Prediction of Gestational Diabetes via Logistic Regression

Group Project

import usful libraries

```
library(dplyr)

##
## 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(magrittr)
library(knitr)
library(ggplot2)
```

A) Load the dataset in R Studio. Examine the first few rows of data using R. Explain your findings. Did you notice anything abnormal or interesting?

We first import the provided data set

```
patients <- read.csv("C:/Users/Z/Desktop/CIS 690/Group project/patients.csv")
```

To check and examine the data and the summary of the data

```
head(patients)

##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI Pedigree Ag
e
## 1           6     148            72            35       0 33.6    0.627  5
0
## 2           1      85            66            29       0 26.6    0.351  3
1
## 3           8     183            64             0       0 23.3    0.672  3
2
## 4           1      89            66            23      94 28.1    0.167  2
1
## 5           0     137            40            35     168 43.1    2.288  3
3
## 6           5     116            74             0       0 25.6    0.201  3
0
##   Diagnosis
## 1         1
## 2         0
## 3         1
## 4         0
## 5         1
## 6         0
```

```
summary(patients)
```

```
##   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           Pedigree           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
##    Diagnosis
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
##  Max.   :1.000
```

from the summary table we can say that there are outliers in the data, also age , pedigree, pregnancies, insulin and glucose are positively skewed(mean>median) and blood pressure, skin thickness and bmi are negatively skewed(median>mean).

for insulin inter-quartile range looks larger compared to other features

In pregnancies, 0,1,2,3,4 are more compared to remaining values

B) calculate the mean, median, sd and quartiles of all the features means of all the values

```
# For all the mean values
pregnancies_mean <- mean(patients$Pregnancies)
glucose_mean <- mean(patients$Glucose)
bloodpressure_mean <- mean(patients$BloodPressure)
skinThickness_mean <- mean(patients$SkinThickness)
insulin_mean <- mean(patients$Insulin)
bmi_mean <- mean(patients$BMI)
pedigree_mean <- mean(patients$Pedigree)
age_mean <- mean(patients$Age)

# For all of the sd values
pregnancies_sd <- sd(patients$Pregnancies)
```

```
glucose_sd <- sd(patients$Glucose)
blood_sd <- sd(patients$BloodPressure)
skinth_sd <- sd(patients$SkinThickness)
insulin_sd <- sd(patients$Insulin)
bmi_sd <- sd(patients$BMI)
pedigree_sd <- sd(patients$Pedigree)
age_sd <- sd(patients$age)
```

To find the quartiles (where Q3 is 75% percentile, Q1 is the 25% percentile and Q2 is the 50% percentile which is the median)

To find the values of Q3

```
pregnancies_q3 <- quantile(patients$Pregnancies, 0.75)
glucose_q3 <- quantile(patients$Glucose, 0.75)
blood_q3 <- quantile(patients$BloodPressure, 0.75)
skinThickness_q3 <- quantile(patients$SkinThickness, 0.75)
insulin_q3 <- quantile(patients$Insulin, 0.75)
bmi_q3 <- quantile(patients$BMI, 0.75)
pedigree_q3 <- quantile(patients$Pedigree, 0.75)
age_q3 <- quantile(patients$Age, 0.75)
```

To find the values of Q1

```
pregnancies_q1 <- quantile(patients$Pregnancies, 0.25)
glucose_q1 <- quantile(patients$Glucose, 0.25)
blood_q1 <- quantile(patients$BloodPressure, 0.25)
skinThickness_q1 <- quantile(patients$SkinThickness, 0.25)
insulin_q1 <- quantile(patients$Insulin, 0.25)
bmi_q1 <- quantile(patients$BMI, 0.25)
pedigree_q1 <- quantile(patients$Pedigree, 0.25)
age_q1 <- quantile(patients$Age, 0.25)
```

And finally to find the values of Q2 (median), we can either use similar code or use the median function

```
pregnancies_q2 <- quantile(patients$Pregnancies, 0.5)
glucose_q2 <- quantile(patients$Glucose, 0.5)
blood_q2 <- quantile(patients$BloodPressure, 0.5)
skinThickness_q2 <- quantile(patients$SkinThickness, 0.5)
insulin_q2 <- quantile(patients$Insulin, 0.5)
bmi_q2 <- quantile(patients$BMI, 0.5)
pedigree_q2 <- quantile(patients$Pedigree, 0.5)
age_q2 <- quantile(patients$Age, 0.5)

print(paste("Mean for pregnancies :",pregnancies_mean))

## [1] "Mean for pregnancies : 3.84505208333333"

print(paste("Mean for glucose :",glucose_mean))

## [1] "Mean for glucose : 120.89453125"

print(paste("Mean for blood pressure :",bloodpressure_mean))
```

```
## [1] "Mean for blood pressure : 69.10546875"
print(paste("Mean for skin thickness :",skinThickness_mean))
## [1] "Mean for skin thickness : 20.5364583333333"
print(paste("Mean for insulin :",insulin_mean))
## [1] "Mean for insulin : 79.7994791666667"
print(paste("Mean for bmi :",bmi_mean))
## [1] "Mean for bmi : 31.992578125"
print(paste("Mean for pedigree :",pedigree_mean))
## [1] "Mean for pedigree : 0.471876302083333"
print(paste("Mean for age :",age_mean))
## [1] "Mean for age : 33.2408854166667"
print(paste("Standard Deviation for pregnancies :",pregnancies_sd))
## [1] "Standard Deviation for pregnancies : 3.36957806269887"
print(paste("Standard Deviation for glucose :",glucose_sd))
## [1] "Standard Deviation for glucose : 31.9726181951362"
print(paste("Standard Deviation for blood pressure :",blood_sd))
## [1] "Standard Deviation for blood pressure : 19.3558071706448"
print(paste("Standard Deviation for skin thickness :",skinth_sd))
## [1] "Standard Deviation for skin thickness : 15.9522175677276"
print(paste("Standard Deviation for insulin :",insulin_sd))
## [1] "Standard Deviation for insulin : 115.244002351338"
print(paste("Standard Deviation for bmi :",bmi_sd))
## [1] "Standard Deviation for bmi : 7.88416032037545"
print(paste("Standard Deviation for pedigree :",pedigree_sd))
## [1] "Standard Deviation for pedigree : 0.331328595012775"
print(paste("Standard Deviation for age :",age_sd))
## [1] "Standard Deviation for age : NA"
print(paste("Q3 for pregnancies :",pregnancies_q3))
```

```
## [1] "Q3 for pregnancies : 6"
print(paste("Q3 for glucose :",glucose_q3))
## [1] "Q3 for glucose : 140.25"
print(paste("Q3 for blood pressure :",blood_q3))
## [1] "Q3 for blood pressure : 80"
print(paste("Q3 for skin thickness :",skinThickness_q3))
## [1] "Q3 for skin thickness : 32"
print(paste("Q3 for insulin :",insulin_q3))
## [1] "Q3 for insulin : 127.25"
print(paste("Q3 for bmi :",bmi_q3))
## [1] "Q3 for bmi : 36.6"
print(paste("Q3 for pedigree :",pedigree_q3))
## [1] "Q3 for pedigree : 0.62625"
print(paste("Q3 for age :",age_q3))
## [1] "Q3 for age : 41"
print(paste("Q1 for pregnancies :",pregnancies_q1))
## [1] "Q1 for pregnancies : 1"
print(paste("Q1 for glucose :",glucose_q1))
## [1] "Q1 for glucose : 99"
print(paste("Q1 for blood pressure :",blood_q1))
## [1] "Q1 for blood pressure : 62"
print(paste("Q1 for skin thickness :",skinThickness_q1))
## [1] "Q1 for skin thickness : 0"
print(paste("Q1 for insulin :",insulin_q1))
## [1] "Q1 for insulin : 0"
print(paste("Q1 for bmi :",bmi_q1))
## [1] "Q1 for bmi : 27.3"
print(paste("Q1 for pedigree :",pedigree_q1))
```

```
## [1] "Q1 for pedigree : 0.24375"

print(paste("Q1 for age :",age_q1))

## [1] "Q1 for age : 24"

print(paste("Q2 for pregnancies :",pregnancies_q2))

## [1] "Q2 for pregnancies : 3"

print(paste("Q2 for glucose :",glucose_q2))

## [1] "Q2 for glucose : 117"

print(paste("Q2 for blood pressure :",blood_q2))

## [1] "Q2 for blood pressure : 72"

print(paste("Q2 for skin thickness :",skinThickness_q2))

## [1] "Q2 for skin thickness : 23"

print(paste("Q2 for insulin :",insulin_q2))

## [1] "Q2 for insulin : 30.5"

print(paste("Q2 for bmi :",bmi_q2))

## [1] "Q2 for bmi : 32"

print(paste("Q2 for pedigree :",pedigree_q2))

## [1] "Q2 for pedigree : 0.3725"

print(paste("Q2 for age :",age_q2))

## [1] "Q2 for age : 29"
```

C) Find missing values for each independent variable and fill them with median values. The missing values for independent variables in the dataset are coded 0.

```
# To first check for all missing values
colSums(patients == 0)

##   Pregnancies       Glucose BloodPressure SkinThickness       Insulin
##           111             5            35           227           374
##           BMI      Pedigree           Age     Diagnosis
##            11             0             0           500

# To replace the missing values with median
patients$Pregnancies[patients$Pregnancies==0]=median(patients$Pregnancies[pat
ients$Pregnancies>0])

patients$Glucose[patients$Glucose==0]=median(patients$Glucose[patients$Glucos
e>0])
```

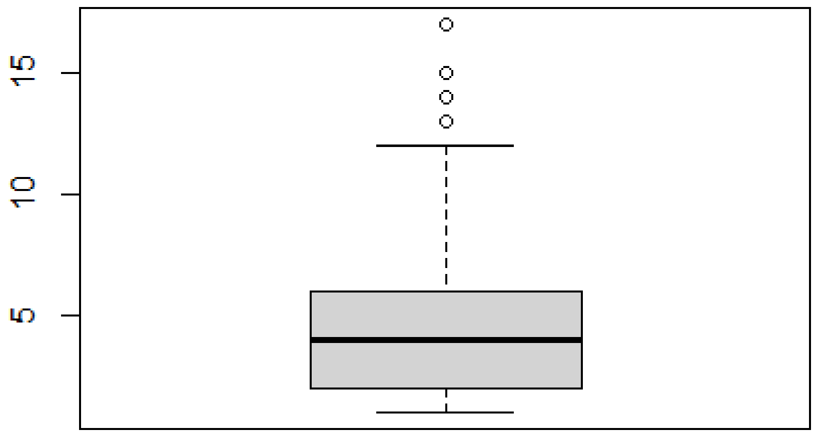
```
patients$BloodPressure[patients$BloodPressure==0]=median(patients$BloodPressu
re[patients$BloodPressure>0])

patients$SkinThickness[patients$SkinThickness==0]=median(patients$SkinThickne
ss[patients$SkinThickness>0])

patients$Insulin[patients$Insulin==0]=median(patients$Insulin[patients$Insuli
n>0])

patients$BMI[patients$BMI==0]=median(patients$BMI[patients$BMI>0])

patients$Pedigree[patients$Pedigree==0]=median(patients$Pedigree[patients$Ped
igree>0])

patients$Age[patients$Age==0]=median(patients$Age[patients$Age>0])

#To check for missing values or lack thereof
colSums(patients == 0)

##   Pregnancies       Glucose BloodPressure SkinThickness       Insulin 
##             0             0             0             0             0 
##           BMI      Pedigree           Age     Diagnosis 
##             0             0             0           500
```

There are no null values in Pedigree and Age, and the remaining independent variables have some null values. After Replacing the null values in independent variables with median, there are no more null values there are more missing values in insulin compared to other features.

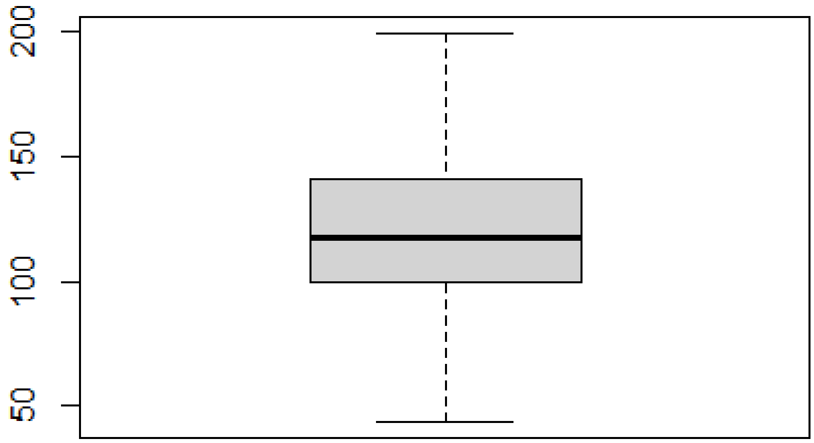
The preprocessing stage of our research also included checking for missing values and outliers.
In order to find missing values in the dataset, which were indicated as 0 in the dataset, we used *colSums(patients==0)* function. We then substituted the missing values with medians of corresponding variables.

D) Find outliers for each independent variable using the IQR rule

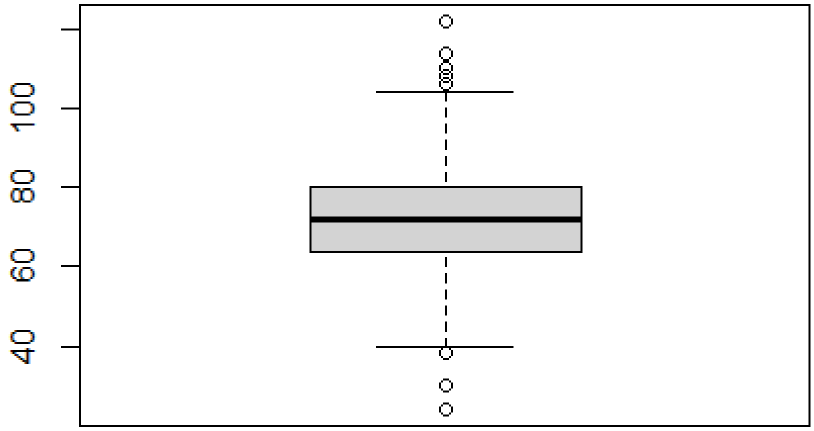
```
# To find the outliers using the IQR rule
preg_out1 = pregnancies_q3 + ((pregnancies_q3 - pregnancies_q1) * 1.5)
preg_out2 = pregnancies_q1 - ((pregnancies_q3 - pregnancies_q1) * 1.5)

glucose_out1 = glucose_q3 + ((glucose_q3 - glucose_q1) * 1.5)
glucose_out2 = glucose_q1 - ((glucose_q3 - glucose_q1) * 1.5)

blood_out1 = blood_q3 + ((blood_q3 - blood_q1) * 1.5)
blood_out2 = blood_q1 - ((blood_q3 - blood_q1) * 1.5)

skinThickness_out1 = skinThickness_q3 + ((skinThickness_q3 - skinThickness_q1
) * 1.5)
skinThickness_out2 = skinThickness_q1 - ((skinThickness_q3 - skinThickness_q1
) * 1.5)

insulin_out1 = insulin_q3 + ((insulin_q3 - insulin_q1) * 1.5)
insulin_out2 = insulin_q1 - ((insulin_q3 - insulin_q1) * 1.5)

bmi_out1 = bmi_q3 + ((bmi_q3 - bmi_q1) * 1.5)
bmi_out2 = bmi_q1 - ((bmi_q3 - bmi_q1) * 1.5)

pedigree_out1 = pedigree_q3 + ((pedigree_q3 - pedigree_q1) * 1.5)
pedigree_out2 = pedigree_q1 - ((pedigree_q3 - pedigree_q1) * 1.5)

age_out1 = age_q3 + ((age_q3 - age_q1) * 1.5)
age_out2 = age_q1 - ((age_q3 - age_q1) * 1.5)

# To check the outliers using boxplots
boxplot(x=patients$Pregnancies)
```



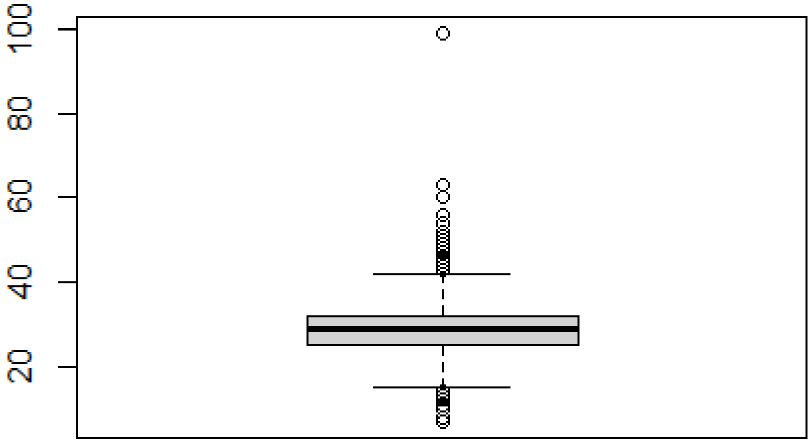
```
boxplot(x=patients$Glucose)
```



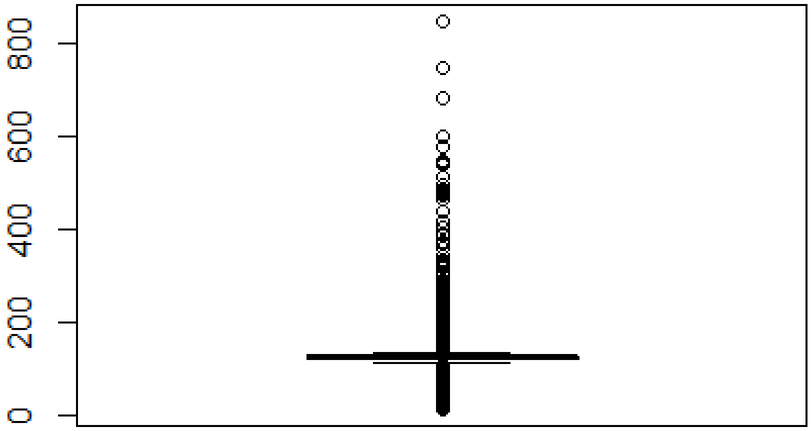
```
boxplot(x=patients$BloodPressure)
```



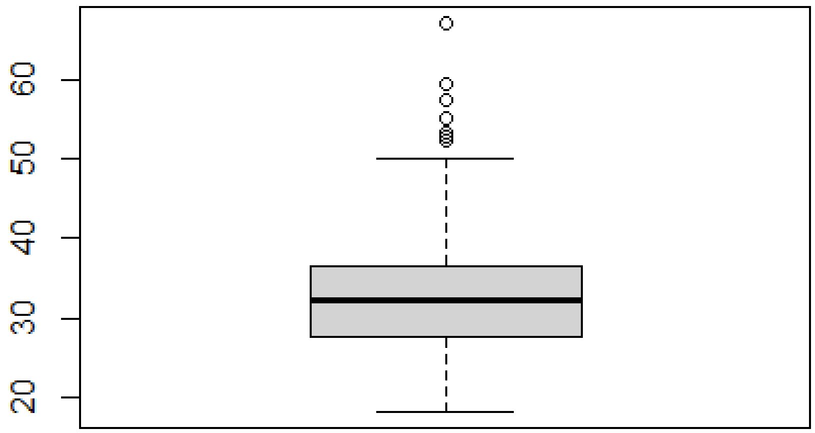
```
boxplot(x=patients$SkinThickness)
```



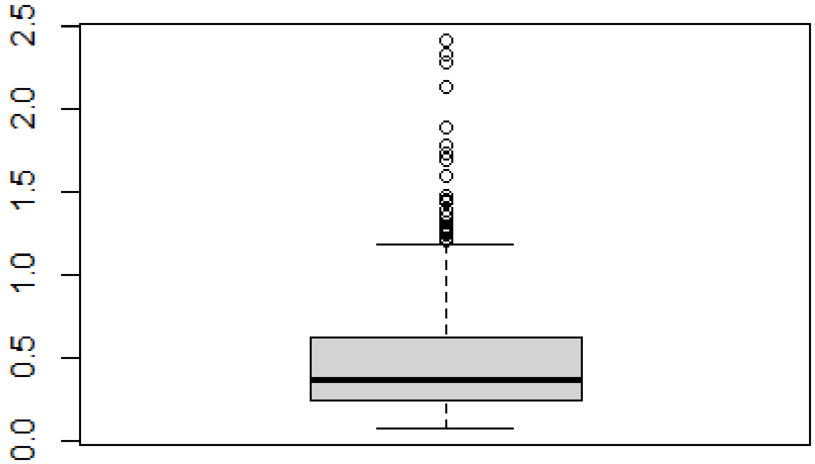
```
boxplot(x=patients$Insulin)
```



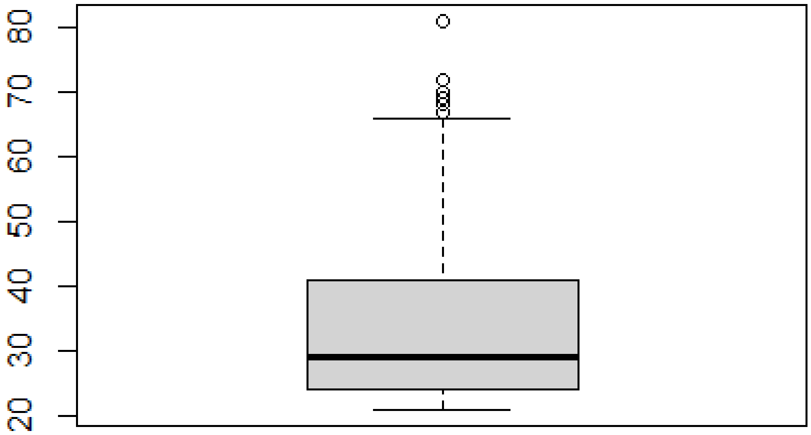
```
boxplot(x=patients$BMI)
```



```r
boxplot(x=patients$Pedigree)
```



```r
boxplot(x=patients$Age)
```



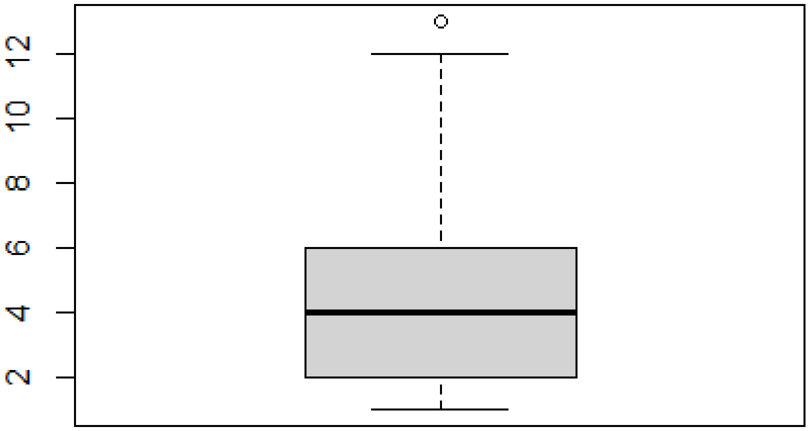
To now check and find what the outliers are for each of the features

```
# pregnancies
preg_out1

##  75%
## 13.5

preg_out2

##  25%
## -6.5

# Since glucose does not have any outliers (as observed in the box plot)

# blood pressure
blood_out1

## 75%
## 107

blood_out2

## 25%
##  35

# Skin Thickness
skinThickness_out1
```

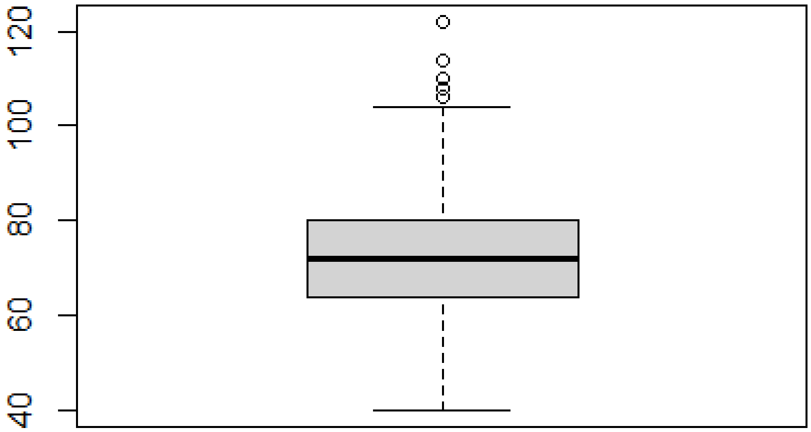
```
## 75%
##  80

skinThickness_out2

## 25%
## -48

# insulin
insulin_out1

##     75%
## 318.125

insulin_out2

##      25%
## -190.875

# BMI
bmi_out1

##   75%
## 50.55

bmi_out2

##   25%
## 13.35

# pedigree
pedigree_out1

## 75%
## 1.2

pedigree_out2

##   25%
## -0.33

# age
age_out1

##  75%
## 66.5

age_out2

##  25%
## -1.5
```

E) Replace outliers with median Values.

```
# for pregnancies

patients.clean <- patients %>%
  mutate(Pregnancies = replace(Pregnancies, Pregnancies > preg_out1, median(P
regnancies)))

# to check
preg_out <- boxplot.stats(patients.clean$Pregnancies)$out
preg_out

##  [1] 13 13 13 13 13 13 13 13 13 13

boxplot(x=patients.clean$Pregnancies)
```



```
# for glucose - There are no medians as we observed from the box plot
# to check
glucose_out <- boxplot.stats(patients.clean$Glucose)$out
glucose_out

## integer(0)

# for blood pressure
patients.clean <- patients %>%
  mutate(BloodPressure = replace(BloodPressure, BloodPressure > 104, median(B
loodPressure)))
patients.clean <- patients %>%
  mutate(BloodPressure = replace(BloodPressure, BloodPressure < 40, median(Bl
oodPressure)))

# to check
bloodp_out <- boxplot.stats(patients.clean$BloodPressure)$out
bloodp_out

##  [1] 110 108 122 110 108 110 106 106 106 114

boxplot(x=patients.clean$BloodPressure)
```



```
# For skin thickness
patients.clean <- patients %>%
  mutate(SkinThickness = replace(SkinThickness, SkinThickness > skinThickness
_out1, median(SkinThickness)))
patients.clean <- patients %>%
  mutate(SkinThickness = replace(SkinThickness, SkinThickness < skinThickness
_out2, median(SkinThickness)))

# to check
skinT_out <- boxplot.stats(patients.clean$SkinThickness)$out
skinT_out

##  [1] 45 47 11 47 11 10 60 13 13 54 51 56 14 13 50 44 12 46 44 13 44 54 14
7 50
```

```
## [26] 52 10 44 43 45 14 10 11 12 43 13 12 48 43 43  8 13 14 12 49 46 46 11
8 12
## [51] 63 12 45 13 48 13 10 45  7 52 49 43 14 47 99 46 11 50 45 14 13 13 47
12 48
## [76] 43 46 46 45 10 46 49 11 13 46 44 48
```

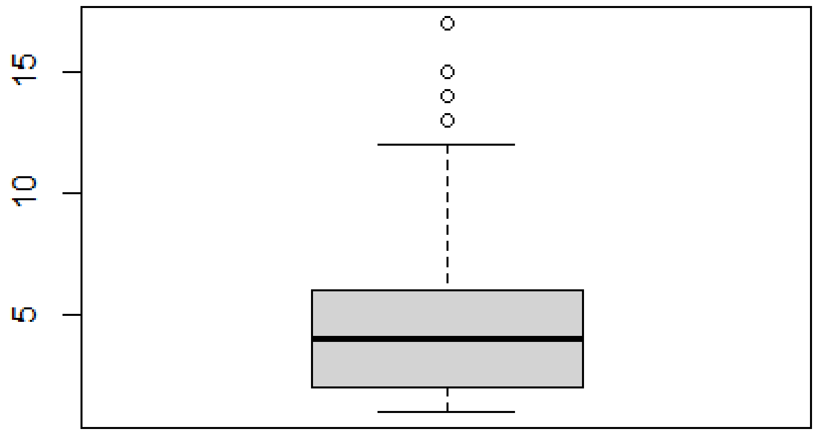
```
# for insulin
patients.clean <- patients %>%
  mutate(Insulin = replace(Insulin, Insulin > insulin_out1, median(Insulin)))
patients.clean <- patients %>%
  mutate(Insulin = replace(Insulin, Insulin < insulin_out2, median(Insulin)))

# to check
insulin_out <- boxplot.stats(patients.clean$Insulin)$out
insulin_out
```

```
##   [1]  94 168  88 543 846 175 230  83  96 235 146 140 110 245  54 192 207
70
##  [19] 240  82  36  23 300 342 304 110 142  38 100  90 140 270  71  71 110
176
##  [37]  48  64 228  76  64 220  40 152 140  18  36 495  37 175  51 100 100
99
##  [55]  94 145 168 225  49 140  50  92 325  63 284 204 155 485  94  53 105
285
##  [73] 156  78  48  55  92  23 495  58 160  94 210  48  99 318  44 190 280
87
##  [91] 175 271 478 190  56  32 744  53 370  37  45 192  88 176 194 680 402
55
## [109] 258 375 150  67  56  45  57 278 155 545 220  49  75  40  74 182 194
360
## [127] 215 184  42 105 148 180 205 148  96  85  94  64 140 231  29 168 156
68
## [145]  52  58 255 171 105  73 108  83  74  43 167  54 249 325 293  83  66
140
## [163] 465  89  66  94 158 325  84  75  72  82 182  59 110  50 285  81 196
415
## [181]  87 275  88 165 579 176 310  61 167 474 170  76  78 210 277 180 145
180
## [199]  85  60  50  14  70  92  64  63  95 210 105  71 237  60  56  49 105
36
## [217] 100 140 191 110  75 328  49 250 480 265  66  76 145 193  71  79  90
170
## [235]  76 210  86 105 165 326  66  82 105 188 106  65  56 210 155 215 190
56
## [253]  76 225 207 166  67 106  44 215 274  77  54  88  18 165  44 330  63
600
## [271] 156 140 230 185  25 293  41 272 182 158 194 321 144  15 160  54  90
183
## [289]  66  91  46 105 152 440 144 159 100 106  77 540  90 200  70 231 190
100
```

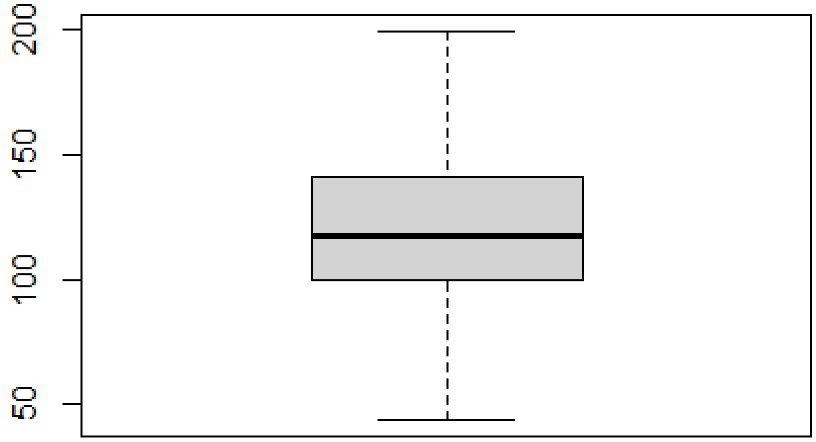
```
## [307] 168  49 240 265  45 105 205 180 180  95 480 155 200 100 335 160 387
22
## [325] 291 392 185 178 200 105 180  79 165 160 150  94 140 105  57 200  74
510
## [343] 110  16 180

# for BMI
patients.clean <- patients %>%
  mutate(BMI = replace(BMI, BMI > bmi_out1, median(BMI)))
patients.clean <- patients %>%
  mutate(BMI = replace(BMI, BMI < bmi_out2, median(BMI)))

# to check
bmi_out <- boxplot.stats(patients.clean$BMI)$out
bmi_out

## [1] 53.2 55.0 67.1 52.3 52.3 52.9 59.4 57.3

# for pedigree
patients.clean <- patients %>%
  mutate(Pedigree = replace(Pedigree, Pedigree > pedigree_out1, median(Pedigr
ee)))
patients.clean <- patients %>%
  mutate(Pedigree = replace(Pedigree, Pedigree < pedigree_out1, median(Pedigr
ee)))

# to check
pedigree_out <- boxplot.stats(patients.clean$Pedigree)$out
pedigree_out

##  [1] 2.288 1.441 1.390 1.893 1.781 1.222 1.400 1.321 1.224 2.329 1.318 1.2
13
## [13] 1.353 1.224 1.391 1.476 2.137 1.731 1.268 1.600 2.420 1.251 1.699 1.2
58
## [25] 1.282 1.698 1.461 1.292 1.394

# for age
patients.clean <- patients %>%
  mutate(Age = replace(Age, Age > age_out1, median(Age)))
patients.clean <- patients %>%
  mutate(Age = replace(Age, Age < age_out2, median(Age)))

# to check
age_out <- boxplot.stats(patients.clean$Age)$out
age_out

## [1] 69 67 72 81 67 67 70 68 69
```

To check the updated boxplots for observing the changes in outliers

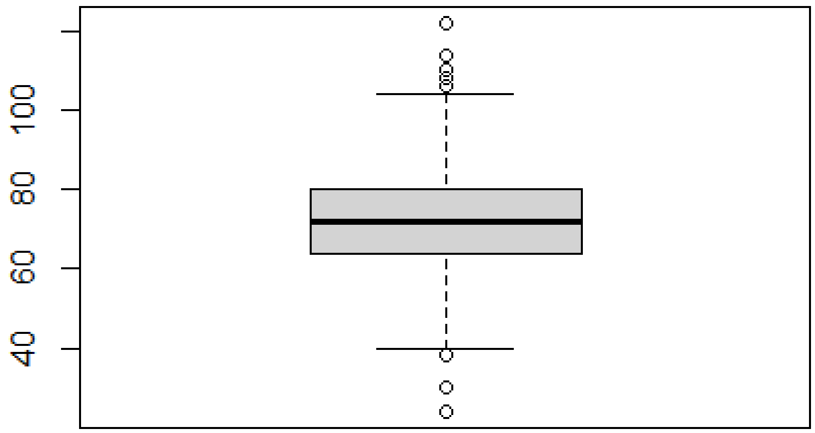
```
boxplot(x=patients.clean$Pregnancies)
```



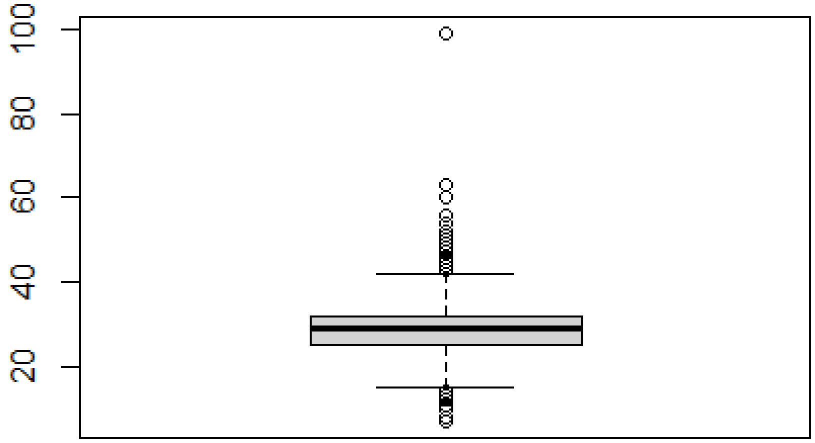
```
boxplot(x=patients.clean$Glucose)
```



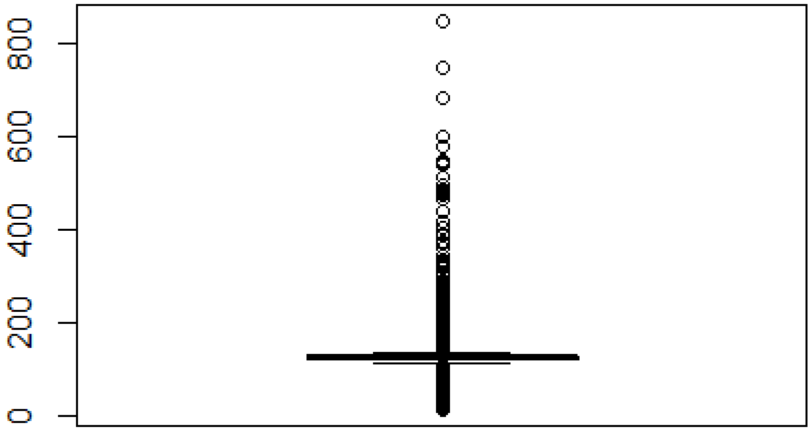
```
boxplot(x=patients.clean$BloodPressure)
```



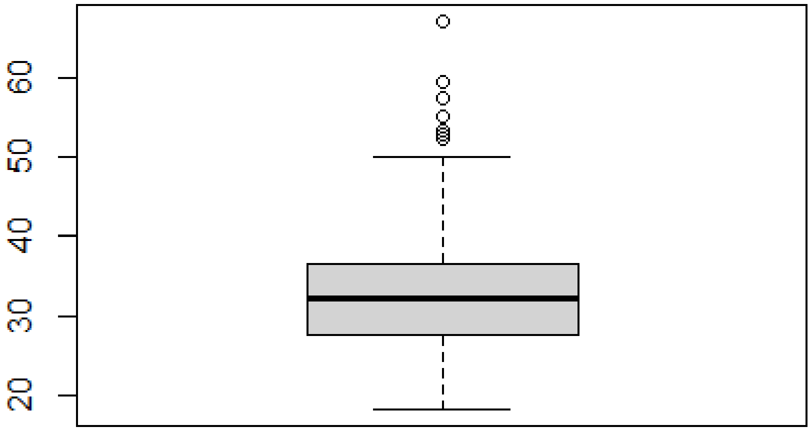
```
boxplot(x=patients.clean$SkinThickness)
```



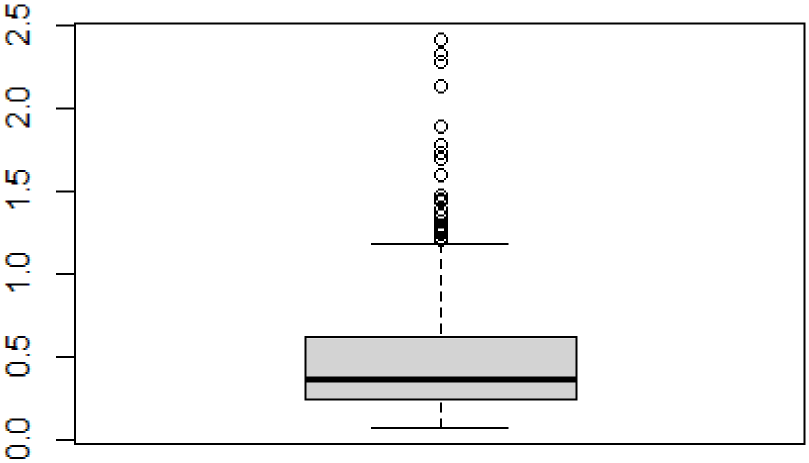
```
boxplot(x=patients.clean$Insulin)
```



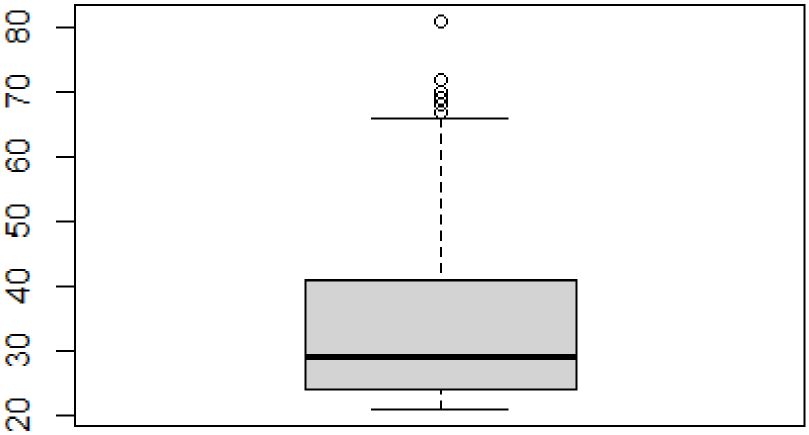
```
boxplot(x=patients.clean$BMI)
```



```
boxplot(x=patients.clean$Pedigree)
```

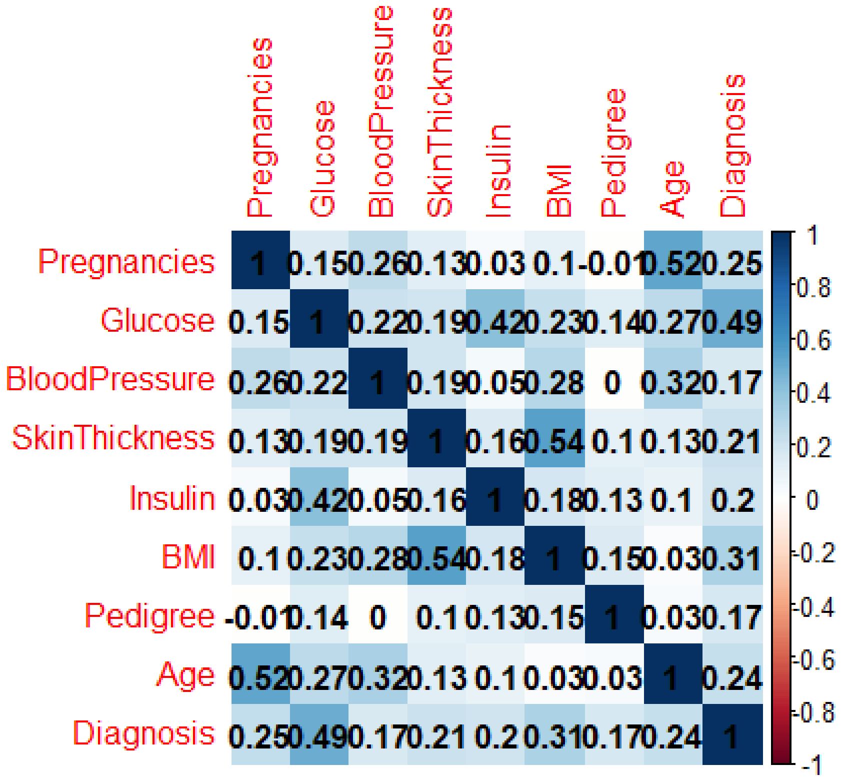


```
boxplot(x=patients.clean$Age)
```



F) Find the best performing variables / features using a correlogram.

```
library(corrplot)

## corrplot 0.92 loaded

P.cor <- cor(patients)
head(round(P.cor,2))

##               Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## Pregnancies          1.00    0.15          0.26          0.13    0.03 0.10
## Glucose              0.15    1.00          0.22          0.19    0.42 0.23
## BloodPressure        0.26    0.22          1.00          0.19    0.05 0.28
## SkinThickness        0.13    0.19          0.19          1.00    0.16 0.54
## Insulin              0.03    0.42          0.05          0.16    1.00 0.18
## BMI                  0.10    0.23          0.28          0.54    0.18 1.00
##               Pedigree  Age Diagnosis
## Pregnancies      -0.01 0.52      0.25
## Glucose           0.14 0.27      0.49
## BloodPressure     0.00 0.32      0.17
## SkinThickness     0.10 0.13      0.21
## Insulin           0.13 0.10      0.20
## BMI               0.15 0.03      0.31

corrplot(P.cor, method = "color", addCoef.col = "black")
```



Correlation is a statistical term which in common usage refers to how close two variables are to having a linear relationship with each other.

The most highly correlated features are (Diagnosis is our target function):

- Glucose=0.49
- BMI = 0.31
- Pregnancies = 0.25.

These are the features that we have chosen for our analysis of our first Regression Model.

G. Standardaize your features to Guassian distribution

```
# to standardize the data, we first split the data for training and testing u
sing a 70/30 split
dt = sort(sample(nrow(patients), nrow(patients)*.7))
#head(dt)
traindata <- patients[dt,]
testdata <- patients[-dt,]
#head(traindata)
# And to now standardize both the train and test data

xtrain <- as.data.frame(scale(traindata[1:8]))
xtest <- as.data.frame(scale(testdata[1:8]))

head(xtrain)

##   Pregnancies    Glucose BloodPressure SkinThickness    Insulin        BMI
## 1   0.5221283  0.8663014    -0.0261782   0.677738918 -0.1918371  0.1990446
## 2  -1.1428533 -1.2005422    -0.5252679  -0.009816118 -0.1918371 -0.8263200
## 3   1.1881210  2.0145479    -0.6916311  -0.009816118 -0.1918371 -1.3097061
## 4  -1.1428533 -1.0693141    -0.5252679  -0.697371154 -0.5320793 -0.6065990
## 5  -0.1438643  0.5054239    -2.6879899   0.677738918  0.2801116  1.5906108
## 6   0.1891320 -0.1835239     0.1401850  -0.009816118 -0.1918371 -0.9728006
##     Pedigree        Age
## 1  0.4882792  1.3856521
## 2 -0.3589822 -0.2152774
## 3  0.6264196 -0.1310180
## 4 -0.9238232 -1.0578719
## 5  5.5871967 -0.0467585
## 6 -0.8194504 -0.2995368

#head(xtest)
```

Before we build any model we will standardize features with gaussian distribution.

It is one of the **scaling technique where the values are centered around the mean with a unit standard deviation**.

So here all the independent variables are transformed in such a way that it will have properties of a standard normal distribution with mean = 0 and standard deviation = 1.

H) Create a logistic regression model (call it LRM1) using your best features. Describe your model.

```
# We will first set the seed so that we can have a constant out put
set.seed(1)

# we will then use 70% of the split data as training and the remaining 30% as
testing.
# Before running the LRM1 we will first add the target function (diagnosis) b
ack to our train and test data sets

xtrain$Diagnosis <- traindata$Diagnosis

xtest$Diagnosis <- testdata$Diagnosis

# We then ran the LRM1 model
options(scipen = 999)
LRM1 <- glm(Diagnosis~Glucose+Pregnancies+BMI, family = "binomial",data=xtrai
n)

summary(LRM1)

##
## Call:
## glm(formula = Diagnosis ~ Glucose + Pregnancies + BMI, family = "binomial"
,
##     data = xtrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.21811  -0.7224  -0.4022   0.7078   2.3844
##
## Coefficients:
##             Estimate Std. Error z value             Pr(>|z|)
## (Intercept)  -0.9250     0.1161  -7.967     0.00000000000296 ***
## Glucose       1.0257     0.1242   8.255 < 0.0000000000000002 ***
## Pregnancies   0.5202     0.1116   4.662     0.00001886037909 ***
## BMI           0.6652     0.1240   5.362     0.00000005854564 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 686.36  on 536  degrees of freedom
## Residual deviance: 502.88  on 533  degrees of freedom
## AIC: 510.88
##
## Number of Fisher Scoring iterations: 5
```

```
# to predict the outcome of the training set
predictTrain1 <- predict(LRM1, newdata= xtest, type = "response")
summary(predictTrain1)
```

```
##    Min. 1st Qu.   Median    Mean 3rd Qu.    Max.
## 0.01256 0.121163 0.25385 0.33693 0.55622 0.91275
```

Here we are considering our best features Glucose, Pregnancies and BMI. We are only considering these three features as they have correlation with the target variable Diagnosis. And also, Pregnancies and BMI has high correlation with independent variables age and skin Thickness respectively. So, we thought it's better to consider Glucose, Pregnancies and BMI for first model LRM1.

I) Create a classification report of your model.

```
threshold_0.5_1 <- table(xtest$Diagnosis, predictTrain1 > 0.5)
threshold_0.5_1
```

```
##
##     FALSE TRUE
##   0   142   17
##   1    26   46
```

Classification report is an evaluation metric to compare true labels of our data to predicted labels from our built model, so we can know how our model is performing. Classification report is built from confusion Matrix which has true negatives, true positives, false negatives and false positives. From above confusion matrix, we can see true positives as 46 and true negatives as 142

J) Describe your classification report (precision, recall, F1 score, and support).

```
err_metric=function(CM)
{
  TN =CM[1,1]
  TP =CM[2,2]
  FP =CM[1,2]
  FN =CM[2,1]
  precision =(TP)/(TP+FP)
  recall_score =(FP)/(FP+TN)
  f1_score=2*((precision*recall_score)/(precision+recall_score))
  accuracy_model  =(TP+TN)/(TP+TN+FP+FN)
  False_positive_rate =(FP)/(FP+TN)
  False_negative_rate =(FN)/(FN+TP)

  print(paste("Precision value of the model: ",round(precision,2)))
  print(paste("Accuracy of the model: ",round(accuracy_model,2)))
  print(paste("Recall value of the model: ",round(recall_score,2)))
  print(paste("False Positive rate of the model: ",round(False_positive_rate,
2)))

  print(paste("False Negative rate of the model: ",round(False_negative_rate,
2)))

  print(paste("f1 score of the model: ",round(f1_score,2)))
}
err_metric(threshold_0.5_1)

## [1] "Precision value of the model:  0.73"
## [1] "Accuracy of the model:  0.81"
## [1] "Recall value of the model:  0.11"
## [1] "False Positive rate of the model:  0.11"
## [1] "False Negative rate of the model:  0.36"
## [1] "f1 score of the model:  0.19"
```

- **Accuracy** – It tells how many times we predicted correctly as our true data labels. 0.81
- **Precision** - When model predicts true, how often it was right? 0.73
- **Recall -** When the class was actually true , how often or class get it right? 0.11
- **F1Score** – combination of both precision and recall. 0.19

Even though Accuracy and Precision are high, Recall and F1Score are less

Recall below mean that our models was able to detect less no of patients having diabetes when we have more, which is kind of bad, so it better to deploy models like SVM etc than going with logistic regression.

K) Create the accuracy score of your model. Describe the accuracy score.

```
accuracy_0.5 <- round(sum(diag(threshold_0.5_1))/sum(threshold_0.5_1),2)
sprintf("Accuracy is %s",accuracy_0.5)

## [1] "Accuracy is 0.81"

#Mis-classification error rate

MC_0.5 <- 1-accuracy_0.5
sprintf("Mis-classification error is %s",MC_0.5)

## [1] "Mis-classification error is 0.19"
```

- **Accuracy** – It tells how many times we predicted correctly as our true data labels. 0.81

Our model accuracy is good, that means our model has high prediction rate with correct diagnosis.

Misclassification rate is like measuring how messed up our model is which is 1-accuracy i.e 0.19

L) Create another logistic regression model (call it LRM2). Use all the independent features this time (instead of your best performing features).

```
LRM2 <- glm(Diagnosis ~ Pregnancies + Glucose +Insulin + BloodPressure + Skin
Thickness + BMI + Age + Pedigree, family="binomial", data=xtrain)

summary(LRM2)
```

```
##
## Call:
## glm(formula = Diagnosis ~ Pregnancies + Glucose + Insulin + BloodPressure
+
##     SkinThickness + BMI + Age + Pedigree, family = "binomial",
##     data = xtrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6678  -0.7531  -0.4332   0.7453   2.3256
##
## Coefficients:
##                Estimate Std.   Error  z value         Pr(>|z|)
## (Intercept)    -0.85207     0.11281  -7.553 0.0000000000004238 ***
## Pregnancies     0.42742     0.12523   3.413           0.000642 ***
## Glucose         1.04188     0.13390   7.781 0.0000000000000719 ***
## Insulin         0.01283     0.11860   0.108           0.913888
## BloodPressure  -0.14544     0.12065  -1.205           0.228017
## SkinThickness   0.09712     0.14050   0.691           0.489412
## BMI             0.45679     0.14116   3.236           0.001212 ***
## Age             0.16895     0.12724   1.328           0.184240
## Pedigree        0.24586     0.11301   2.175           0.029594 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 619.63  on 536  degrees of freedom
## Residual deviance: 519.76  on 528  degrees of freedom
## AIC: 537.76
##
## Number of Fisher Scoring iterations: 4
```

```
predictTrain2 <- predict(LRM2, newdata= xtest, type = "response")
summary(predictTrain2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01496 0.11306 0.27422 0.35069 0.56120 0.98783
```

```
threshold_0.5_2 <- table(xtest$Diagnosis, predictTrain2 > 0.5)
threshold_0.5_2
```

```
##
##      FALSE TRUE
##   0   131   17
##   1    34   49
```

```
accuracy_0.5 <- round(sum(diag(threshold_0.5_2))/sum(threshold_0.5_2),2)
sprintf("Accuracy is %s",accuracy_0.5)
```

```
## [1] "Accuracy is 0.78"
```

```
MC_0.5 <- 1-accuracy_0.5
sprintf("Mis-classification error is %s",MC_0.5)
```

```
## [1] "Mis-classification error is 0.22"
```

```
err_metric=function(CM)
{
  TN =CM[1,1]
  TP =CM[2,2]
  FP =CM[1,2]
  FN =CM[2,1]
  precision =(TP)/(TP+FP)
  recall_score =(FP)/(FP+TN)

  f1_score=2*((precision*recall_score)/(precision+recall_score))
  accuracy_model  =(TP+TN)/(TP+TN+FP+FN)
  False_positive_rate =(FP)/(FP+TN)
  False_negative_rate =(FN)/(FN+TP)

  print(paste("Precision value of the model: ",round(precision,2)))
  print(paste("Accuracy of the model: ",round(accuracy_model,2)))
  print(paste("Recall value of the model: ",round(recall_score,2)))
  print(paste("False Positive rate of the model: ",round(False_positive_rate,
2)))

  print(paste("False Negative rate of the model: ",round(False_negative_rate,
2)))

  print(paste("f1 score of the model: ",round(f1_score,2)))
}
err_metric(threshold_0.5_2)
```

```
## [1] "Precision value of the model:  0.74"
## [1] "Accuracy of the model:  0.78"
## [1] "Recall value of the model:  0.11"
## [1] "False Positive rate of the model:  0.11"
## [1] "False Negative rate of the model:  0.41"
## [1] "f1 score of the model:  0.2"
```

Finally, for our second logistic regression model in addition to highly correlated features: pregnancies, glucose and BMI, we used all independent features listed in our dataset, those were: Insulin, SkinThickness, Age, BloodPressure, Pedigree.

Based on p-values, we can see that, as mentioned before, Pregnancies, Glucose, BMI are significant features (p-value < 0.05), however we see that Pedigree can also be considered significant as its p-value is less than 0.05.
The Null deviance or fit dependent variable for the model is 619.63
The Residual deviance/ fit dependent variable with all independent variable is 519.76. This big difference between Null deviance and Residual deviance means that our model significantly improved.
Akaike's Information Criterion is 537.76 with 78% of Accuracy and 74% Precision rates. The less the AIC value the better the model.

F1 score – combination of precision and recall is 0.20.

M) Compare the two models (LRM1 and LRM2) based on the classification report and accuracy score. Which one is a better model? Why?

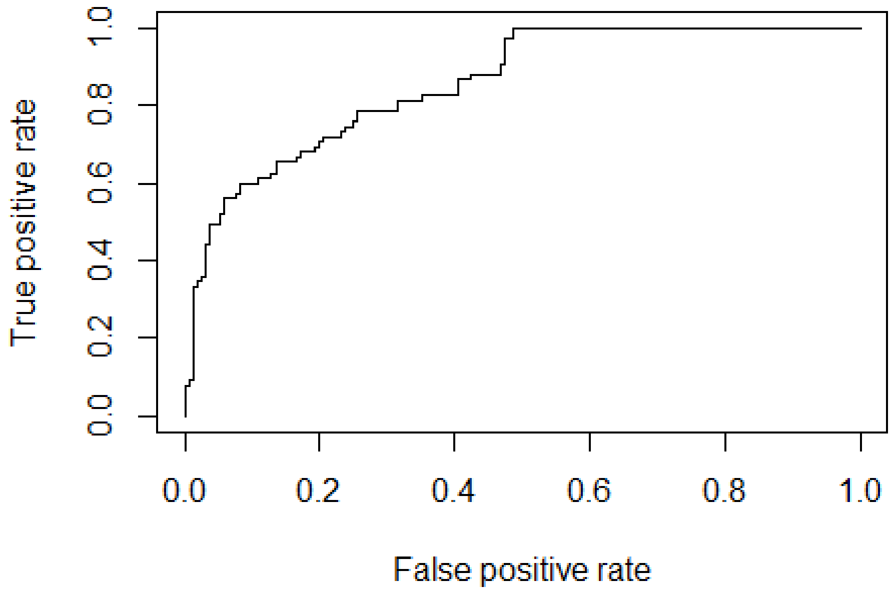
```
AIC(LRM1,LRM2)
```

```
##      df      AIC
## LRM1  4 510.88
## LRM2  9 537.76
```

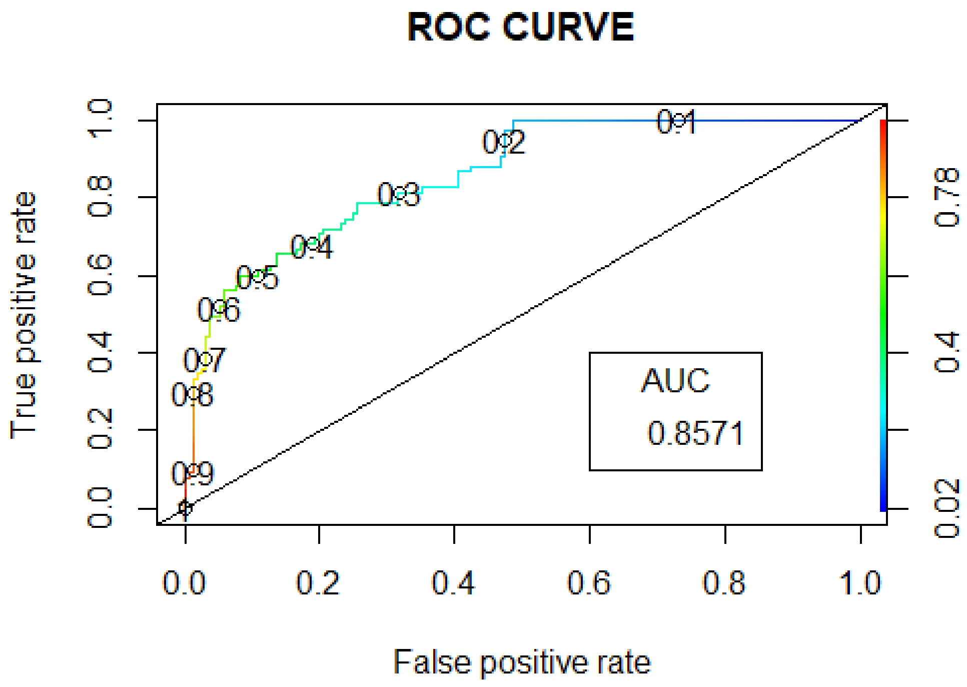
```
library(caTools)
library(ROCR)

ROCPred <- prediction(predictTrain1, xtest$Diagnosis)
ROCPer <- performance(ROCPred, measure = "tpr",
                             x.measure = "fpr")

auc <- performance(ROCPred, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.857094
```

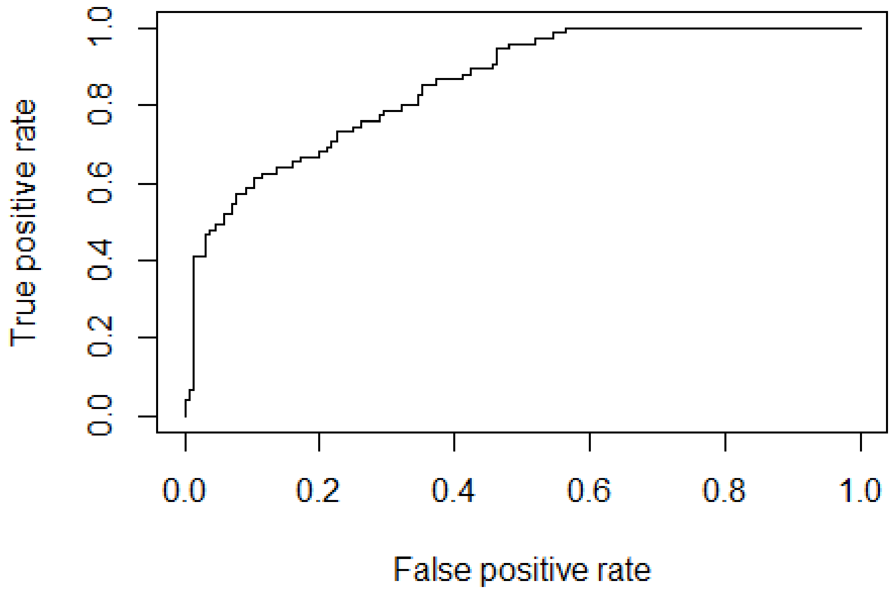
```
plot(ROCPer)
```



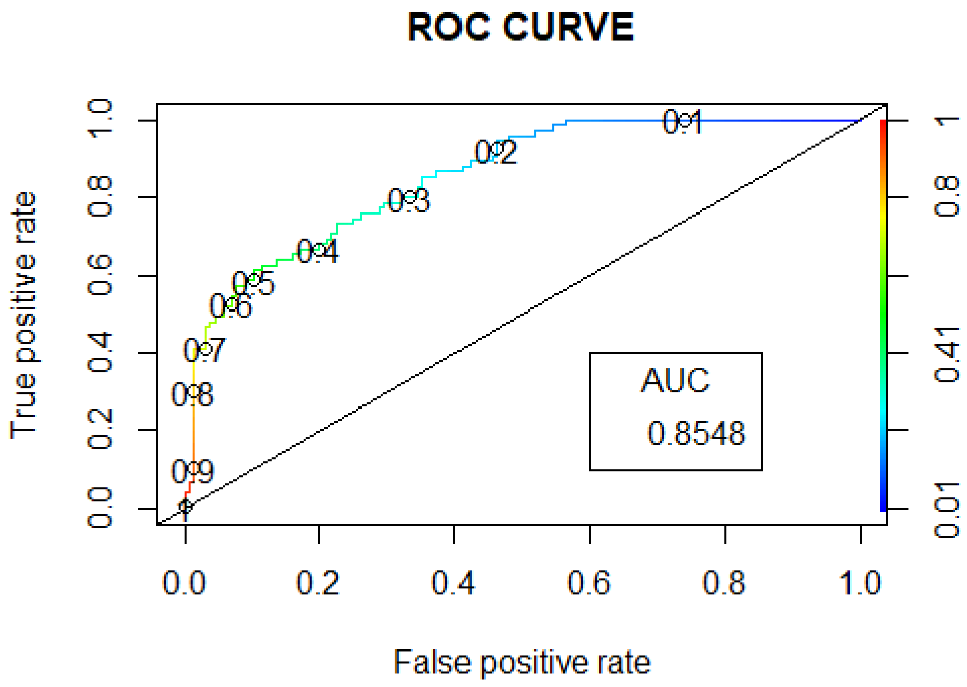
```
plot(ROCPer, colorize = TRUE,
     print.cutoffs.at = seq(0.1, by = 0.1),
     main = "ROC CURVE")
abline(a = 0, b = 1)

auc <- round(auc, 4)
legend(.6, .4, auc, title = "AUC", cex = 1)
```



```
library(caTools)
library(ROCR)

ROCPred <- prediction(predictTrain2, xtest$Diagnosis)
ROCPer <- performance(ROCPred, measure = "tpr",
                               x.measure = "fpr")

auc <- performance(ROCPred, measure = "auc")
auc <- auc@y.values[[1]]
auc

## [1] 0.8547863

plot(ROCPer)
```



```
plot(ROCPer, colorize = TRUE,
     print.cutoffs.at = seq(0.1, by = 0.1),
     main = "ROC CURVE")
abline(a = 0, b = 1)

auc <- round(auc, 4)
legend(.6, .4, auc, title = "AUC", cex = 1)
```


ROC CURVE
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
AUC
0.8548
1
0.8
0.41
0.01

We start off by comparing deviance differences of models. Null deviance is the value when you only have intercept in your equation with no variables and Residual deviance is the value when you are taking all the variables into account. The greater the difference between Null deviance and Residual deviance the better the fit of the model. For the first model the deviance difference was 184 and for the second model it was 172. Based only on that we can say that the first model is performing better.

Next, let's compare accuracy scores. The greater the accuracy value the better the model can predict diagnosis. The first model had an accuracy of 81%, the second model = 78%. Again, the first model has greater accuracy, and it is performing better.

Next, compare Akaike's Information Criterion or AIC, which is an estimator of prediction error and thereby relative quality of statistical models for a given set of data. The lower AIC value the better the model! We had LRM1 = 510.88, LRM2 = 537.76. The first model shas lower AIC value, hence it is performing better than the second value.

Area under the ROC curve - absolute value of quality of prediction, with maximum of 1 meaning perfect prediction. LRM1 = 0.8705. LRM2 = 0.8754. The difference between them is about 0.001, which is not a huge difference, however if we look at the ROC curve, we see that the first model has 0.61-0.62 true positive rate ay a threshold of 0.5, while ROC curve for the second model has 0.59-0.60 true positive rate at a threshold of 0.5. This means that our first model has a better-quality prediction than the second model.

N) What would be your suggestions for further improving the accuracy of your chosen model?

Based on the models that we made and the results we had found when running said models, we concluded that:

Increasing the data collected and to also make sure to diversify the sample collected to represent more demographics to better resemble the population would yield better results. This we believe is true not only for our models but in real world applications as well.

We also suggest that finetuning the parameters used could potentially result in the model providing better predictions. We have observed such occurrences when using different threshold values. In the end, we used a threshold value of 0.5 as it is the more recommended value. But when using different values, we observed that not only did the models perform at varying levels but also at times the performance levels shifted as my teammate eluded on the AUC curves where the second model performed better than the first in few of the values, but this leads us to assume that further investigation and analysis could yield better performance from our models.

And finally, our group strongly suggests that better feature selection would aid in improving the accuracy of the chosen model. Since we had only included the top 3 highly correlated features for our selected model, we believe that if we diversified and included other features that may not be as closely correlated to our target function and try different combinations of the features, this could yield with better prediction performance.

O) What would be the pitfalls or weaknesses of your model if the hospital decided to deploy it to predict diabetes?

The first one is that there could be a possible ethical pitfall.

In the research that we have done, we observed this being mentioned a couple of times. One of the main ethical pitfalls experienced by medical personnel of any health institution is that their diagnosis of their patients is always derived from the data they collect about the patient. Sometimes more data is collected to help clarify or better classify a patient's symptoms to aid in a final diagnosis but even with that, they could end up wrongfully diagnosing a patient, which is common in the medical field. If the hospital were to implement this model to help aid their medical personnel in concluding a patient's diagnosis and this has a negative outcome either because the model provided the wrong prediction or conclusion or due to a wrongful data entry, the fault would still be on the medical doctor that mis-diagnosed the patient and hence might have either caused more issues and damage to the patient or was unhelpful. The other side of this, is that if the doctor provides sufficient data to the model and the model predicts a certain diagnosis and the doctor does not agree with this conclusion and ends up being wrong, then again, they would be held accountable for not only wrongful diagnosis of the patient but also not following protocol or the results provided from the ML model.

The next pitfall we want to point out is that the model would not predict effectively to patients that might fall into the, what we considered as, outliers. We say this because as discussed earlier, we have replaced these outliers with the median values of each of these features. This reduces the accuracy of the model to predict a correct diagnosis of any patients that might have values that fall into said outliers.

The final pitfall could be that the model would only be effective on patients with demographics, races or areas that the sample was taken from. This could cause a big bias to communities that were not included in the collected sample, and this could result in wrongful diagnosis if the model is used on such patients. For example, if the training and test data for the algorithm during development predominantly involves medical cases of middle-aged Westerners, its medical diagnosis might be less accurate regarding people with East Asian ancestry. Due to different dietary habits or genetic predispositions, the latter might be prone to developing the disease due to different causes or data values compared with the former. Thus, the machine learning algorithm might face difficulties at providing a reliable medical diagnosis for a patient coming from Eastern Asia.

P) If you were to present your analysis and findings to the CEO of the hospital, what would be your top five key points?

The first point we would like to mention is that when observing the different features that are included in the diagnosis of a patient as to whether they have gestational diabetes, we saw that not all the features have a direct or normal correlation to the diagnosis. This means that not all the data collected from the patients is equally useful for the model's prediction accuracy. So, we advise that you pay more attention to these values when conducting the health screenings of your patients.

The second point we would like to make is that based on the models we tested, as mentioned above on our first point, not all of the features are needed to better predict the diagnosis of a patient using our ML model. Unlike medical personnel this model would not necessarily perform better when provided with more features but would be more effective when a select few of the features are used. As mentioned previously in our presentation, we created and tested two models, the first only containing three of our more highly correlated features and the second containing all the provided features. And we have seen that the first model performs better in predicting the diagnosis of the patients more accurately. Hence, you should always pay more attention to the features provided to the ML model and make sure to select what would aid the model to perform better and not to add more features with the expectation of better results.

The third point we would like to make is that because we developed our models based on some assumptions and redacted the data based on those assumptions, we must be very vigilant when deploying the ML model and the patients we are using them for. For instance, one of the adjustments we had made was to replace all the data outliers with the median value. When this was done, some of the data points were ultimately altered; hence the ML model would only best predict and work accurately for patients that were in the adjusted data used to create said model.

The fourth point we would like to make is that this ML model could be used for potential future diagnosis of patients that have certain values in their data that is similar to patients that have had a positive result to having gestational diabetes in the past. This could be implemented to use caution and better prepare patients that plan for pregnancies of for patients that recently have gotten pregnant. If their data shows that they are more prone to having or developing gestational diabetes, then their physicians could warn them to take better care and lookout for any signs which could help prevent said patients from developing gestational diabetes.

And finally, our fifth point, going off the fourth, is that by using this model you would also be able to create a treatment plan that is more individualized. You could tackle issues with patients that have high values in certain features in their data that have proven to be more correlated and could result in a positive diagnosis of gestational diabetes. You can develop on the existing model by documenting the treatments used for your patients to find which treatments were most effective to help the patient in overcoming the disease. For instance, if a patient is diagnosed as having gestational diabetes and one can observe that the patient’s glucose levels are high and when comparing all other features in the data and they do not show any differences with other patients that do not have gestational diabetes, this could be indicative of the cause of the disease and a good insight as to what type of treatment would best suit this patient. Using this ML model, you would not only be able to foresee and prevent your patients from developing gestational diabetes but also in the case that they do develop this disease, you would be better equipped and prepared to treat your patients in the most effective way. This would not only help save costs of treatments both for your patients and your intuition, but it would result in high patient satisfaction and a much smoother working and operating institution overall.

Thank you.