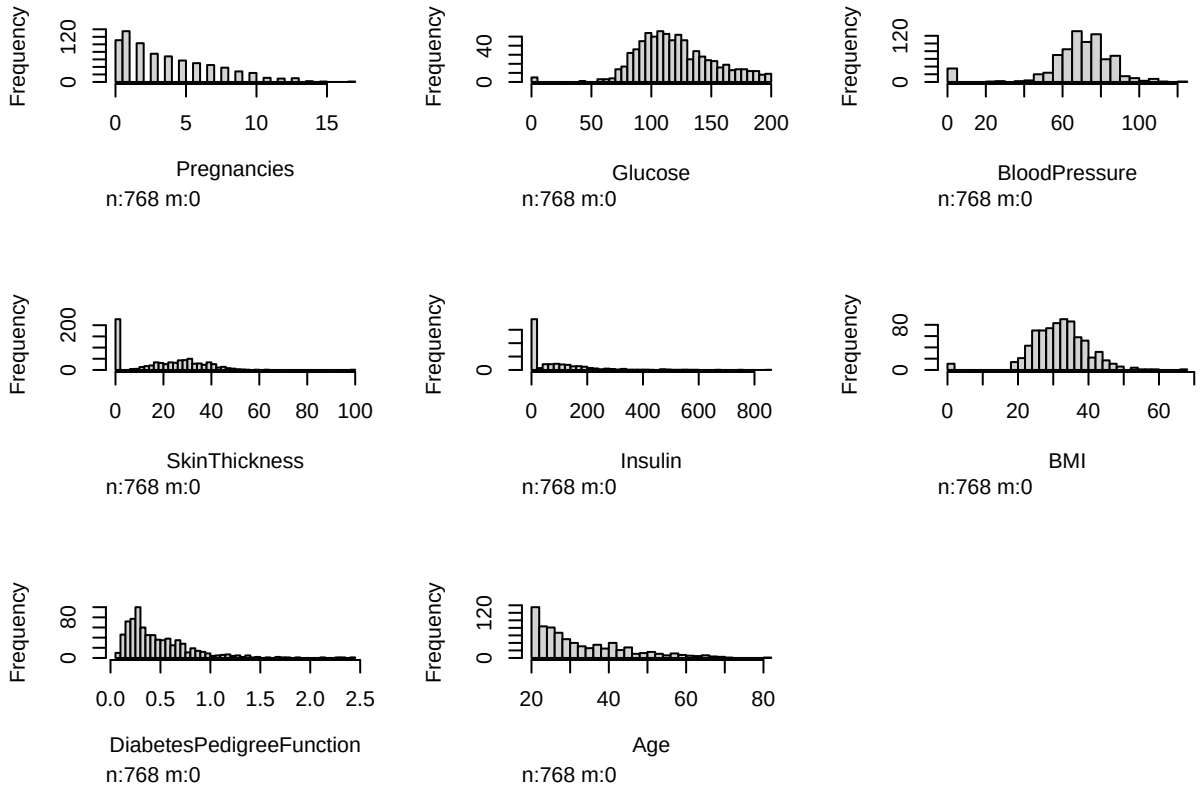
Comparison of patients with and without diabetes through Bar charts

```
df %>%
  group_by(Outcome)%>%
  summarise(n=n())%>%
  ggplot(aes(x=Outcome,y=n))+
    geom_bar(stat = 'identity',fill='purple')+
    geom_label(aes(label=n))+
  theme_minimal()+
  labs(title = "Count of Diabetic patients and Non-Diabetic patients")+
  theme(panel.grid = element_blank())+
  theme(plot.title = element_text(hjust = 0.5))
```



Plotting histogram to observe the distribution of data

```
hist.data.frame(df)
```



Obtaining correlations matrix

```
cor(df)   ##computed correlation matrix
```

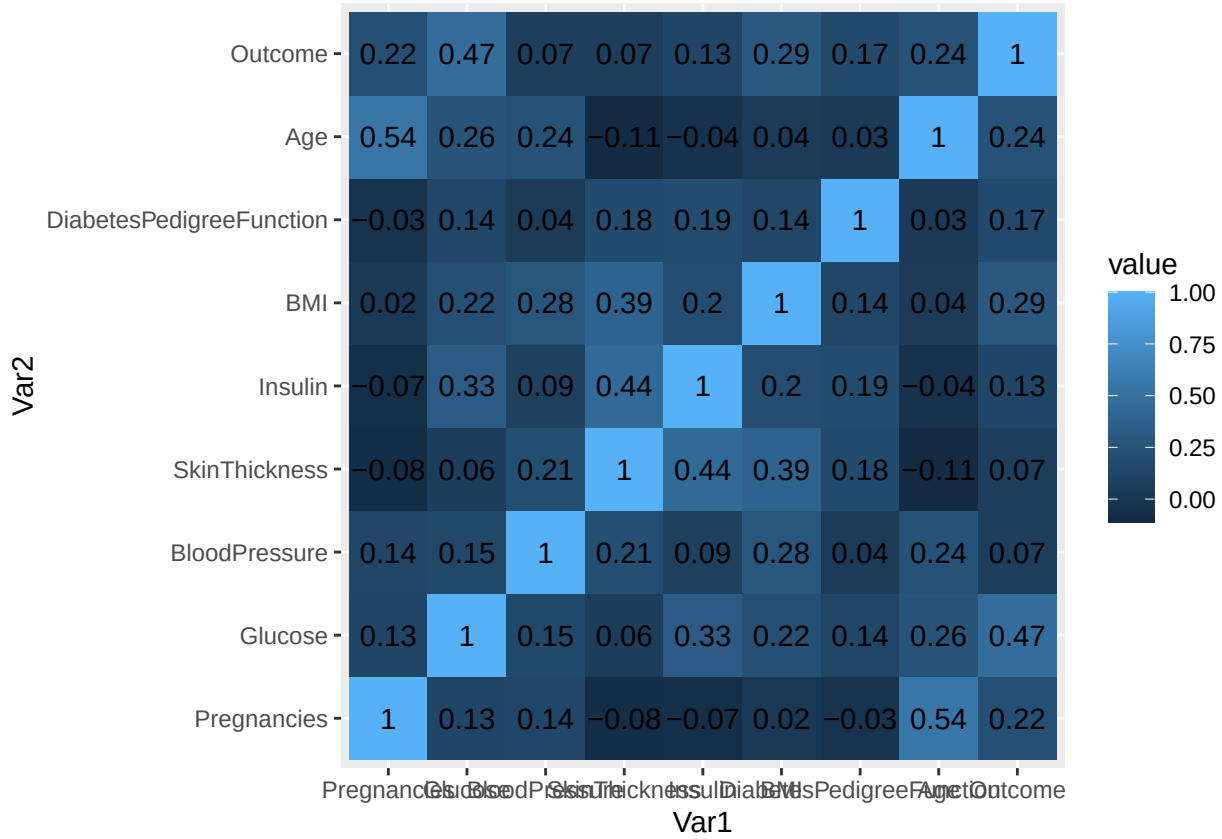
```
##                           Pregnancies    Glucose BloodPressure SkinThickness
## Pregnancies                1.00000000 0.12945867    0.14128198   -0.08167177
## Glucose                    0.12945867 1.00000000    0.15258959    0.05732789
## BloodPressure              0.14128198 0.15258959    1.00000000    0.20737054
## SkinThickness             -0.08167177 0.05732789    0.20737054    1.00000000
## Insulin                   -0.07353461 0.33135711    0.08893338    0.43678257
## BMI                        0.01768309 0.22107107    0.28180529    0.39257320
## DiabetesPedigreeFunction -0.03352267 0.13733730    0.04126495    0.18392757
## Age                        0.54434123 0.26351432    0.23952795   -0.11397026
## Outcome                    0.22189815 0.46658140    0.06506836    0.07475223
##                              Insulin        BMI DiabetesPedigreeFunction
## Pregnancies               -0.07353461 0.01768309              -0.03352267
## Glucose                    0.33135711 0.22107107               0.13733730
## BloodPressure              0.08893338 0.28180529               0.04126495
## SkinThickness              0.43678257 0.39257320               0.18392757
## Insulin                    1.00000000 0.19785906               0.18507093
## BMI                        0.19785906 1.00000000               0.14064695
## DiabetesPedigreeFunction   0.18507093 0.14064695               1.00000000
```

```
## Age                         -0.04216295 0.03624187                0.03356131
## Outcome                      0.13054795 0.29269466                0.17384407
##                                     Age    Outcome
## Pregnancies                  0.54434123 0.22189815
## Glucose                      0.26351432 0.46658140
## BloodPressure                0.23952795 0.06506836
## SkinThickness               -0.11397026 0.07475223
## Insulin                     -0.04216295 0.13054795
## BMI                          0.03624187 0.29269466
## DiabetesPedigreeFunction     0.03356131 0.17384407
## Age                          1.00000000 0.23835598
## Outcome                      0.23835598 1.00000000
```

```
corr_mat=round(cor(df),2)  ##rounded to 2 decimel
melted_corr_mat <- melt(corr_mat)
```

Plotting the correlation heatmap with annotations

```
ggplot(data = melted_corr_mat, aes(x=Var1, y=Var2,
                                   fill=value)) +
geom_tile()+
geom_text(aes(Var2, Var1, label = value),
          color = "black", size = 4)
```



looking for missing values

```
sum(is.na(df$Pregnancies))
```

```
## [1] 0
```

```
sum(is.na(df$Glucose))
```

```
## [1] 0
```

```
sum(is.na(df$BloodPressure))
```

```
## [1] 0
```

```
sum(is.na(df$SkinThickness))
```

```
## [1] 0
```

```
sum(is.na(df$Insulin))
```

```
## [1] 0
```

```
sum(is.na(df$BMI))
```

```
## [1] 0
```

```
sum(is.na(df$Age))
```

```
## [1] 0
```

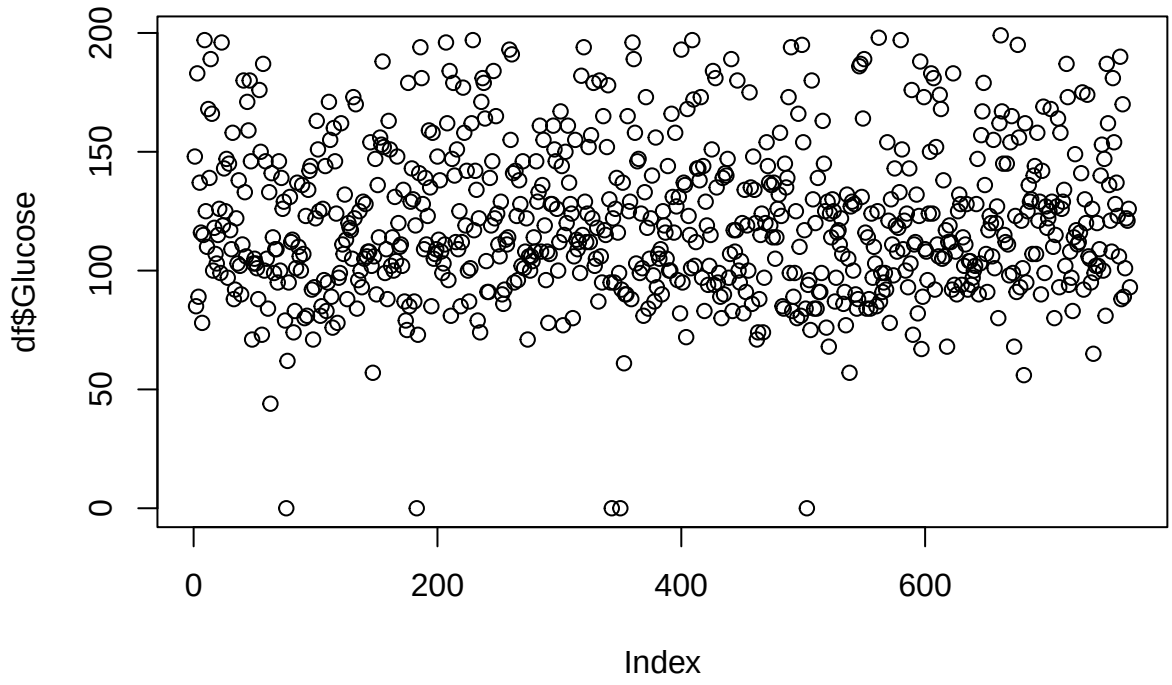
```
sum(is.na(df$Outcome))
```

```
## [1] 0
```

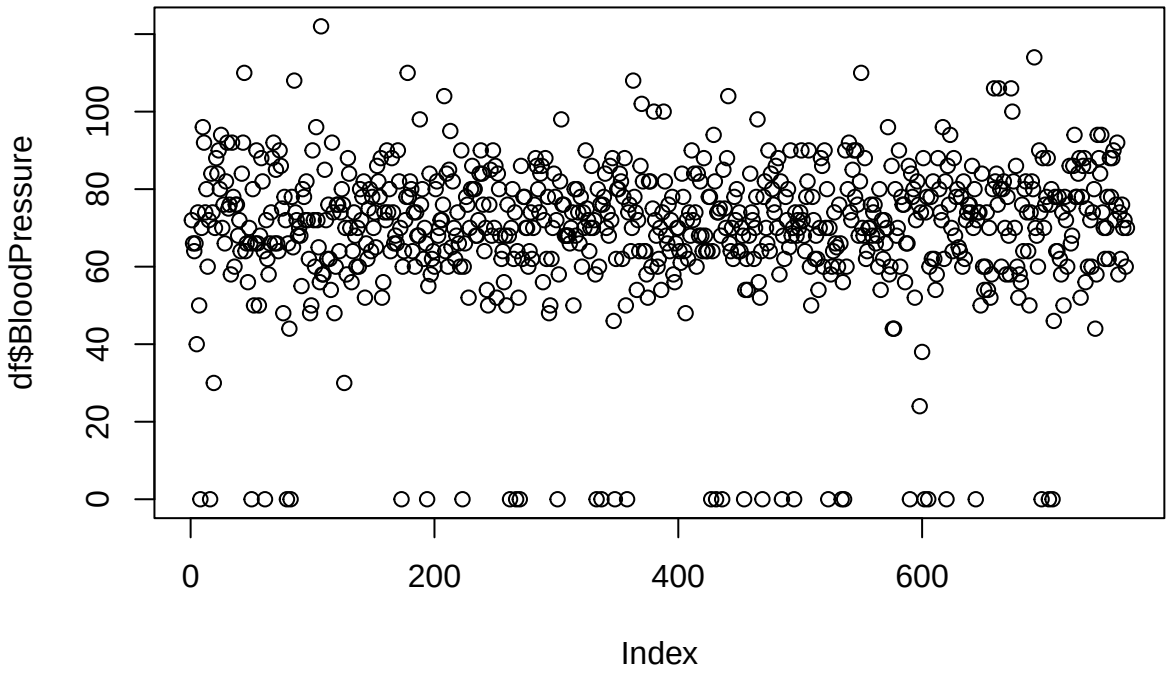
```
#so the data has no missing values
```

Checking Outliers using Scatterlot

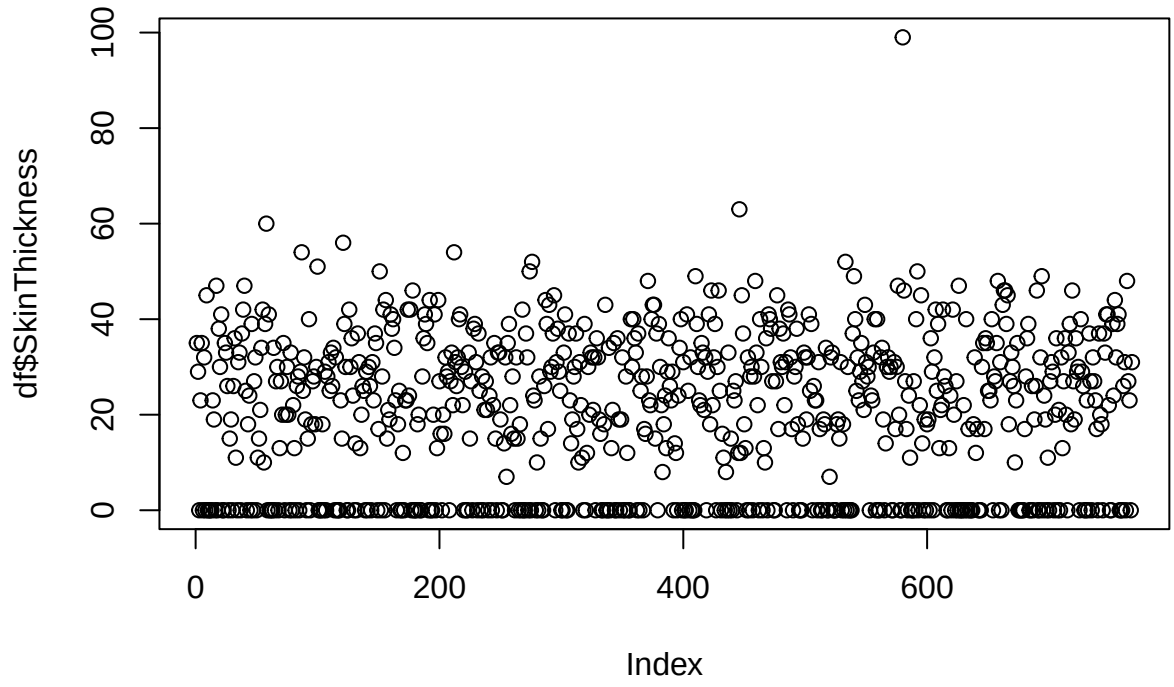
```
plot(df$Glucose)
```



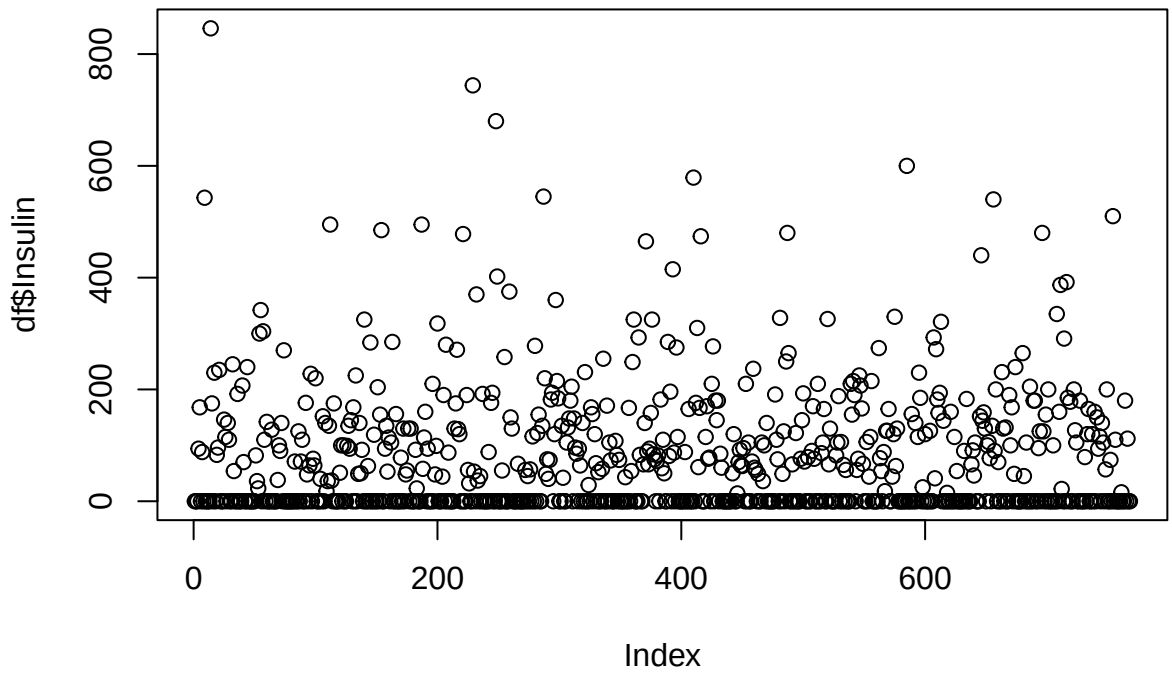

```
plot(df$BloodPressure)
```



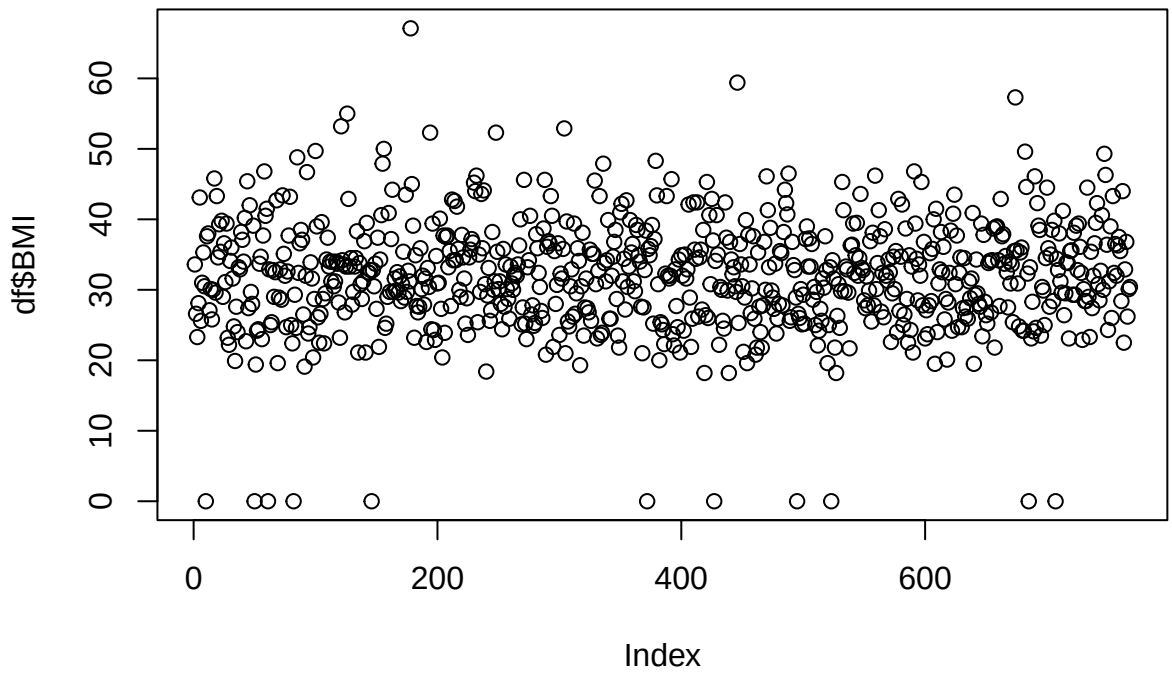

```
plot(df$SkinThickness)
```



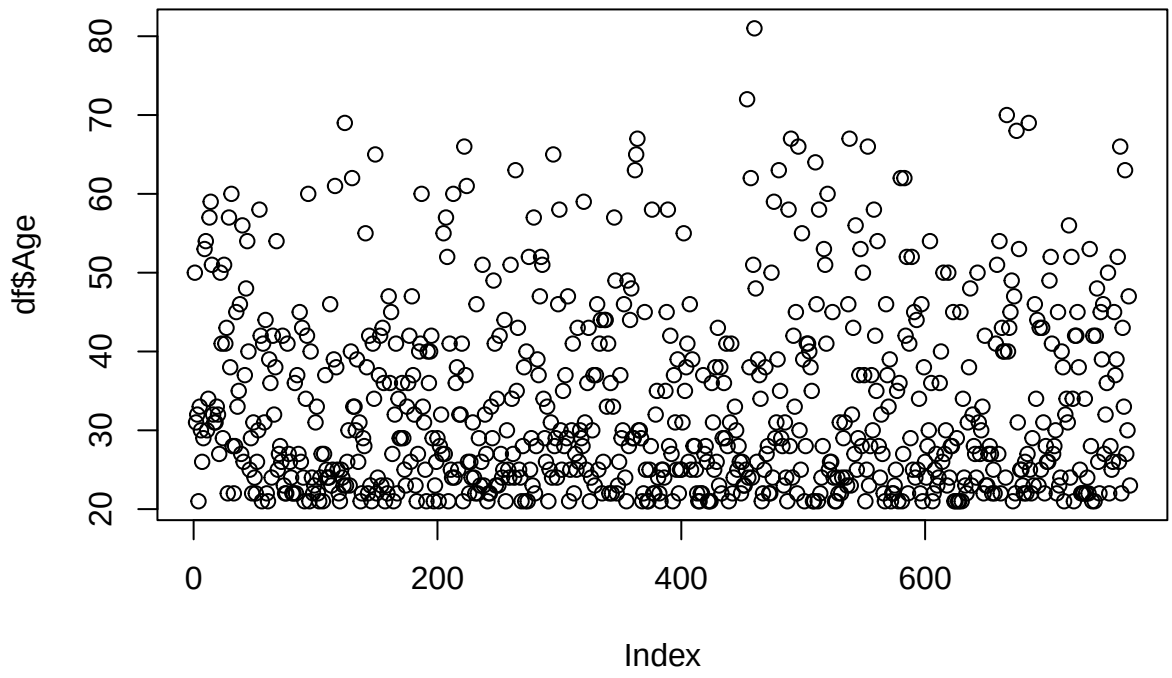

```
plot(df$Insulin)
```




```
plot(df$BMI)
```




```
plot(df$Age)
```



It is observed that the above features have many outliers and the outliers are zero values

```
###check for summary of data frame
summary(df)
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##     Outcome
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
```

```
##  Max.   :1.000
```

```
colSums(df==0)  ##checking the number of zeros in each features
```

```
##              Pregnancies                  Glucose            BloodPressure
##                      111                        5                       35
##            SkinThickness                  Insulin                      BMI
##                      227                      374                       11
## DiabetesPedigreeFunction                      Age                  Outcome
##                        0                        0                      500
```

```
#replacing the zeros by imputing mean in each of the important features
##converting the zero values to NA
df$Glucose[df$Glucose==0]<-NA
df$BloodPressure[df$BloodPressure==0]<-NA
df$SkinThickness[df$SkinThickness==0]<-NA
df$Insulin[df$Insulin==0]<-NA
df$BMI[df$BMI==0]<-NA

df$Glucose<-round(na_mean(df$Glucose, option = "mean", maxgap = Inf),0)##this is only for mean imputati
df$BloodPressure<-round(na_mean(df$BloodPressure, option = "mean", maxgap = Inf),0)##this is only for m
df$SkinThickness<-round(na_mean(df$SkinThickness, option = "mean", maxgap = Inf),0)##this is only for m
df$Insulin<-round(na_mean(df$Insulin, option = "mean", maxgap = Inf),0)##this is only for mean imputati
df$BMI<-round(na_mean(df$BMI, option = "mean", maxgap = Inf),1)##this is only for mean imputation
```

Logistic Regression Model

```
##split features and target
X=df[-8]          ##features
y=df[8]             ##target
```

Splitting Train Test Data

```
##splitting the data into training and test data

set.seed(1)
sample <- sample.split(df, SplitRatio = 0.8)
train_df  <- subset(df, sample == TRUE)
test_df   <- subset(df, sample == FALSE)
```

```
##spliting the data set into outcome and features
train_X=train_df[-8]
train_y=train_df[8]
test_X=test_df[-8]
test_y=test_df[8]
```

```
#checking the dimensions
dim(train_y)
```

```
## [1] 598   1
```

```
dim(test_y)
```

```
## [1] 170   1
```

```
dim(train_X)
```

```
## [1] 598   8
```

```
dim(test_X)
```

```
## [1] 170   8
```

```
head(train_X)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35     156 33.6
## 2           1      85            66            29     156 26.6
## 3           8     183            64            29     156 23.3
## 5           0     137            40            35     168 43.1
## 6           5     116            74            29     156 25.6
## 8          10     115            72            29     156 35.3
##   DiabetesPedigreeFunction Outcome
## 1                    0.627       1
## 2                    0.351       0
## 3                    0.672       1
## 5                    2.288       1
## 6                    0.201       0
## 8                    0.134       0
```

```
head(train_df)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35     156 33.6
## 2           1      85            66            29     156 26.6
## 3           8     183            64            29     156 23.3
## 5           0     137            40            35     168 43.1
## 6           5     116            74            29     156 25.6
## 8          10     115            72            29     156 35.3
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 5                    2.288  33       1
## 6                    0.201  30       0
## 8                    0.134  29       0
```

```
###Model training
logit_model<- glm( Outcome ~  Age+BMI+ Insulin+SkinThickness+BloodPressure+Glucose+Pregnancies,
                   data = train_df,
                   family = "binomial")
summary(logit_model)
```

```
##
## Call:
## glm(formula = Outcome ~ Age + BMI + Insulin + SkinThickness +
##     BloodPressure + Glucose + Pregnancies, family = "binomial",
##     data = train_df)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -8.9503738  0.9278439  -9.646  < 2e-16 ***
## Age            0.0135998  0.0111284   1.222 0.221674
## BMI            0.0840806  0.0202238   4.158 3.22e-05 ***
## Insulin       -0.0001525  0.0013175  -0.116 0.907868
## SkinThickness  0.0092059  0.0153923   0.598 0.549783
## BloodPressure -0.0081168  0.0095356  -0.851 0.394651
## Glucose        0.0387344  0.0044635   8.678  < 2e-16 ***
## Pregnancies    0.1327873  0.0363912   3.649 0.000263 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 766.25  on 597  degrees of freedom
## Residual deviance: 554.27  on 590  degrees of freedom
## AIC: 570.27
##
## Number of Fisher Scoring iterations: 5
```

```
###model testing
predict_model <- predict(logit_model,
                      test_df, type = "response")
```

```
###model performance
test_class<-ifelse(predict_model<=0.5, 0, 1)   ###assuming 0.5 as the optimal threshold probability
```

```
#it is required to  factor the outcomes in order to create a confusion matrix
test_class<-as.factor(test_class)
test_df$Outcome<-as.factor(test_df$Outcome)

confusionMatrix(test_df$Outcome,test_class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 95 10
##          1 24 41
##
##                Accuracy : 0.8
##                  95% CI : (0.7319, 0.8573)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.00216
##
##                   Kappa : 0.5584
```

```
##
##  Mcnemar's Test P-Value : 0.02578
##
##             Sensitivity : 0.7983
##             Specificity : 0.8039
##          Pos Pred Value : 0.9048
##          Neg Pred Value : 0.6308
##              Prevalence : 0.7000
##          Detection Rate : 0.5588
##    Detection Prevalence : 0.6176
##       Balanced Accuracy : 0.8011
##
##        'Positive' Class : 0
##
```

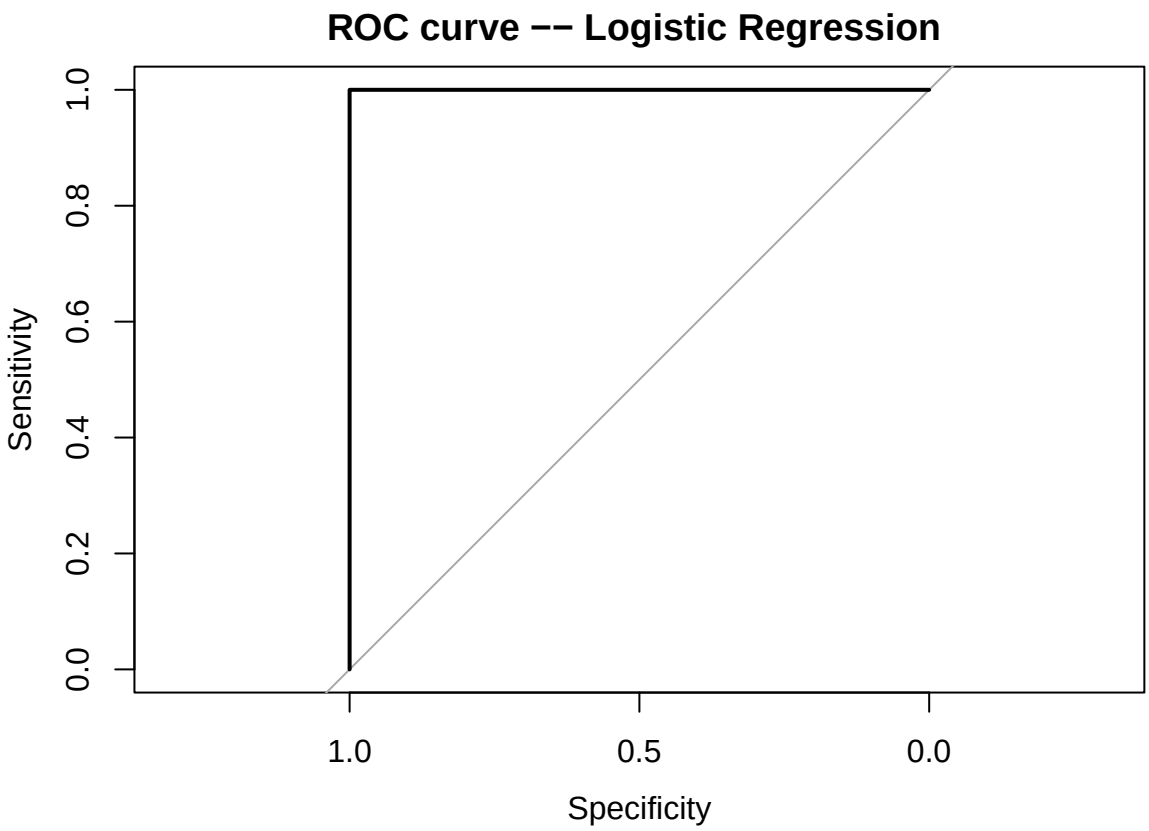
```
##ordering the predictor and test dataframe
test_df$Outcome<-order(test_df$Outcome)
test_class<-order(test_class)

##plot the ROC curve
#ROC-curve using pROC library

roc_score=roc(test_df$Outcome, test_class) #AUC score
```

```
## Warning in roc.default(test_df$Outcome, test_class): 'response' has more than
## two levels. Consider setting 'levels' explicitly or using 'multiclass.roc'
## instead
```

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

```
## Setting direction: controls < cases
```

```
plot(roc_score ,main ="ROC curve -- Logistic Regression ")
```



Thus from the ROC-AUC curve we can see that performance of logistic regression is better than some random classifier.