Chua Feng Qing: EDA, ML　　**Justin Kwek:** Dataprep, ML
Derek Chang: EDA, ML　　**Koong Kwang Hwee:** Backgrd, EDA

1 Introduction

1.1 Background

Diabetes is a disease due to the body's inability to regulate blood sugar (glucose). The WHO estimates that 1.5 million deaths were directly caused by diabetes in 2019 [1]. In the U.S, diabetes is most common among Native Americans [2]. Thus, we analyse medical records of Pima Indian (native) women at least 21 years old. The data's source is originally from the National Institute of Diabetes and Digestive and Kidney Diseases [3]. The data "PimaIndiansDiabetes2" is extracted from the mlbench package in CRAN. The data contains medical examination variables of the women and a class variable for diabetes test. The class variable indicates a positive test for diabetes between 1 to 5 years from the examination, or a negative test for diabetes after 5 or more years from the examination.

1.2 Objective & Questions

Early detection can help prevent diabetes, as pre-diabetes is reversible. We want to detect diabetes outcome using early physical indicators as predictors.
From our data, we aim to investigate:

1. *How many natural clusters do the predictor variables belong to?*

Hypothesis: Two natural clusters if data is highly predictive for diabetes (two level outcome).
We then explore the distinction between clusters and find out:

2. *Can we classify which cases will be diabetic based on their physical indicators?*

2 Data Preparation

Data "PimaIndiansDiabetes2" was extracted instead of "PimaIndiansDiabetes" as the latter contains physical impossibilities which are replaced as NA in "PimaIndiansDiabetes2". Upon inspection, the data contains 9 variables. 8 are numeric predictors and 1 is the two-level factor outcome, "diabetes". The meaning of each variable is compiled in Appendix.
We obtain summary statistics for each variable (in Appendix) and there are 768 total cases, their distribution by outcome shown in Table 1. We also note the number of NA cases for each variable in Table 2.

diabetes	count	% of cases
neg	500	65.1
pos	268	34.9

Table 1: Original distribution of diabetes outcomes

insulin	triceps	pressure	mass	glucose
374	227	35	11	5

Table 2: No. of NA cases for each variable

First, there is a class imbalance in positive to negative cases of 7:13. This indicates some difficulty in predicting positive class as the rarer event has smaller proportion in training data. The imbalance is not severe, so we do not under-sample the negative cases. Also, since a classification model that blindly classifies every case as negative will have 65% accuracy, we establish a baseline performance of 65% accuracy.
We drop all NA cases to enable unbiased investigation of relationships between predictor variables. A total of 376 cases were dropped and the sample size left is 392. The distribution of diabetes outcome remained almost the same (67% neg) corresponding to class imbalance of 1:2.

Finally, diabetes is closely related to insulin resistance [4] and can be indicated by Glucose/Insulin ratio. We then add a 9th feature G/I ratio to the data, transformed by dividing glucose by insulin.

3 Exploratory Data Analysis

3.1 Variable Distribution & Relationship with Diabetes Outcome

To investigate the distribution of variables, we group them by diabetes outcome and plot a boxplot + violin for each predictor variable. First, we observe that a few variables contain extreme outliers, which may be data entry/measurement errors or otherwise unreasonably skew the distribution. An example is a case with insulin>800uU/mL, likely to be an error. We remove several of such outliers (see Appendix), while keeping those not too extreme as they may be natural variations and removing may lower generalisability of our model. We also want to maintain sufficient sample size. After dropping 10 extreme outliers, the final sample size is 382.

From these plots we can also visualise the general relation of predictors variables with diabetes outcome. We see a difference in the distributions for negative vs positive outcomes.

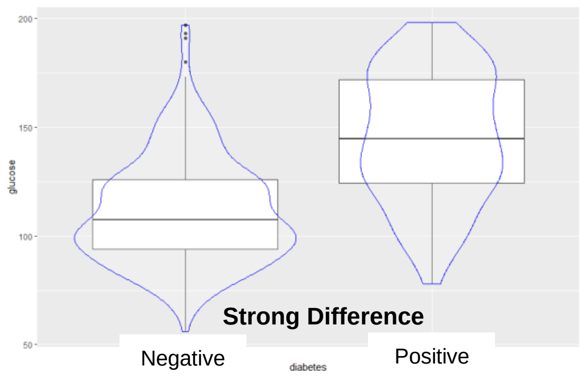


*Figure 1: Boxplot of **glucose** variable.*

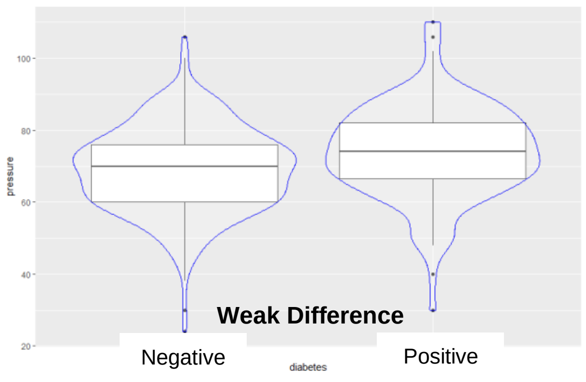


*Figure 2: Boxplot of **pressure** variable.*

For example, from Fig 1, the median glucose for negative is clearly lower than even the lower quartile for positive, suggesting a strong likelihood for women with higher glucose level to develop diabetes. This makes glucose a significant predictor when classifying for diabetes outcome. In contrast, Fig 2 shows a similar distribution of pressure for both groups, so pressure is not as strong a predictor for diabetes class.

We further our analysis by charting the mean outcome which is the probability of positive diabetes for each variable case. An example is shown in Fig 3 below.

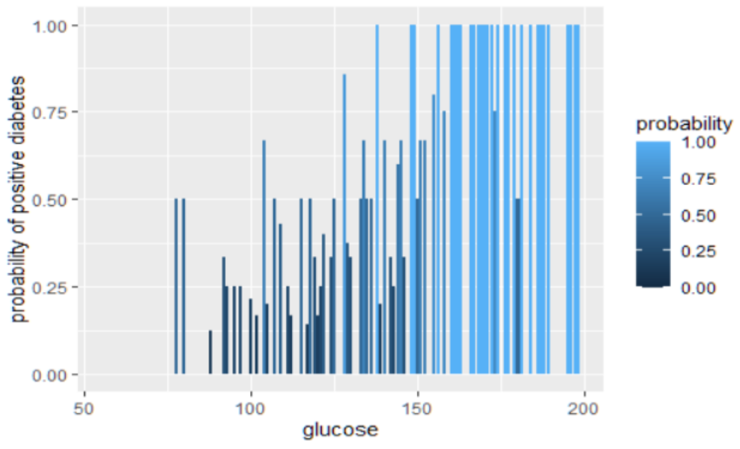


*Figure 3: Probability distribution for **glucose.***

The overall trend shows that probability increases as glucose increases.

It supports the conclusion drawn from Fig 3, that glucose is a strong predictor for diabetes class.

From our EDA, we summarise the association of variables with diabetes class in Table 3 below. We also note that other than G/I ratio which tends to be lower for positive cases, all other variables tend to be higher for positive cases.

Degree of association with diabetes class	Variables
Low	Pressure
Medium	Pregnant, Triceps, Insulin, Mass, Pedigree
High	Glucose, Age, G/I ratio

Table 3: Summary of predictor variables' degree of association with diabetes class.

Next, we check the pairwise correlation of variables with scatterplot and correlation plot.(Appendix) From the scatterplot, we can observe clusters of pink (positive) that are distinct from regions of green (negative), which tells us which variables can separate the class clearly. From the correlation coefficients, we find the highly correlated pairs to be glucose & insulin, triceps & mass, pregnant & age.

3.2 Hierarchical Clustering

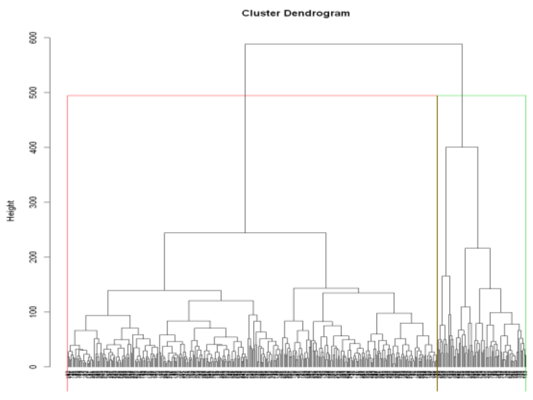


Figure 4: Cluster Dendrogram, 2 natural clusters

Finally, we test our hypothesis of two natural clusters corresponding to diabetes outcome, using hierarchical clustering. We observe 2 distinct clusters (red vs green), at the highest level, across all cases. Cluster 1 has 308 cases and Cluster 2 has 74 cases. Positive cases make up 28% of Cluster 1 and 50% of Cluster 2. Comparing this to the overall dataset which has 35% positive cases, Cluster 1 has higher proportion of negative cases and Cluster 2 has much higher proportion of positive cases. Though not conclusive, it shows that the clusters have separation by diabetes class, supporting hypothesis of two natural clusters, allowing us to proceed with classification using our predictor variables.

4 ML Classification

4.1 Train:Test Split

As the sample size is only 382, the Train:Test split ratio must be optimized as the variance in test data will be too large for accurate testing. We achieved this by running different split ratios with 500 seed values to obtain lowest uncertainty use logistic regression as a testing model.

Train : Test	80:20	70:30	60:40	50:50
Accuracy Uncertainty Range	23.6%	19.3%	15.9%	15.3%

Table 4: Different accuracy range for different split ratio

The model is able to classify with same accuracy even with lower train size and higher test size. This is most likely due to the saturation of model fitting; hence using a larger test data with less variance is more suitable. The 60:40 split was decided as further uncertainty improvement is not significant. We also kept test data consistent to compare across models.

4.2 Model Tests & Comparison

We attempt Logistic Regression, kNN classification and SVM as they are useful for classification of binary diabetes outcome. All predictor variables were used except pressure, which has low association with diabetes outcome. For kNN, the data is first scaled with min-max normalisation and the best k=20. SVM Polynomial kernel was also tested as we suspected possible non-linear decision surface.

	Logistic Regression	**kNN Classification**	**SVM Linear**	**SVM Polynomial**
Accuracy	75%	77%	76%	76%
Confusion Matrix	$\begin{matrix}86 & 22\\16 & 29\end{matrix}$	$\begin{matrix}94 & 27\\8 & 24\end{matrix}$	$\begin{matrix}87 & 22\\15 & 29\end{matrix}$	$\begin{matrix}90 & 25\\12 & 26\end{matrix}$

Table 5: Summary of models' performance

All models beat the baseline 65% accuracy, indicating that all can classify diabetes outcome with some success. There is no obvious winner but kNN has the best accuracy of 77%, beating baseline by additional 12%. It also has the lowest False Positive (FP) rate of 7.8%. Logistic regression and SVM linear both have the lowest False Negative (FN) rate of 43%. The models all have much higher FN rate than FP rate, indicating a higher difficulty in classifying positive cases as they tend to predict negative, possibly due to the class imbalance.

4.3 Final Model

Since the models are less likely to predict positive cases, the low FP rate on kNN could be due to overfitting on the class imbalanced data. Thus, we select SVM linear as our final model due to its lowest FN rate. False negative classifications will lead to undetected cases which defeats the purpose of our model to prevent future diabetes, so we want the lowest FN rate. SVM is also suitable as the predictor variables are all quantitative, so there is no categorical predictor affecting margin calculation.

5 Conclusion

First, using Hierarchical Clustering, we found 2 natural clusters in the dataset, which seem to have distinction in proportion of positive diabetes cases. Second, using SVM linear we are able to classify if a case will develop diabetes within the next 5 years with 76% success rate on the test data.

A limitation of our model is that the FN rate is still considered high even in SVM linear. The class imbalance causing high FN rate may be improved by oversampling the positive cases. Another limitation is the small sample size causing high variance in accuracy which makes it difficult to compare models. An improvement could be to create an artificially larger sample size by imputing the NA cases using regression methods to keep our NA rows. More causal predictors like cholesterol level can be obtained from other datasets to increase predictive power.

Finally, our findings may possibly enable some extrapolation to the general population outside of Pima Indians.

References

[1] https://www.who.int/news-room/fact-sheets/detail/diabetes

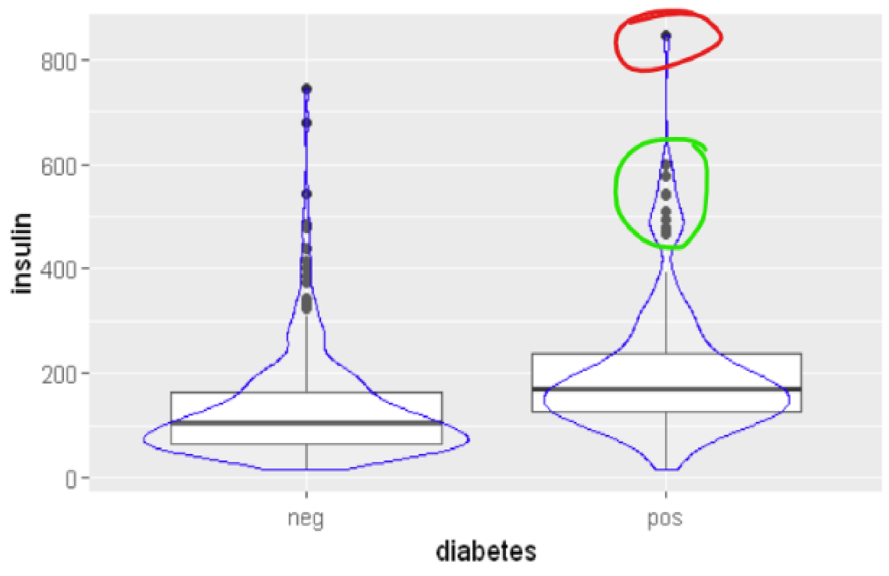
[2] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3830901/

[3] Grace Whaba, Chong Gu, Yuedong Wang, and Richard Chappell (1995), Soft Classification a.k.a. Risk Estimation via Penalized Log Likelihood and Smoothing Spline Analysis of Variance, in D. H. Wolpert (1995), The Mathematics of Generalization, 331-359, Addison-Wesley, Reading, MA.

[4] https://medicine.musc.edu/departments/family-medicine/research/rcmar/insulin-resistance

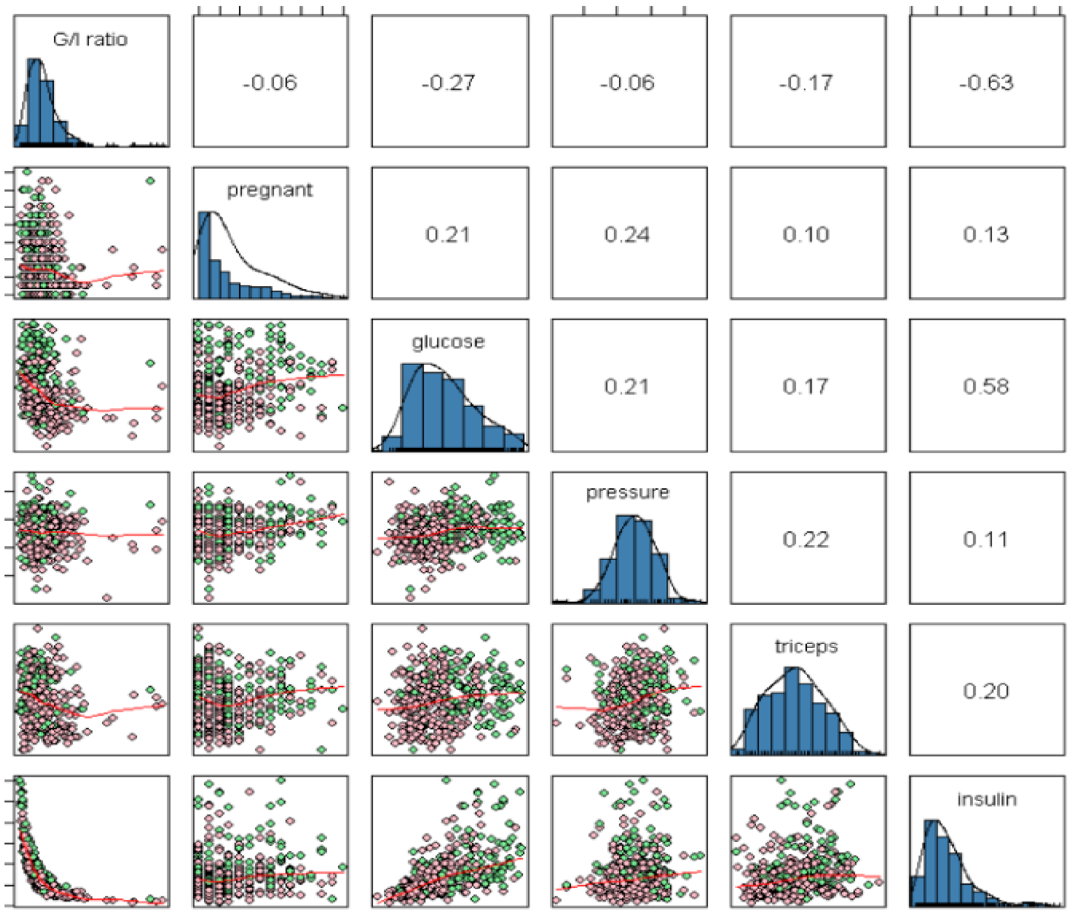
Appendix

pregnant	Number of times pregnant
glucose	Plasma glucose concentration (glucose tolerance test)
pressure	Diastolic blood pressure (mm Hg)
triceps	Triceps skin fold thickness (mm)
insulin	2-Hour serum insulin (mu U/ml)
mass	Body mass index (weight in kg/(height in m)\^2)
pedigree	Diabetes pedigree function
age	Age (years)
diabetes	Class variable (test for diabetes)



RED: Outliers removed

Green: Outliers kept



FUNCTIONS, RCODE FURTHER BELOW

#function for finding best split %

```
x = rep(0,500)
for(i in 1:500){
  set.seed(i)
  training.idx = sample(1:nrow(dia_factored),nrow(dia_factored)*0.4)
  train.data = dia_factored[training.idx,]
  test.data = dia_factored[-training.idx,]
  mlogist <- glm(y~., data=train.data, family="binomial")
  predictions <- predict(mlogist,test.data, type='response')
  y_pred <- factor(ifelse(predictions>0.5,1,0),levels=c(0,1))
  x[i] = mean(y_pred == test.data$y) }
max(x) ; min(x)
max(x)-min(x)
```

#function for geombar

```
for(i in 1:8){
  diageomplot <- dia %>% mutate(y=ifelse(diabetes == 'pos',1,0)) %>% select(-
diabetes) %>% group_by(dia[i]) %>% summarise_at(vars(y), list(probability =mean))
print(ggplot(diageomplot,aes(x=diageomplot[[1]],y=probability,fill=probability))+geom_bar(
stat='identity')+xlab(colnames(diageomplot[1])) +ylab('% of positive diabetes'))
}
```

PS0002 Diabetes

April 18, 2021

1 Data preparation

[18]:
```
library(dplyr);library(ggplot2);library(class);library(lattice)
library(caret);library(psych);library(mlbench)
data("PimaIndiansDiabetes2");str(PimaIndiansDiabetes2);
 ↪summary(PimaIndiansDiabetes2)
PimaIndiansDiabetes2%>% group_by(diabetes) %>% summarise(count=n(),"% of␣
 ↪cases"=round(100*n()/nrow(PimaIndiansDiabetes2),1))
```

```
'data.frame':   768 obs. of  9 variables:
 $ pregnant: num  6 1 8 1 0 5 3 10 2 8 …
 $ glucose : num  148 85 183 89 137 116 78 115 197 125 …
 $ pressure: num  72 66 64 66 40 74 50 NA 70 96 …
 $ triceps : num  35 29 NA 23 35 NA 32 NA 45 NA …
 $ insulin : num  NA NA NA 94 168 NA 88 NA 543 NA …
 $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 NA …
 $ pedigree: num  0.627 0.351 0.672 0.167 2.288 …
 $ age     : num  50 31 32 21 33 30 26 29 53 54 …
 $ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 1 2 2 …

    pregnant         glucose         pressure         triceps
 Min.   : 0.000   Min.   : 44.0   Min.   : 24.00   Min.   : 7.00
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 64.00   1st Qu.:22.00
 Median : 3.000   Median :117.0   Median : 72.00   Median :29.00
 Mean   : 3.845   Mean   :121.7   Mean   : 72.41   Mean   :29.15
 3rd Qu.: 6.000   3rd Qu.:141.0   3rd Qu.: 80.00   3rd Qu.:36.00
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
                  NA's   :5       NA's   :35       NA's   :227
    insulin            mass          pedigree           age         diabetes
 Min.   : 14.00   Min.   :18.20   Min.   :0.0780   Min.   :21.00   neg:500
 1st Qu.: 76.25   1st Qu.:27.50   1st Qu.:0.2437   1st Qu.:24.00   pos:268
 Median :125.00   Median :32.30   Median :0.3725   Median :29.00
 Mean   :155.55   Mean   :32.46   Mean   :0.4719   Mean   :33.24
 3rd Qu.:190.00   3rd Qu.:36.60   3rd Qu.:0.6262   3rd Qu.:41.00
 Max.   :846.00   Max.   :67.10   Max.   :2.4200   Max.   :81.00
 NA's   :374      NA's   :11
```

diabetes	count	% of cases
neg	500	65.1
pos	268	34.9

```
[19]: dia <- PimaIndiansDiabetes2[complete.cases(PimaIndiansDiabetes2),]
      rownames(dia) <- 1:nrow(dia)
      dia <- dia %>% mutate(GIratio = glucose/insulin)
      dia <- dia[,c(ncol(dia),1:(ncol(dia)-1))]
      nrow(dia)
      diaplot <- dia%>% group_by(diabetes) #to boxplot
      summarise(diaplot,count=n(),"% of cases"=round(100*n()/nrow(diaplot),1))
      summary(dia)
```

392

diabetes	count	% of cases
neg	262	66.8
pos	130	33.2

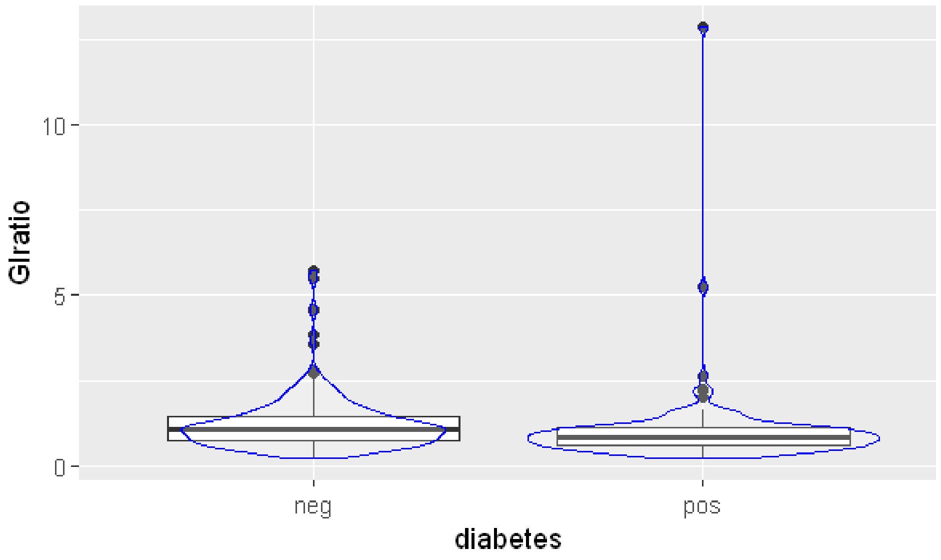
```
    GIratio            pregnant          glucose          pressure     
 Min.   : 0.2067   Min.   : 0.000   Min.   : 56.0   Min.   : 24.00  
 1st Qu.: 0.6698   1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00  
 Median : 0.9872   Median : 2.000   Median :119.0   Median : 70.00  
 Mean   : 1.1555   Mean   : 3.301   Mean   :122.6   Mean   : 70.66  
 3rd Qu.: 1.3333   3rd Qu.: 5.000   3rd Qu.:143.0   3rd Qu.: 78.00  
 Max.   :12.8571   Max.   :17.000   Max.   :198.0   Max.   :110.00  
    triceps          insulin              mass          pedigree     
 Min.   : 7.00   Min.   : 14.00   Min.   :18.20   Min.   :0.0850  
 1st Qu.:21.00   1st Qu.: 76.75   1st Qu.:28.40   1st Qu.:0.2697  
 Median :29.00   Median :125.50   Median :33.20   Median :0.4495  
 Mean   :29.15   Mean   :156.06   Mean   :33.09   Mean   :0.5230  
 3rd Qu.:37.00   3rd Qu.:190.00   3rd Qu.:37.10   3rd Qu.:0.6870  
 Max.   :63.00   Max.   :846.00   Max.   :67.10   Max.   :2.4200  
      age         diabetes 
 Min.   :21.00   neg:262  
 1st Qu.:23.00   pos:130  
 Median :27.00            
 Mean   :30.86            
 3rd Qu.:36.00            
 Max.   :81.00            
```

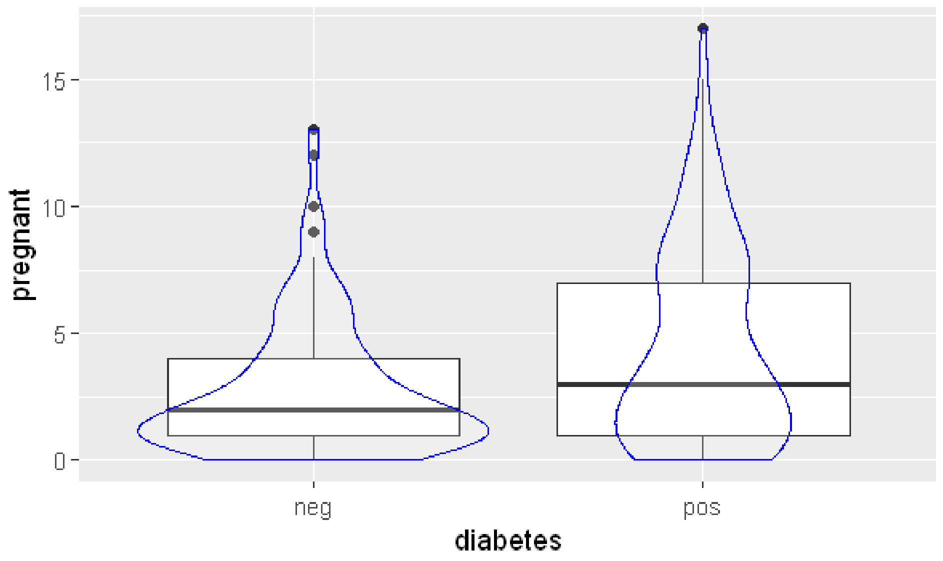
2 EDA

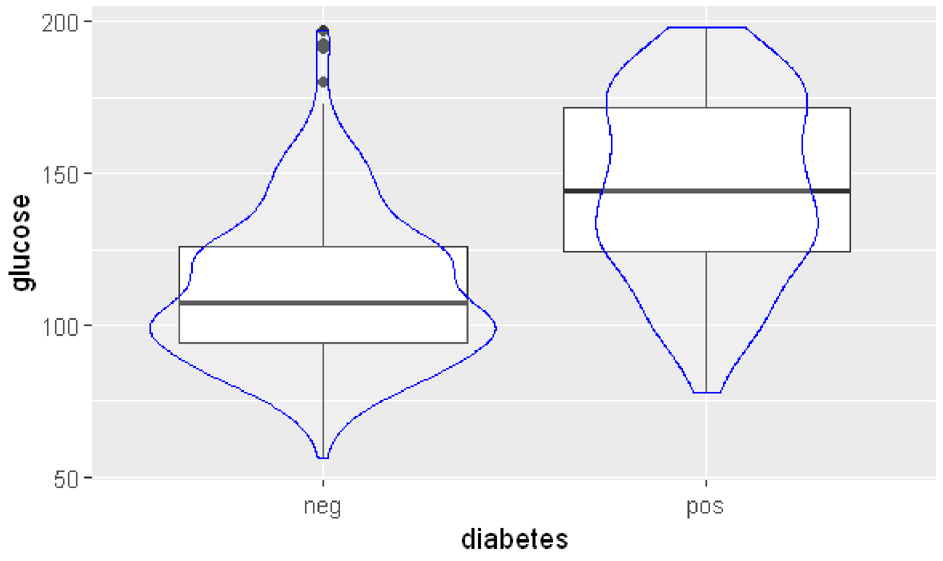
2.1 r/s btw variables and diabetes class (box, histo)

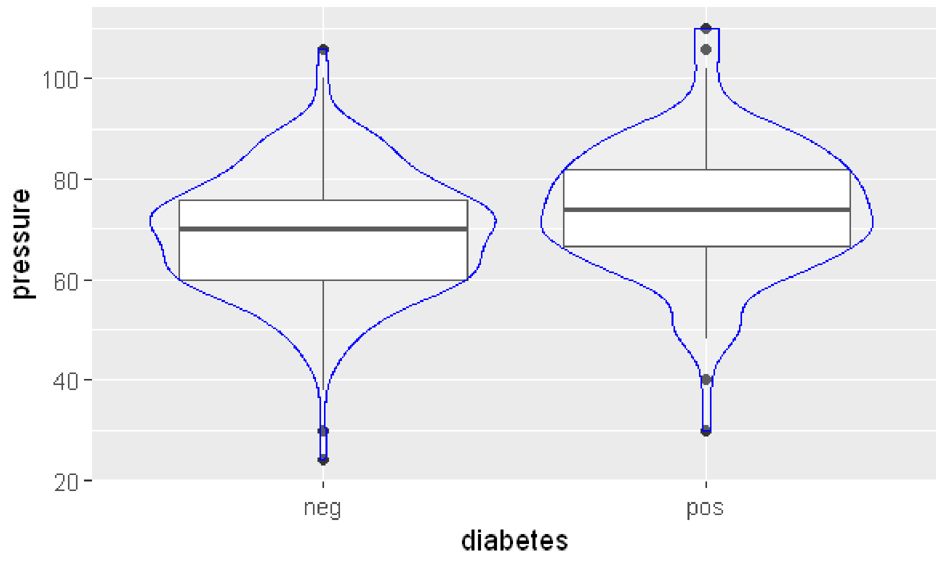
```
[20]: library(repr); options(repr.plot.width=5, repr.plot.height=3)
      diaplot <- dia%>% group_by(diabetes) #to boxplot
      for(i in 1:(ncol(diaplot)-1)){
```

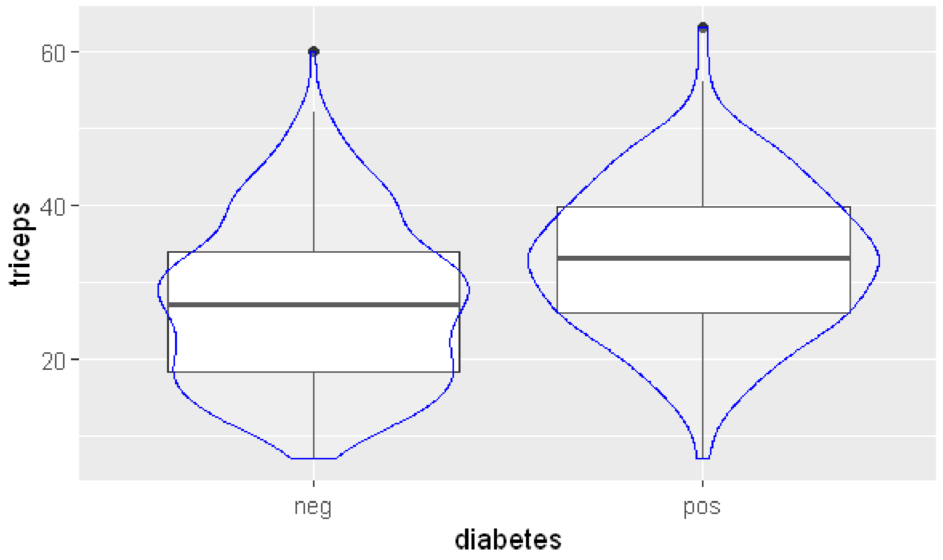
```
  print(ggplot(diaplot,aes(x=diabetes, y=diaplot[[i]]))+geom_boxplot()␣
↪+geom_violin(alpha=0.2,color="blue") +ylab(colnames(diaplot[i])))
  #print(ggplot(diaplot, aes(x=diaplot[[i]],␣
↪fill=diabetes))+geom_histogram(bins=30)+xlab(colnames(diaplot[i])))
}

#for(i in 1:8){
#  diageomplot <- dia %>% mutate(y=ifelse(diabetes == 'pos',1,0)) %>%␣
↪select(-diabetes) %>% group_by(dia[i]) %>% summarise_at(vars(y),␣
↪list(probability =mean))
# ␣
↪print(ggplot(diageomplot,aes(x=diageomplot[[1]],y=probability,fill=probability))+geom_bar(s
↪+ylab('% of positive diabetes'))
#}
```

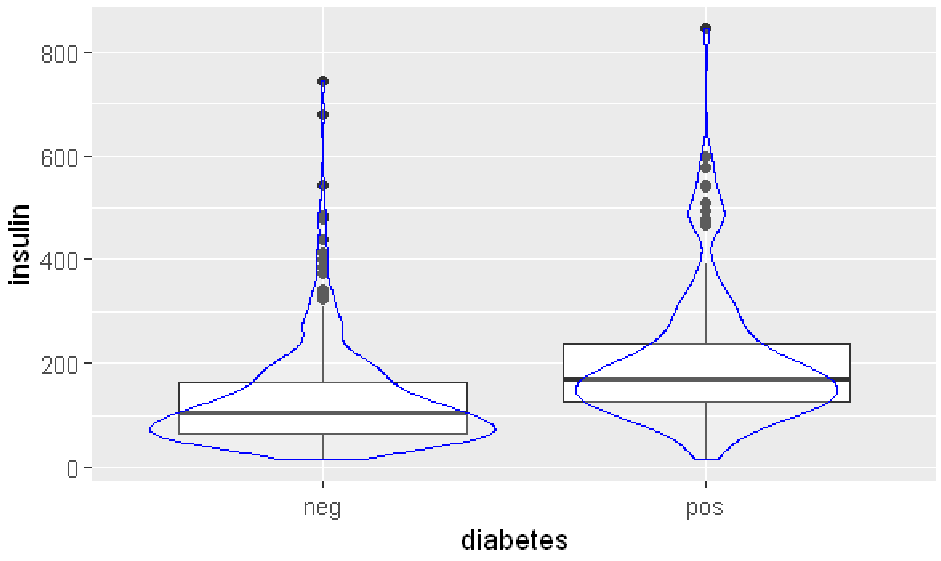


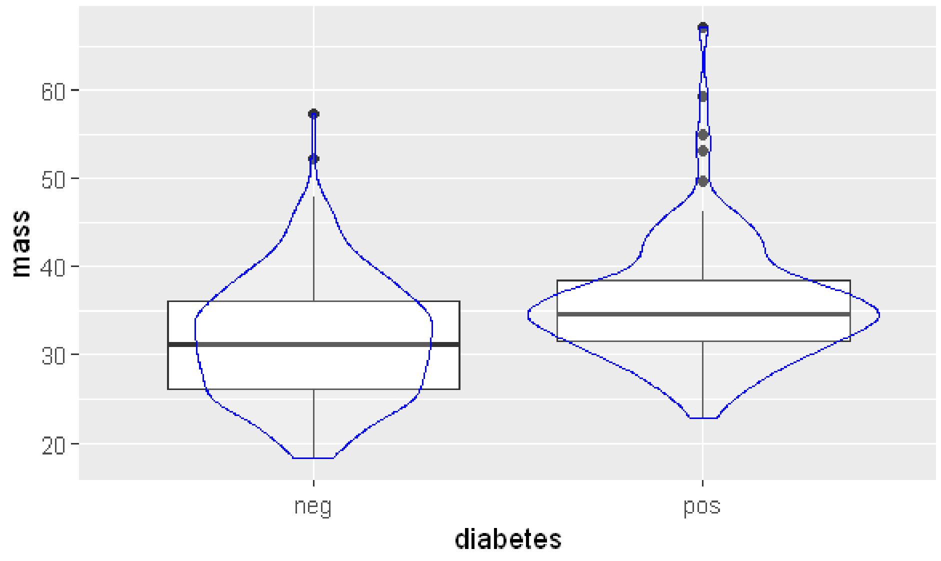

15
10
5
0
pregnant
neg
pos
diabetes

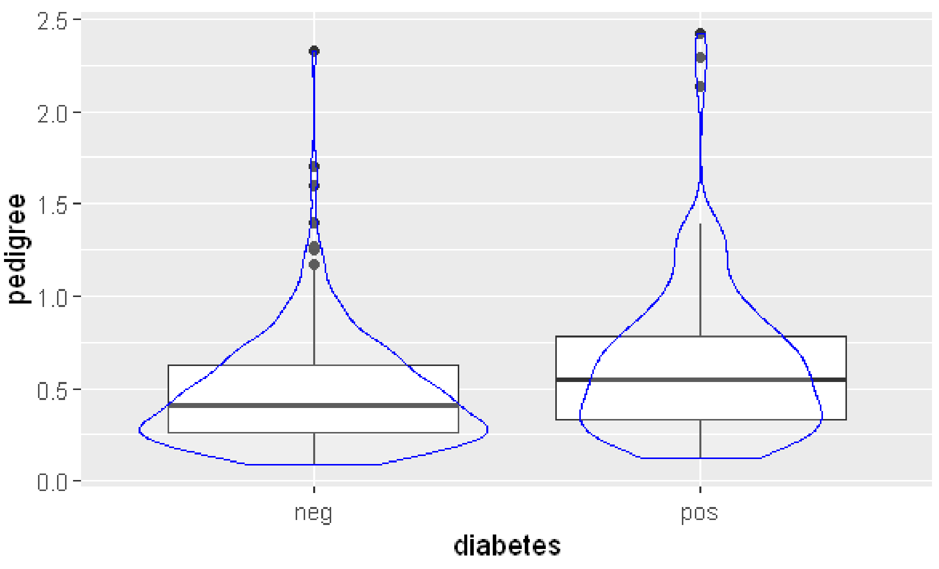

200
150
100
50
glucose
neg
pos
diabetes

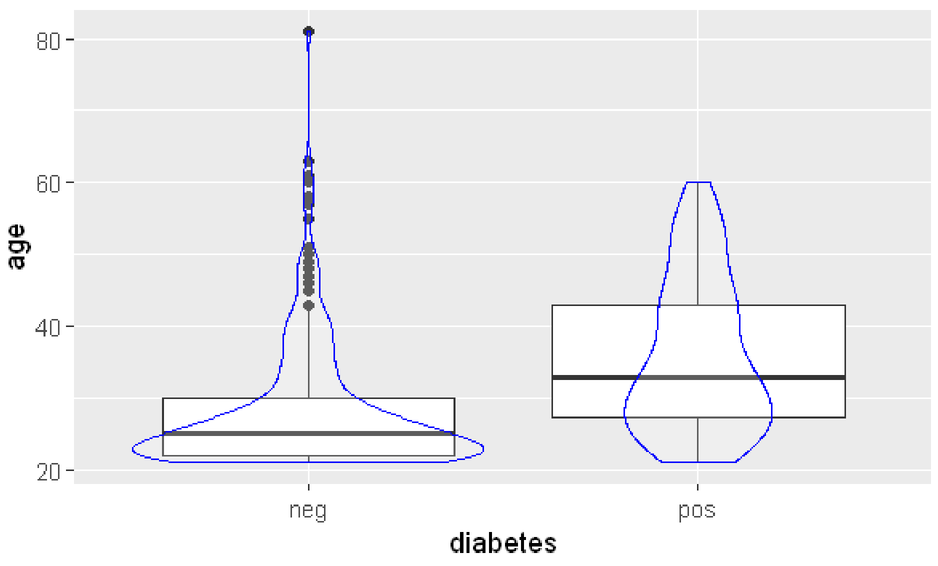

pressure
100
80
60
40
20
neg
pos
diabetes


triceps
60
40
20
neg
pos
diabetes


insulin
800
600
400
200
0
neg
pos
diabetes

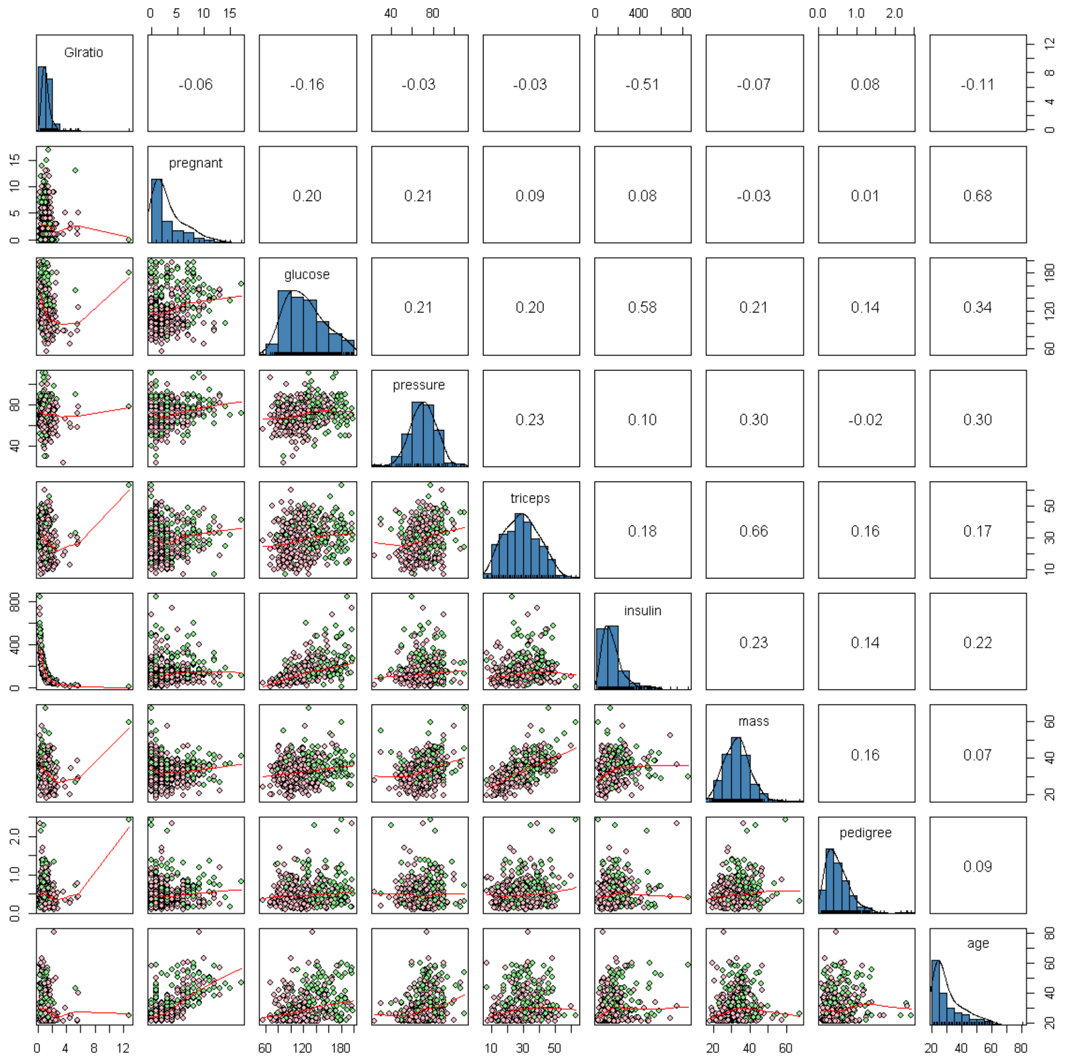

mass
60
50
40
30
20
neg
pos
diabetes





2.1.1 Finding natural clusters and relation between variables

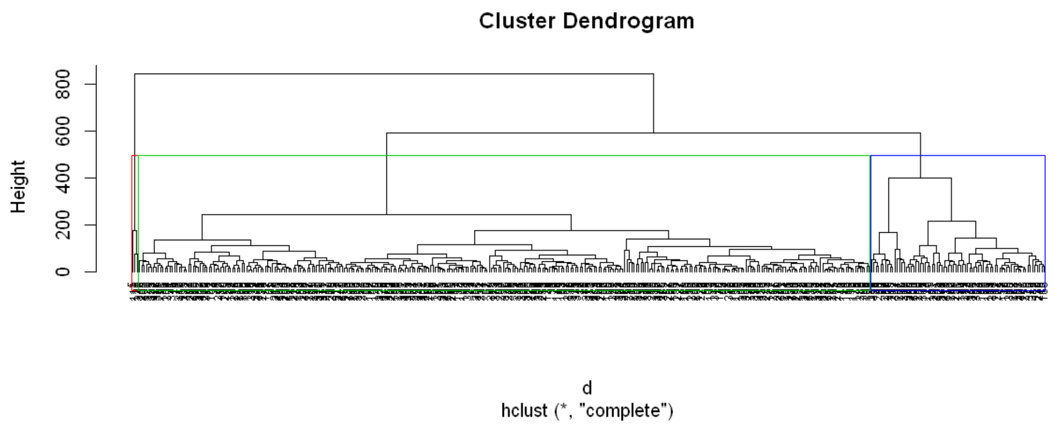
[21]:
```
options(repr.plot.width=10, repr.plot.height=10)
pairs.panels(dia[,-ncol(diaplot)], method = "pearson", hist.col = "steelblue",␣
↪pch = 21, bg = c("pink", "light green")[(dia$diabetes)],
             density = TRUE, ellipses = FALSE)
```



natural cluster in glucose vs insulin, mass vs triceps, pregnant vs age

2.1.2 Hierarchical Clustering

[22]:
```
options(repr.plot.width=10, repr.plot.height=4)
dia_numeric <- select(dia,-diabetes)
d <- dist(dia_numeric, method="euclidean")
hc <- hclust(d, method = 'complete')
plot(hc, cex = 0.6, hang = -1)
rect.hclust(hc, k = 3, border = 2:4)
```



suspect that first cluster formed by outliers

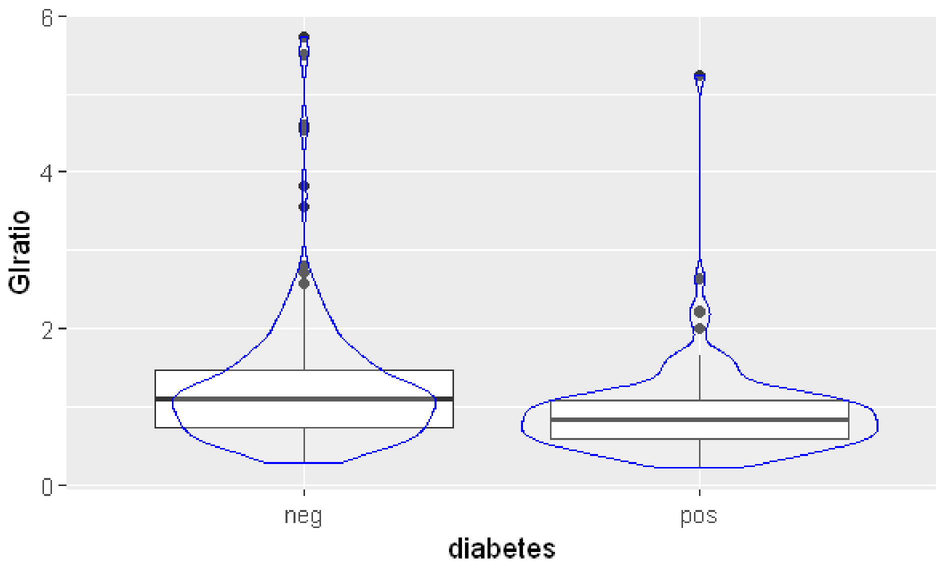
2.1.3 Remove extreme statistical outliers

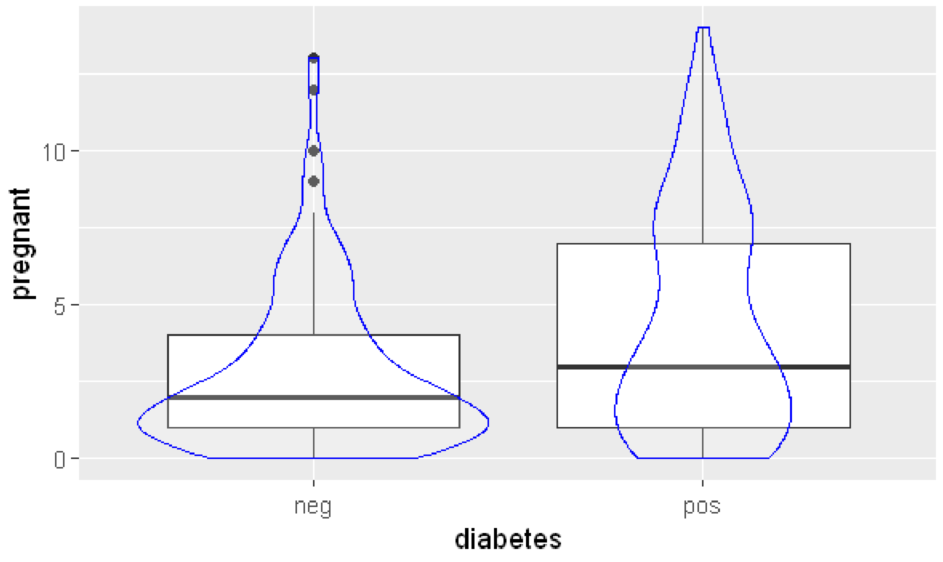
[23]:
```
dia_outlier_removed <- dia
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",dia_outlier_removed[1]<6,dia_outlier_removed[1]<6))
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",pregnant<15,pregnant<15))
#dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",glucose<170,glucose<200))
#dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",pressure>27,pressure>35))
#dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",triceps<59,triceps<60))
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",insulin<640,insulin<640))
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",mass<65,mass<65))
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",pedigree<2,pedigree<2))
dia_outlier_removed <- dia_outlier_removed %>% filter(ifelse(diabetes ==␣
↪"neg",age<75,age<75))
nrow(dia_outlier_removed)
```

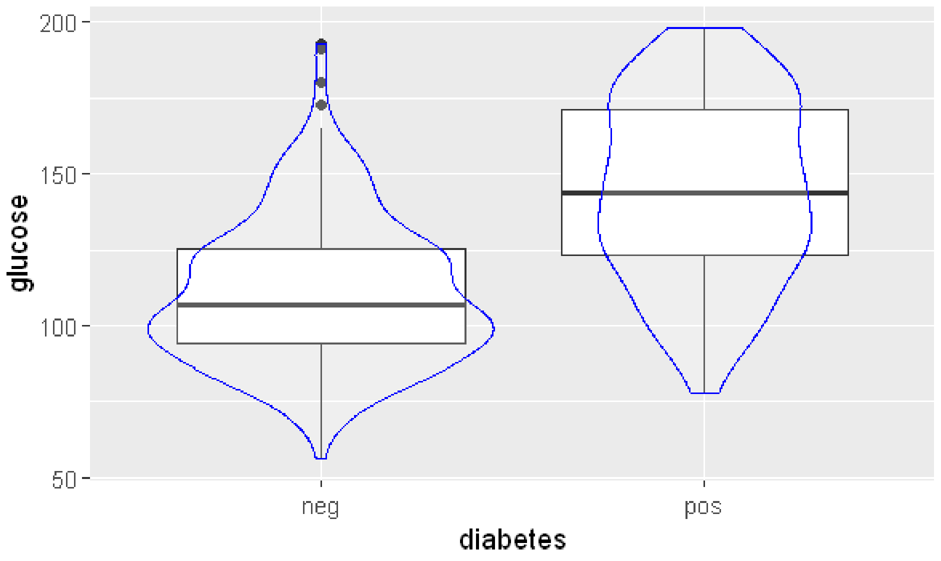
382

2.1.4 Replot

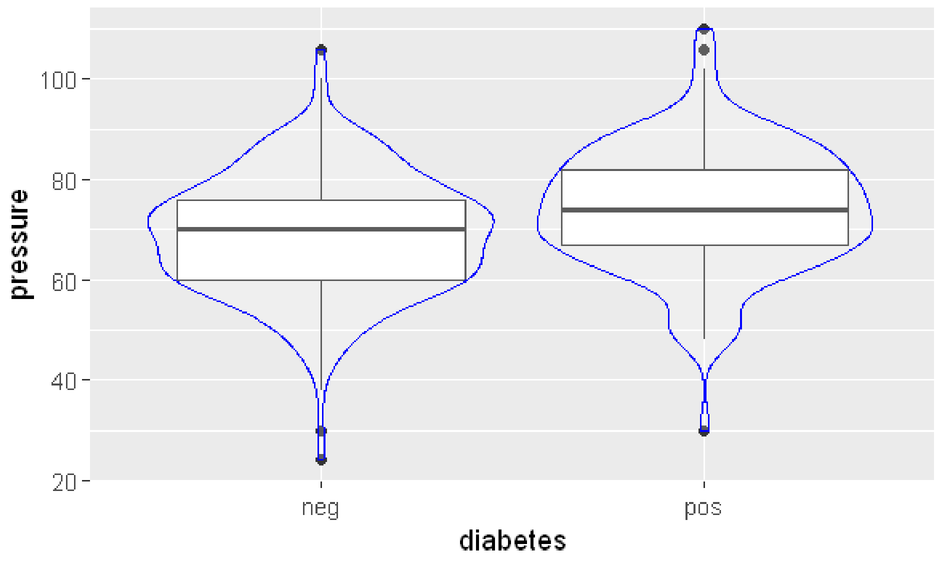
```
[24]: options(repr.plot.width=5, repr.plot.height=3)
      diaplot <- dia_outlier_removed%>% group_by(diabetes) #to boxplot
      for(i in 1:(ncol(diaplot)-1)){
        print(ggplot(diaplot,aes(x=diabetes, y=diaplot[[i]]))+geom_boxplot()␣
       ↪+geom_violin(alpha=0.2,color="blue") +ylab(colnames(diaplot[i])))
        #print(ggplot(diaplot, aes(x=diaplot[[i]],␣
       ↪fill=diabetes))+geom_histogram(bins=30)+xlab(colnames(diaplot[i])))
      }
```



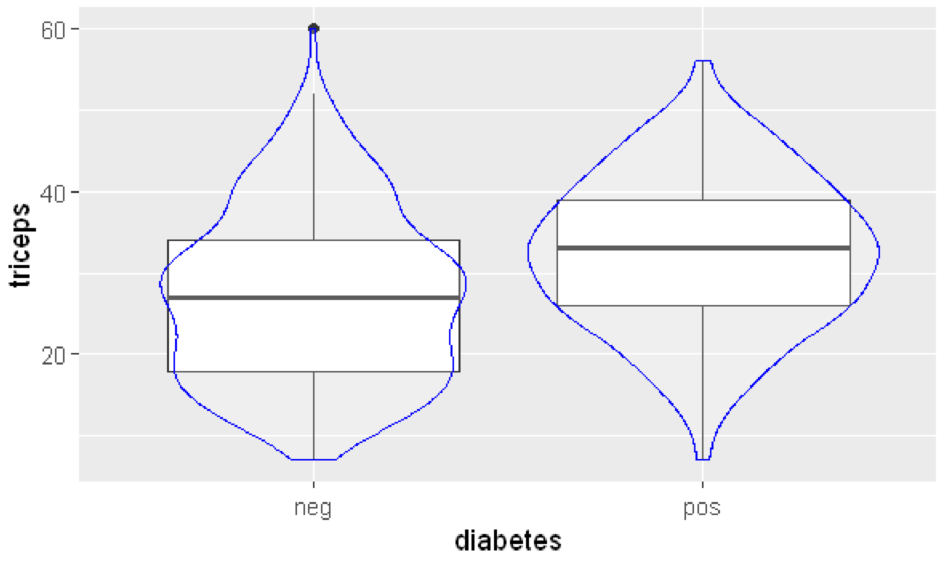


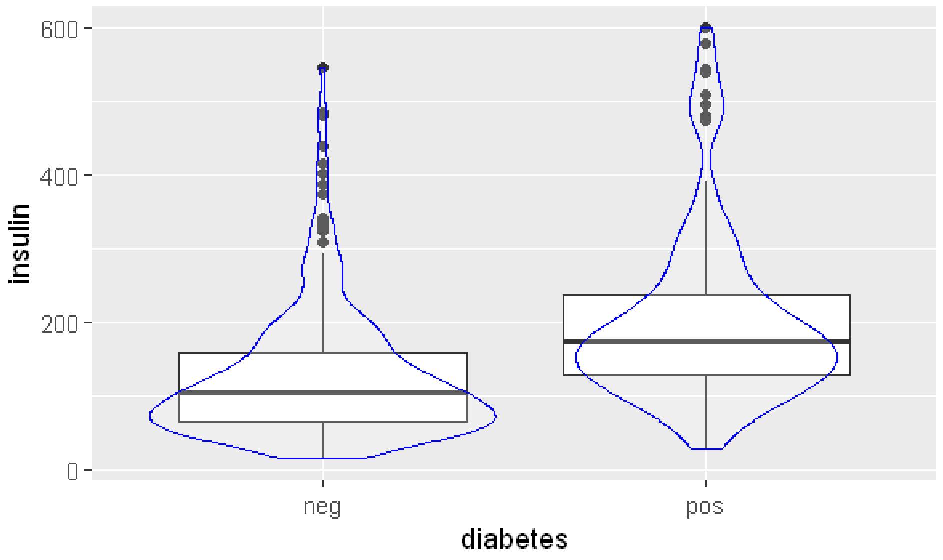


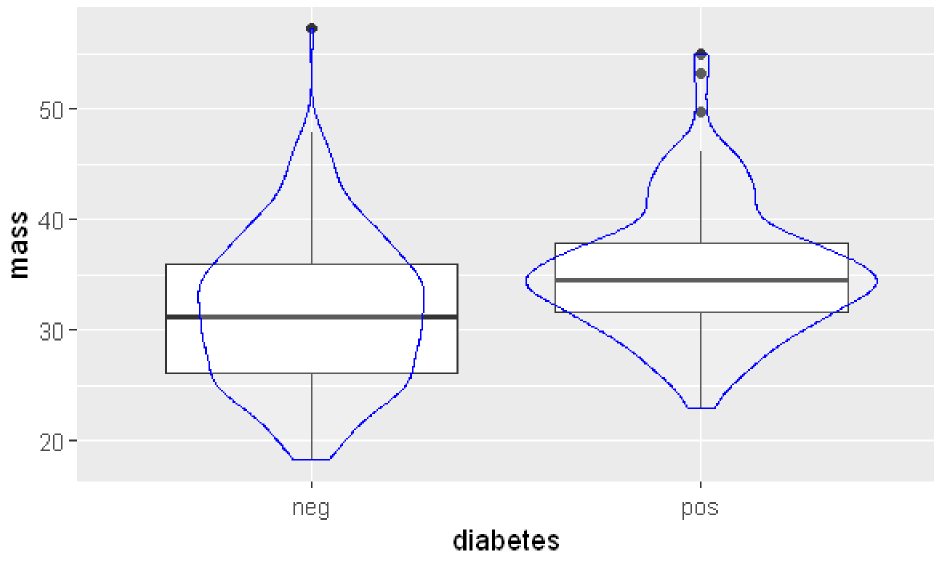
200
150
100
50
glucose
neg
pos
diabetes

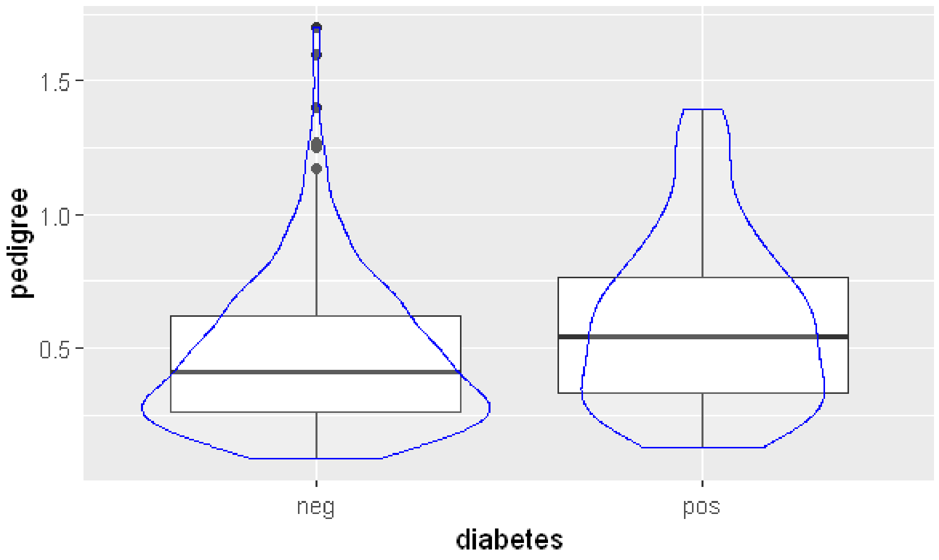


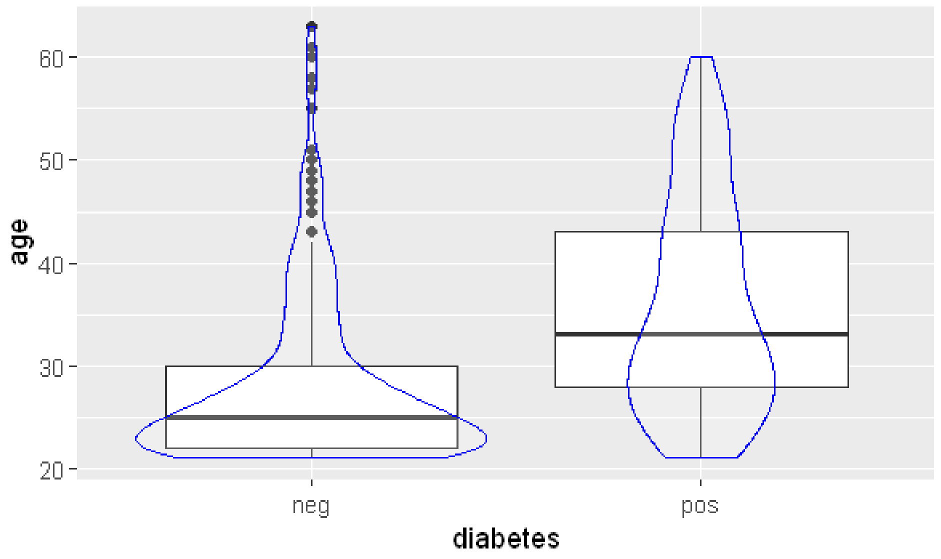
100
80
60
40
20
pressure
neg
pos
diabetes


60
40
20
triceps
neg
pos
diabetes

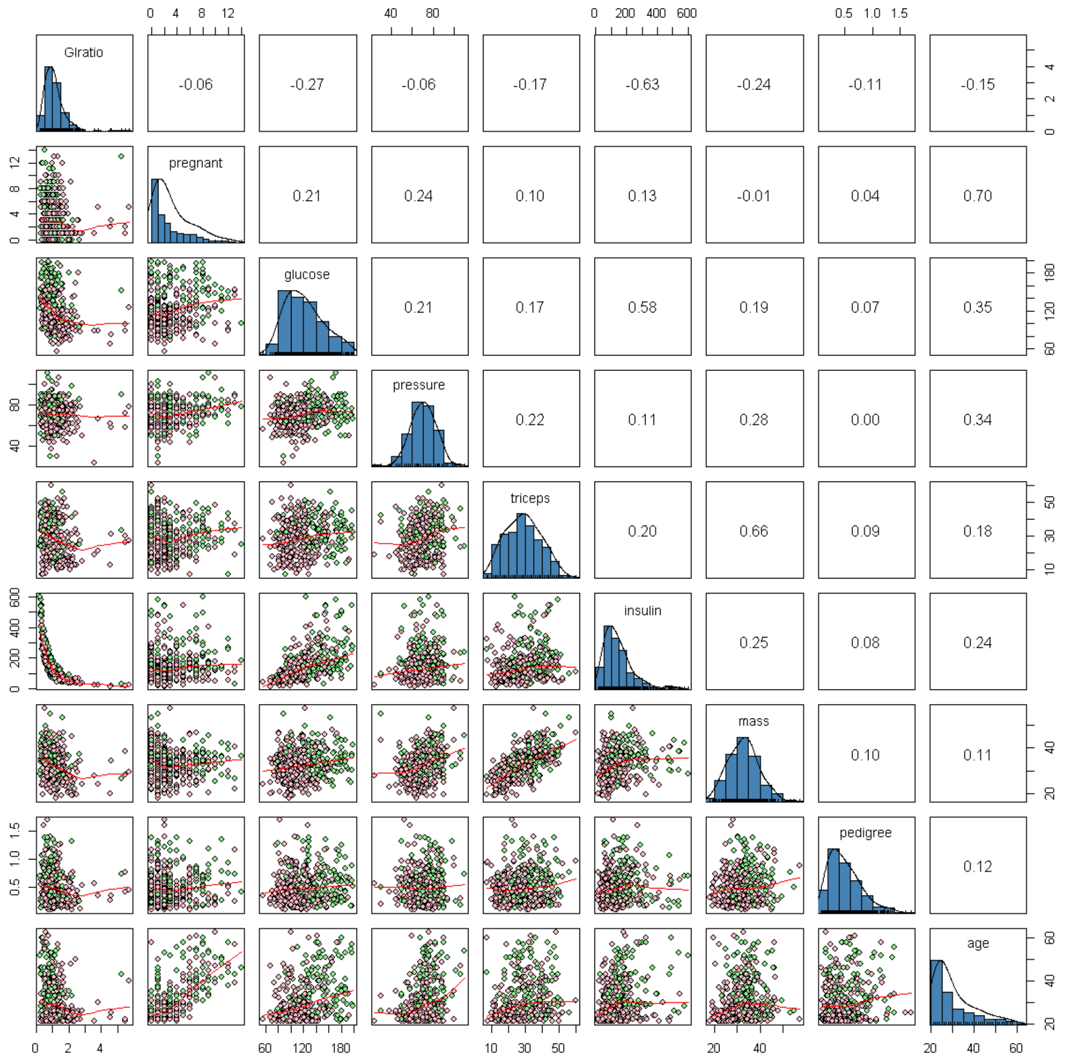

600
400
200
0
insulin
neg
pos
diabetes


mass
50
40
30
20
neg
pos
diabetes


pedigree
1.5
1.0
0.5
neg
pos
diabetes

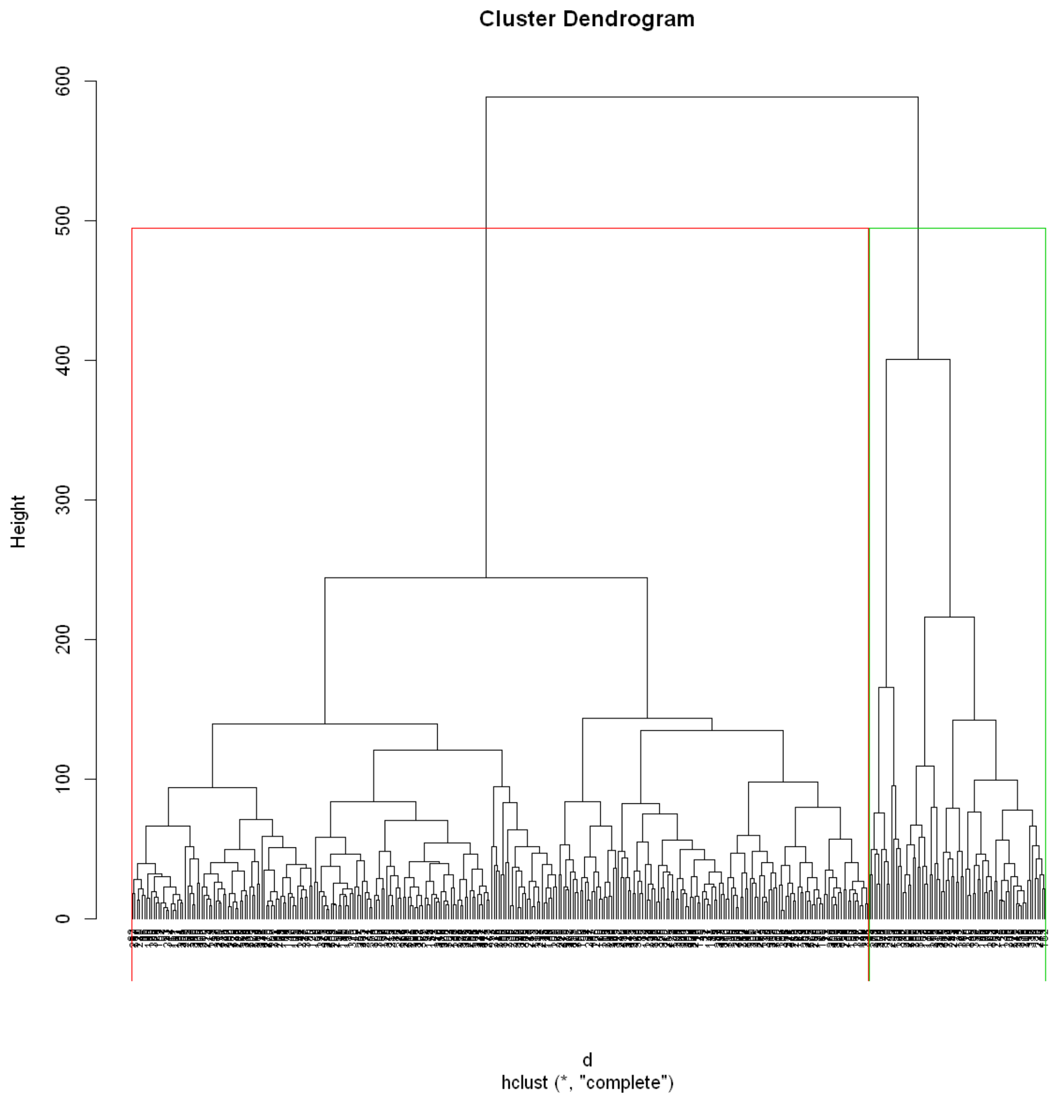


```
[25]: options(repr.plot.width=10, repr.plot.height=10)
pairs.panels(diaplot[,-ncol(diaplot)], method = "pearson", hist.col =␣
 ↪"steelblue", pch = 21, bg = c("pink", "light green")[(diaplot$diabetes)],
              density = TRUE, ellipses = FALSE)
```



2.1.5 Recluster after remove outliers

[26]:
```r
dia_numeric <- select(dia_outlier_removed,-diabetes)
d <- dist(dia_numeric, method="euclidean")
hc <- hclust(d, method = 'complete')
plot(hc, cex = 0.6, hang = -1)
rect.hclust(hc, k = 2, border = 2:3)
```



```
[27]: sub_grp <- cutree(hc, k = 2)
      diacluster <- dia_outlier_removed %>%
        mutate(clust = sub_grp) %>% group_by(clust)
      count(diacluster, diabetes_positive = diabetes == "pos")
      summarise(diacluster, count=n())
```

clust	diabetes_positive	n
1	FALSE	222
1	TRUE	86
2	FALSE	37
2	TRUE	37

clust	count
1	308
2	74

[28]:
```
# clust1 - 86/308= 28% pos
# clust2 - 37/74= 50% pos
```

3 Classification

3.1 Logistic regression

3.1.1 Train/Test prep and split

[149]:
```
dia_factored <- dia_outlier_removed %>% mutate(y=factor(ifelse(diabetes ==␣
↪'pos',1,0))) %>% select(-diabetes,-pressure)

set.seed(100)
training.idx = sample(1:nrow(dia_factored),nrow(dia_factored)*0.6)
train.data = dia_factored[training.idx,]
test.data = dia_factored[-training.idx,]
train.data %>% group_by(y) %>% summarise(count=n(),"% of cases"=round(100*n()/
↪nrow(train.data),1))
test.data %>% group_by(y) %>% summarise(count=n(),"% of cases"=round(100*n()/
↪nrow(test.data),1))
```

y	count	% of cases
0	157	68.6
1	72	31.4

y	count	% of cases
0	102	66.7
1	51	33.3

train and test have similar class imbalance as each other and with distrib of whole dataset

3.1.2 Logistic regression model

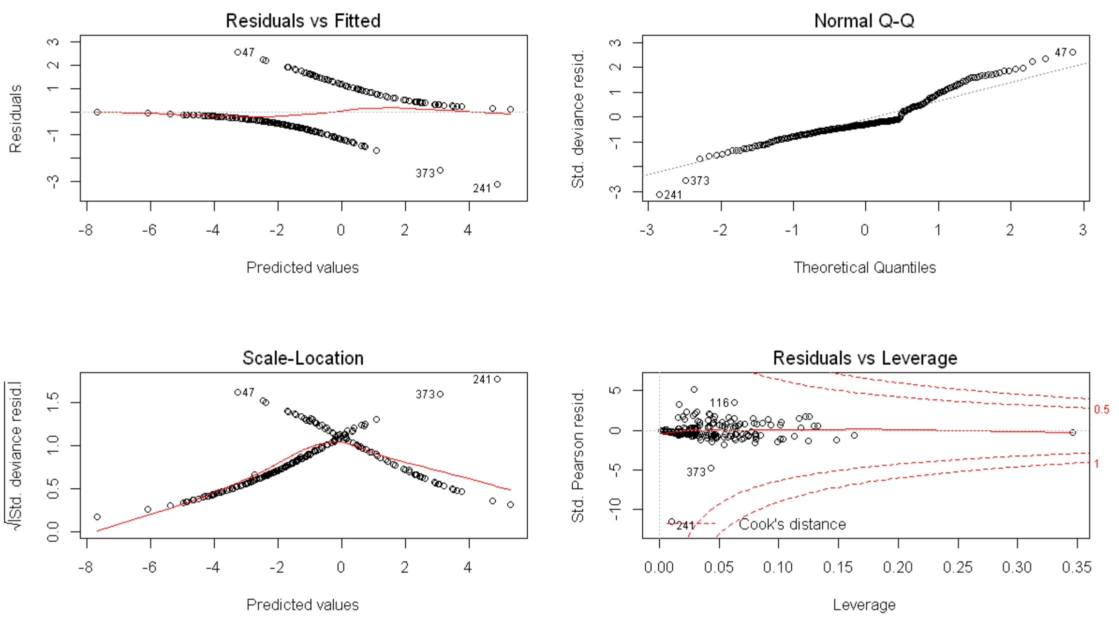
[144]:
```
#pregnant+pressure+triceps+insulin+mass+pedigree+glucose+age 0.77base
#exp(pregnant)
mlogist <- glm(y~pregnant+GIratio+triceps+insulin+mass+pedigree+glucose+age,␣
↪data=train.data, family="binomial")
predictions <- predict(mlogist,test.data, type='response')
y_pred <- factor(ifelse(predictions>0.5,1,0),levels=c(0,1))
mean(y_pred == test.data$y) #0.77778
table(y_pred,test.data$y)
```

0.751633986928105

```
y_pred  0  1
```

```
0  86  22
1  16  29
```

[145]:
```
## plot logistic reg model
par(mfrow=c(2,2)); options(repr.plot.width=10, repr.plot.height=6)
plot(mlogist) #outlier 3
par(mfrow=c(1,1))
```



3.1.3 Improve log reg by removing outlier

[150]:
```
dia_factored[c(47,241),]
241 %in% training.idx
train_improved.data = train.data[-c(47,241),]

mlogist <- glm(y~., data=train_improved.data, family="binomial")
predictions <- predict(mlogist,test.data, type='response')
y_pred <- factor(ifelse(predictions>0.5,1,0),levels=c(0,1))
mean(y_pred == test.data$y)
table(y_pred,test.data$y)
```

	GIratio	pregnant	glucose	triceps	insulin	mass	pedigree	age	y
47	2.6388889	0	95	25	36	37.4	0.247	24	1
241	0.6528302	0	173	32	265	46.5	1.159	58	0

TRUE

0.758169934640523

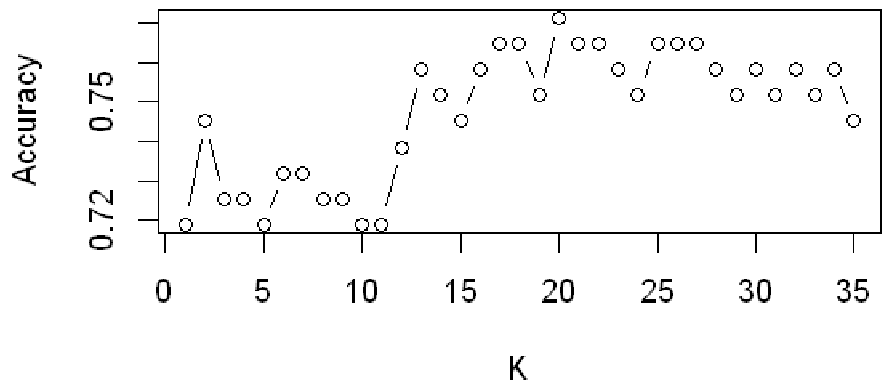
```
y_pred  0  1
     0 85 20
     1 17 31
```

not visible improvement

3.2 kNN classify

```
[155]: nor <-function(x) {(x -min(x))/(max(x)-min(x))}
       #dia_nor <- dia_factored
       #dia_nor[,1:8] <- sapply(dia_nor[,1:8],nor)
       train_knn.data <- train.data
       test_knn.data <- test.data
       train_knn.data[,1:8] <- sapply(train_knn.data[,1:8],nor)
       test_knn.data[,1:8] <- sapply(test_knn.data[,1:8],nor)
       #set.seed(100)
       #training.idx = sample(1:nrow(dia_nor),nrow(dia_nor)*0.8)
       #train_knn.data = dia_nor[training.idx,]
       #test_knn.data = dia_nor[-training.idx,]
       head(train_knn.data)
       x = rep(0,35)
       for(i in 1:35){
         set.seed(100)
         knn.i = knn(train_knn.data[,1:8],test_knn.data[,1:8],train_knn.data$y,k=i)
         x[i] = mean(knn.i==test_knn.data$y)}
         #cat("k=", i, " accuracy=", x[i], "\n")
       options(repr.plot.width=5, repr.plot.height=3); plot(x, type="b",␣
        ↪xlab="K",ylab="Accuracy")
       set.seed(100)
       knn2 = knn(train_knn.data[,1:8],test_knn.data[,1:8],train_knn.data$y,k=20)
```

	GIratio	pregnant	glucose	triceps	insulin	mass	pedigree	age	y
202	0.37845089	0.07142857	0.6126761	0.2830189	0.08156028	0.2173913	0.1128713	0.0000000	0
358	0.03539257	0.50000000	0.9225352	0.4905660	0.66843972	0.4266304	0.4891089	0.3333333	1
112	0.07197173	0.42857143	0.4436620	0.2830189	0.28546099	0.2418478	0.8138614	0.3076923	1
206	0.07635611	0.07142857	0.4436620	0.6415094	0.27482270	0.7364130	0.2785479	0.1282051	0
4	0.12202531	0.35714286	0.7746479	0.2264151	0.28368794	0.2065217	0.3313531	0.7692308	1
311	0.09792402	0.00000000	0.2676056	0.3773585	0.17730496	0.6875000	0.1729373	0.0000000	0



[147]:
```
mean(knn2==test_knn.data$y) #0.7
table(knn2, test_knn.data$y)
```

```
0.77124183006536
```

```
knn2  0  1
   0 94 27
   1  8 24
```

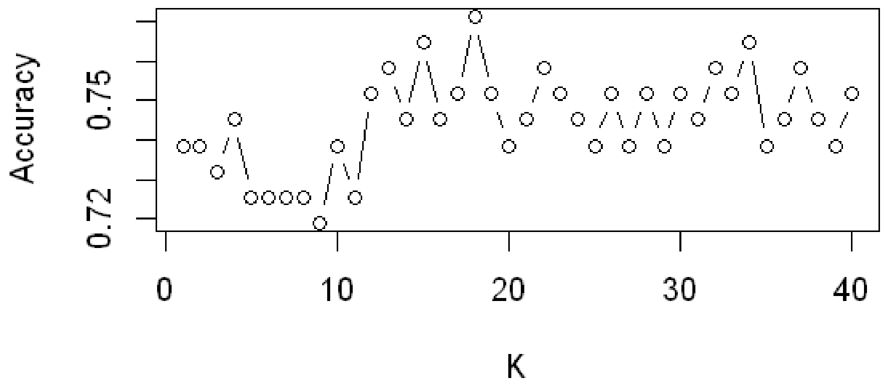
3.2.1 Improve kNN classi by weighting predictors

Weighting predictors

[153]:
```
train_knn_stretch.data <- train_knn.data
test_knn_stretch.data <- test_knn.data

#diamean <- dia_factored %>% group_by(y) %>% summarise_all("mean")
#scaling the normalized data base on significance found by mean
# for(i in 1:8){
#     y <- (diamean[2,i+1] - diamean[1,i+1])/diamean[1,i+1]
#     dia_nor[i] <- dia_nor[i] * (y[1,1])}

#scale gluc5.9, pedi4.1
train_knn_stretch.data$glucose <- train_knn_stretch.data$glucose * (1.59)
test_knn_stretch.data$glucose <- test_knn_stretch.data$glucose * (1.59)
head(train_knn_stretch.data,4)
#dia_nor[7] <- dia_nor[7] * (1.41)
```

	GIratio	pregnant	glucose	triceps	insulin	mass	pedigree	age	y
202	0.37845089	0.07142857	0.9741549	0.2830189	0.08156028	0.2173913	0.1128713	0.0000000	0
358	0.03539257	0.50000000	1.4668310	0.4905660	0.66843972	0.4266304	0.4891089	0.3333333	1
112	0.07197173	0.42857143	0.7054225	0.2830189	0.28546099	0.2418478	0.8138614	0.3076923	1
206	0.07635611	0.07142857	0.7054225	0.6415094	0.27482270	0.7364130	0.2785479	0.1282051	0

[157]:
```
x = rep(0,40)
for(i in 1:40){
  set.seed(100)
  knn.i = knn(train_knn_stretch.data[,1:8],test_knn_stretch.data[,1:
↪8],train_knn_stretch.data$y,k=i)
  x[i] = mean(knn.i==test_knn_stretch.data$y)}
  #cat("k=", i, " accuracy=", x[i], "\n")}
plot(x, type="b", xlab="K",ylab="Accuracy")  #the best k
set.seed(100)
knn2 = knn(train_knn_stretch.data[,1:8],test_knn_stretch.data[,1:
↪8],train_knn_stretch.data$y,k=18)
```



[158]:
```
mean(knn2==test_knn_stretch.data$y) #0.78947
table(knn2, test_knn_stretch.data$y)
```

0.77124183006536

```
knn2  0  1
   0 91 24
   1 11 27
```

improvement??

3.3 SVM classify

[148]:
```
train_svm.data <- train.data
test_svm.data <- test.data

library(e1071)
m.svm <- svm(y~.,data=train_svm.data, kernel='linear')
predictions <- predict(m.svm,test_svm.data)
mean(predictions == test_svm.data$y)
table(predictions, test_svm.data$y)
#exclude radial
```

0.758169934640523

```
predictions  0  1
          0 87 22
          1 15 29
```

[159]:
```
set.seed(100)
m.svm <- svm(y~., data=train_svm.data, kernel='polynomial',degree=2,␣
 ↪cost=10^(-3), gamma=4, coef0=5)
predictions <- predict(m.svm,test_svm.data)
mean(predictions == test_svm.data$y)
table(predictions,test_svm.data$y)
```

0.758169934640523

```
predictions  0  1
          0 90 25
          1 12 26
```

4 Impute insulin NA values to improve classif

[ ]:
```
#From data exploration, glucose mass age has correlation of more than 0.2
dia_regression <- dia %>% select(glucose,mass,age,insulin)
pairs.panels(dia_regression[,], method = "pearson", hist.col = "steelblue", pch␣
 ↪= 21, bg = c("pink", "light green")[(dia$diabetes)],
             density = TRUE, ellipses = FALSE)
```

4.0.1 kNN regression

[ ]:
```
set.seed(100)
training.idx <- sample(1:nrow(dia_regression),nrow(dia_regression)*0.8)
train_regression.data <- dia_regression[training.idx,]
test_regression.data <- dia_regression[-training.idx,]
dim(train_regression.data)
```

```
set.seed(100); options(repr.plot.width=5, repr.plot.height=4)
#training model using KNN model, predicting insulin using all other variables,␣
 ↪1 cv = 50 data row
knn_model <- train(insulin~., data = train_regression.data, method = "knn",␣
 ↪trControl = trainControl("cv", number = 7),
               preProcess = c("center","scale"), tuneLength = 10) #how to find␣
 ↪best tuneLength?
#plot(knn_model)
knn_model$bestTune
predictions <- predict(knn_model,test_regression.data)
RMSE(predictions,test_regression.data$insulin) #75.39407
plot(test_regression.data$insulin, predictions,main="Prediction performance of␣
 ↪kNN regression")
abline(0,1, col="red") #reference line x = y
```

4.1 Logistic reg classify BEFORE imputing, without triceps

```
[ ]: dia_factored <- PimaIndiansDiabetes2 %>% select(-triceps)
     dia_factored <- dia_factored[complete.cases(dia_factored),]
     dia_factored <- dia_factored %>% mutate(y=factor(ifelse(diabetes ==␣
      ↪'pos',1,0))) %>% select(-diabetes)
     set.seed(100)
     trainingbef.idx = sample(1:nrow(dia_factored),nrow(dia_factored)*0.6)
     trainbef.data = dia_factored[trainingbef.idx,]
     testbef.data = dia_factored[-trainingbef.idx,]
     mlogist <- glm(y~., data=trainbef.data, family="binomial")
     predictions <- predict(mlogist,testbef.data, type='response')
     y_pred <- factor(ifelse(predictions>0.5,1,0),levels=c(0,1))
     mean(y_pred == testbef.data$y) #0.7974684
     table(y_pred,testbef.data$y)
```

4.2 Imputing insulin

```
[ ]: dia_impute <- PimaIndiansDiabetes2[is.na(PimaIndiansDiabetes2[,"insulin"]),]
     dia_impute <- dia_impute %>% select(insulin,mass,glucose,age)
     dia_impute <- dia_impute[!is.na(dia_impute[,"mass"]),]
     dia_impute <- dia_impute[!is.na(dia_impute[,"glucose"]),]
     imputevalue <- predict(knn_model,dia_impute)
     dia_impute$insulin <- imputevalue
     dia_r1 <- PimaIndiansDiabetes2[is.na(PimaIndiansDiabetes2[,"insulin"]),]
     dia_r1 <- dia_r1[!is.na(dia_r1[,"mass"]),]
     dia_r1 <- dia_r1[!is.na(dia_r1[,"glucose"]),]
     dia_r1$insulin <- imputevalue
     dia_r1 <- dia_r1 %>% select(-triceps)
     dia_r1 <- dia_r1[complete.cases(dia_r1),] #332 of 8
```

```
dia_r1_factored <- dia_r1 %>% mutate(y=factor(ifelse(diabetes == 'pos',1,0)))␣
 ↪%>% select(-diabetes)
```

4.3 Logistic reg classify AFTER imputing

```
[ ]: train.data = rbind(trainbef.data,dia_r1_factored)
     rownames(train.data) <- 1:nrow(train.data)
     set.seed(100)
     m.svm <- svm(y~pressure+pedigree+age+glucose, data=train.data,␣
      ↪kernel='polynomial',degree=2, cost=10^(-1), gamma=1, coef0=9)
     y_pred <- predict(m.svm,testbef.data)
     mean(y_pred == testbef.data$y)
     table(y_pred,testbef.data$y)
```

```
[ ]: train.data = rbind(trainbef.data,dia_r1_factored)
     rownames(train.data) <- 1:nrow(train.data)
     mlogist_imputed <- glm(y~., data=train.data, family="binomial")
     predictions <- predict(mlogist_imputed,testbef.data, type='response')
     y_pred <- factor(ifelse(predictions>0.5,1,0),levels=c(0,1))
     mean(y_pred == testbef.data$y) #0.7848101
     table(y_pred,testbef.data$y)
```

```
[ ]: predictions_nonimputed <- predict(mlogist,testbef.data, type='response')
     y_pred_nonimputed <- factor(ifelse(predictions_nonimputed>0.
      ↪5,1,0),levels=c(0,1))
     mean(y_pred_nonimputed == testbef.data$y) #0.7974684
     table(y_pred_nonimputed,testbef.data$y)
```