Università di Catania

Pima Indians Diabetes

Report

A.A 2021/2022

Prof: Ingrassia Salvatore

Teaching: Statistical Learning

ALGHASI MAHTAB – CORALLO FABIO – SIMONE ALESSIA

INDEX

INTRODUCTION[1]

This report is addressed to medical centres to help them predict whether the patient has diabetes or not, thanks to some measurements.

The dataset is taken from the National Institute of Diabetes and Digestive and Kidney Diseases[2]. Several constraints were placed on selecting these instances from a more extensive database; in particular, all patients are females at least 21 years old of Pima Indian heritage.

This report aims to predict the presence of diabetes in the response variable: the "class variables" is a binary variable that expresses 0 if the patient has no diabetes, one instead.

The dataset contains eight predictors and 503 observations:

	Type	Description
Pregnant times	Discrete	number of times pregnancies that the patient has had
Glucose tolerance test	Continuous	Plasma glucose concentration after 2 hours of the oral glucose tolerance test. It measures how well the body can process a large amount of sugar
Diastolic blood pressure	Continuous	(mm Hg) the measure of the diastolic blood pressure
Triceps skinfold thickness	Continuous	(mm) measures the body fat
Serum insulin	Continuous	(mu U/ml) therapeutic tool for diagnosed diabetes through the measurement of serum insulin level every 2 hours
Body Mass Index	Continuous	(weight in kg/(height in m)^2) the measure of body fat based on height and weight
Diabetes pedigree function	Continuous	a function that scores the likelihood of diabetes based on family history
Age	Continuous	(years) age of the patients

Table 1 - Variables description

[1] For technical details, see Appendix A

[2] Pima Indians Diabetes Database | Kaggle

In performing the analysis, we want to help medical centres to answer some crucial questions:

- *Is there at least one of the measurements that could help predict the presence of diabetes?*
- *Is there a subset of measurement or all the measurements that can help medical centres predict it?*
- *How well does the final model, among different models, fit the data, and how accurate is our prediction?*

EXPLORATORY DATA ANALYSIS[3]

2.1 Univariate analysis

In this part, we provided an exploratory data analysis of the training set, which contains about 60% of the observations of the original one without any missing values. Therefore, we consider first the univariate analysis of each variable's distribution to understand if there is some measurement that we are interested in the most.

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
pregnant_times	0	1	FALSE	17	1: 97, 0: 71, 2: 71, 3: 51
Class_variable	0	1	FALSE	2	No: 317, Yes: 186

Table 2 - Discrete variables' statistics

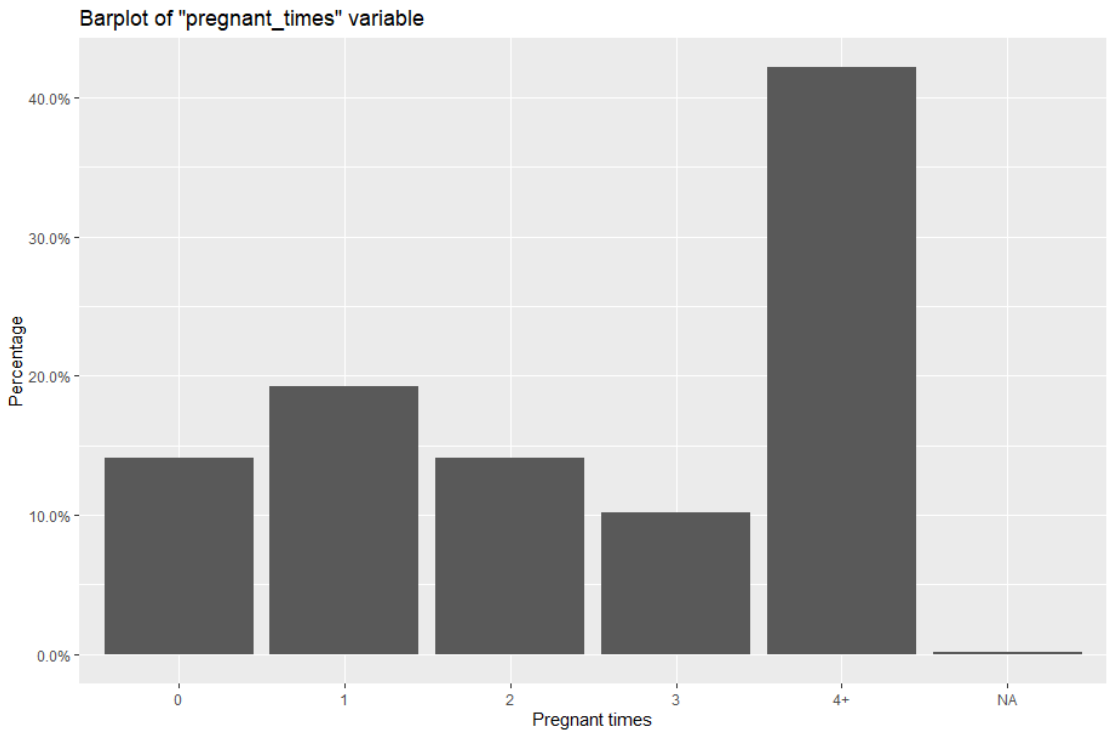

Figure 1 - "pregnant times" barplot

[3] For technical details, see Appendix B

The variable "pregnant_time" is discrete, and it is not balanced: table 2 shows that 71 women had no pregnancies, 97 women had one pregnancy, 71 women had two pregnancies, and 51 women had three pregnancies, and, as we can see in figure 1, remaining women had from 4 to 17 pregnancies.

The "Class_variable" is the response variable and it is a binary; it contains two values: yes if the patient has diabetes and no if the patient has not got diabetes. It is not a balanced distribution as 186 out of 503 patients have diabetes.

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
glucose_tolerance_test	0	1	121.08	31.45	0.00	100.00	118.00	140.00	197.00	▁▁▆▃
Diastolic_blood_pressure	0	1	69.57	18.85	0.00	64.00	72.00	80.00	122.00	▁▁▇▂
Triceps_skin_fold_thickness	0	1	20.87	15.90	0.00	0.00	23.00	33.00	99.00	▇▆▂▁
serum_insulin	0	1	76.35	105.62	0.00	0.00	0.00	131.00	680.00	▆▂▁▁
Body_mass_index	0	1	32.02	7.68	0.00	27.50	32.00	36.35	67.10	▁▃▇▂
Diabetes_pedigree_function	0	1	0.48	0.34	0.08	0.24	0.38	0.62	2.42	▇▃▁▁
Age	0	1	33.46	11.96	21.00	24.00	29.00	41.00	81.00	▇▃▂▁

Table 3 - Continuous variables' statistics

Regarding the continuous variables: the glucose tolerance test values go from 0 to 197, the average value is 121.08, at least 25% of patients have a glucose tolerance test value of 100, and at least half of the patients have a glucose tolerance test value of 118, and at least 75% of the patients have glucose tolerance test of 140.

The diastolic blood pressure values go from 0 to 122, the average value is 69.57, at least 25% of patients have a diastolic blood pressure value of 64, at least half of the patients have a diastolic blood pressure value of 72, and at least 75% of the patients have a diastolic blood pressure value of 140.

The triceps skinfold thickness values go from 0 to 99, the average value is 20.87, at least 25% of patients have a triceps skinfold thickness value of 0, at least half of the patients have a triceps skinfold thickness value of 23, and at least 75% of the patients have triceps skinfold thickness value of 33.

The serum insulin values go from 0 to 0.00, the average value is 76.35, at least 25% of patients have a serum insulin value of 0, at least half of the patients have a serum insulin value of 0, and at least 75% of the patients have a serum insulin value 131.

The BMI values go from 0 to 0.00, the average value is 32.02, at least 25% of patients have a BMI value of 27.50, at least half of the patients have a BMI value of 32.00, and at least 75% of the patients have a BMI value of 36.35.

The age values go from 21 to 81; the average value is 33.46, at least 25% of patients are 24, at least half of the patients are 29, and at least 75% of the patients are 41.

The diabetes pedigree function values go from 0.08 to 2.42; the average value is 0.48, at least 25% of patients have a diabetes pedigree function value of 0.24, at least half of the patients have a diabetes pedigree function value of 0.38, and at least 75% of the patients have a diabetes pedigree function value of 0.62.

2.2 Multivariate analysis

In this part of the analysis, we analyze more than one variable together. In particular, we analyze the behaviour of the distributions according to the presence of diabetes.

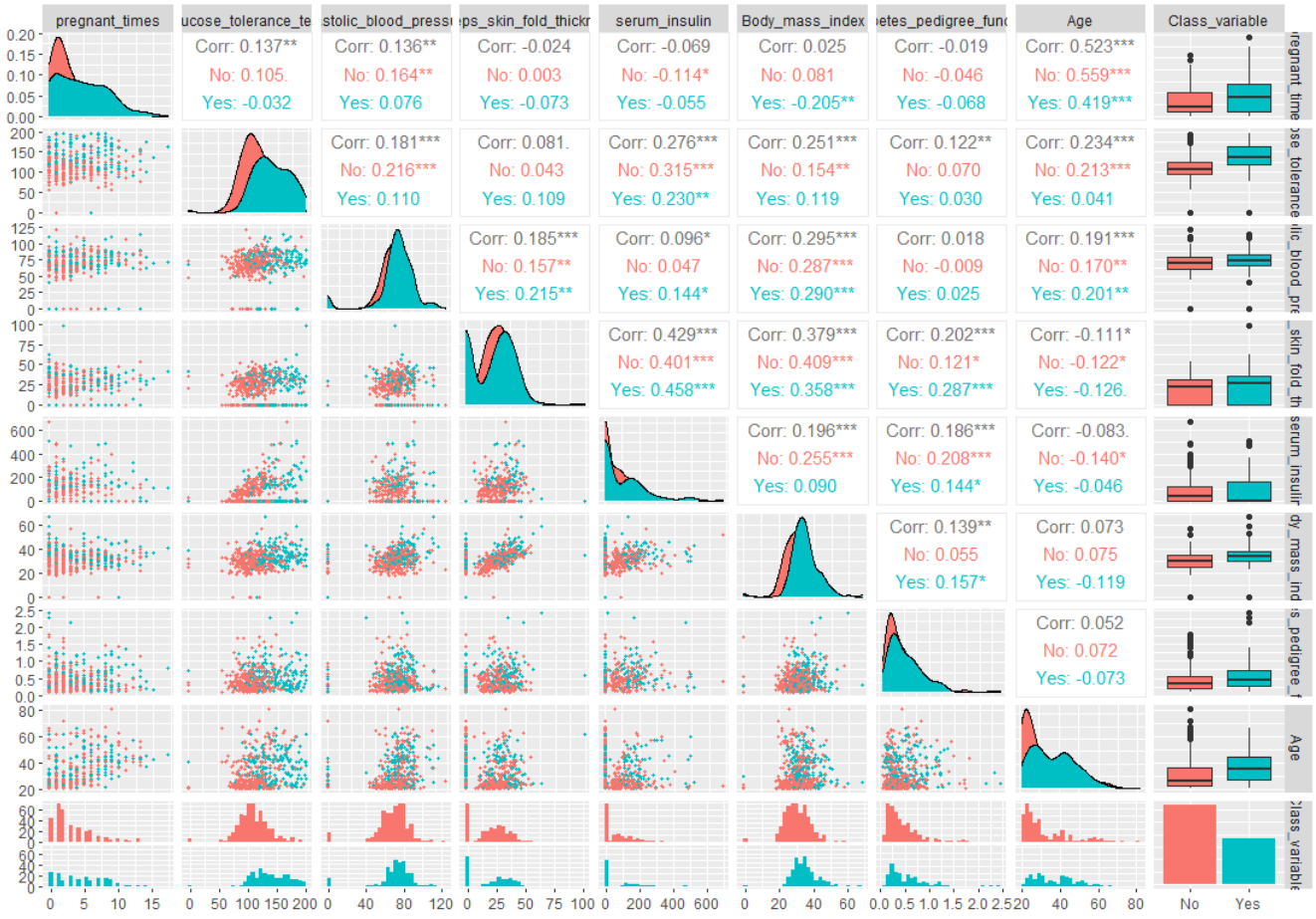

Figure 2 - Multivariate conditional scatterplot

The scatterplot represents the conditional distribution of the variables according to diabetes, the pink points represent the absence of diabetes, and the light-blue points represent the presence of diabetes.

By looking at the upper triangle of the scatterplot, we notice that, in general, there is not a pairwise with a high level of correlation; this can help us to perform a more accurate prediction. However, the most positive correlated pairwise is "Age" and "pregnant_times", in detail: a high number of pregnancies corresponds to older women.

By looking at the diagonal, we can see the probability density functions of the variables according to the presence of diabetes. In detail, we can notice that the highest the number of pregnancies, the highest the probability of having diabetes. Patients with glucose tolerance test values higher than about 125 have a high probability of diabetes. Patients with diastolic blood pressure higher than 70 mm/Hg have a high possibility of diabetes. Triceps skinfold thickness from 10 to 30 millimetres means a low probability of diabetes. The lowest serum insulin diminishes the probability of having diabetes. Patients with BMI over 30 have a high probability of having diabetes.

The lowest diabetes pedigree function means a low probability of having diabetes. Adult people have the highest probability of having diabetes.

Finally, by looking at the boxplot of the variable pregnant times, we can see that the variables glucose tolerance test and triceps skinfold thickness have fewer outliers than the other variables. Likewise, by looking at the boxplots of the class variables, we can notice that the variable triceps skinfold thickness has not got any outliers.

MODELS ASSESSING[4]

3.1 Logistic Regression

In this first section of the chapter, we applied a parametric approach to the training set to predict the probability of each patient having diabetes or not. In detail, the logistic regression method allows us to estimate the parameters which detect a linear relationship between the predictors and the response variable.

The optimal model has been chosen by applying a mixed selection algorithm to select predictors: the initial step starts with the set of all predictors and the multiplication between the variable "Age" and the variable "pregnant_times" as there was a positive correlation of slightly over 0.5:

	Df	Deviance	AIC
-Triceps_skin_fold_thickness	1	477.28	495.28
-Serum_insulin	1	477.50	495.50
<none>		476.44	496.44
-Diastolic_blood_pressure	1	479.96	497.96
-Diabetes_pedigree_function	1	484.04	502.04
-pregnant_times:Age	14	487.34	505.34
-Body_mass_index	1	502.43	520.43
-glucose_tolerance_test	1	559.97	577.97

Table 4 - 1° step mixed variable selection algorithm

The initial AIC value is 496.44; the first step of the algorithm suggests an improvement of our model in terms of the lowest AIC by excluding the "triceps_skin_fold_thickness".

	Df	Deviance	AIC
<none>	1	477.28	495.28
-Serum_insulin	1	479.57	495.57
+Triceps_skin_fold_thickness	1	476.44	496.44
-Diastolic_blood_pressure	1	481.29	497.29
-Diabetes_pedigree_function	1	484.43	500.43
-pregnant_times:Age	14	488.59	504.59
-Body_mass_index	1	502.90	518.90
-glucose_tolerance_test	1	563.25	579.25

Table 5 - 2° step mixed variable selection algorithm

In this second step the initial value of AIC is 495.28 and neither by deleting not adding any other variable we can achieve a model improvement with lowest AIC and deviance value together. For this reason, we consider this as optimal model by taking into consideration the following function:

[4] For technical details, see Appendix C

*Class_variable = -9.731 + 0.549 pregnant_times + 0.037 glucose_tolerance_test – 0.013 Diastolic_blood_pressure – 0.002 serum_insulin + 0.084 Body_mass_index + 0.958 Diabetes_pedigree_function + 0.053 Age -0.010 pregnant_times * Age*

In detail: by increasing pregnant times, the probability of diabetes increases by 55%, by increasing glucose tolerance test, the probability of diabetes increases by 4%, by increasing diastolic blood pressure, the probability of diabetes decreases by 1%; by increasing serum insulin, the probability of diabetes decrease of 0.2%, by increasing body mass index the probability of diabetes increase of 8%, by increasing diabetes pedigree function the probability of diabetes increase of 96%, by increasing the age the probability of diabetes increase of 5% and the older are the pregnant woman, the less probability they reach to have diabetes (1%).

Figure 3 below shows the logistic regression function as a straight blue line, the x-axis represents the predicted probability, and the y-axis represents the outcome:

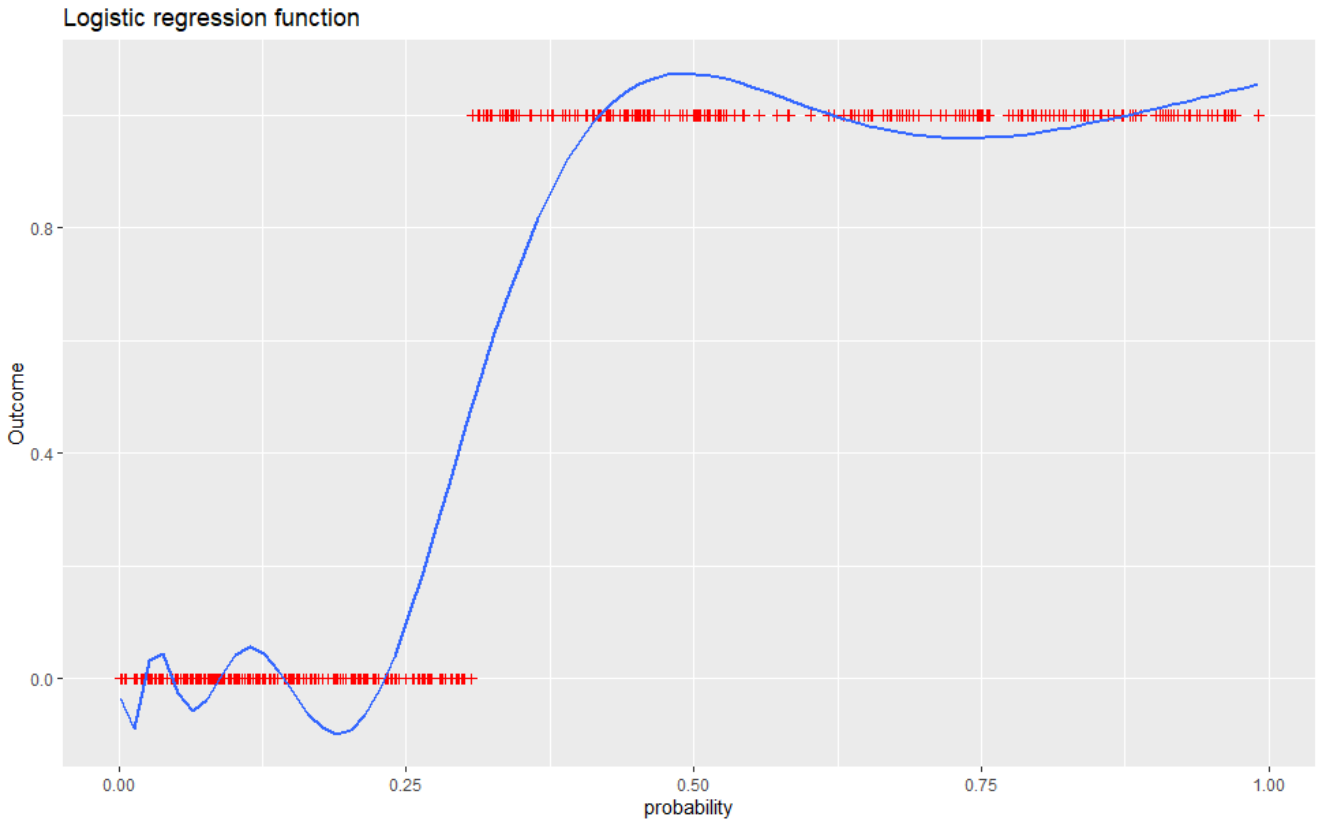

Figure 3 - Logistic regression function

3.1.1 Goodness of fit

Once the optimal model has been detected, we used it to predict the probabilities for each observation to have diabetes or not. Then we compare the predicted results with the real results through a contingency matrix to understand how accurate our model is.

The optimal threshold point has been computed throw a function that detects the higher value between sensitivity and specificity: prediction with a probability higher than 0.35 belongs to the presence of diabetes.

	No	Yes	Sum
No	244	43	287
Yes	73	143	216
Sum	317	186	503

Table 6 – Logistic regression contingency matrix on train set

By analyzing Table 6, we can see in the diagonal that a high percentage of observations matched the prediction; the model's accuracy is 76.94%, with a train error rate of 23.06%.

Table 7 shows the contingency matrix computed by using the actual values of the validation set, which contains about 20% of the units of the original dataset, and the predicted results:

	No	Yes	Sum
No	81	8	89
Yes	32	46	78
Sum	113	54	167

Table 7 – Logistic regression contingency matrix on the validation set

Table 7 shows us that, in this case, there is a high percentage of matched observations. As a result, the accuracy rate is 76.05%, and the error rate is 23.95%.

3.2 Random Forest

In this second section, we applied a non-parametric model to predict the presence of diabetes: the random forest model allows us to build several classification trees that are not correlated with each other by giving a certain weight (importance) to the variables. The final model is an average of the resulting trees.

The algorithm starts to predict the "Class variable" by using all the predictors within 500 classification trees. By training the random forest model, we discover that by using four predictors, we can gain the lowest Out-Of-Bag error (about 20%) with a class error of 15% for the prediction of the absence of diabetes and 33% for the prediction of the presence of diabetes.

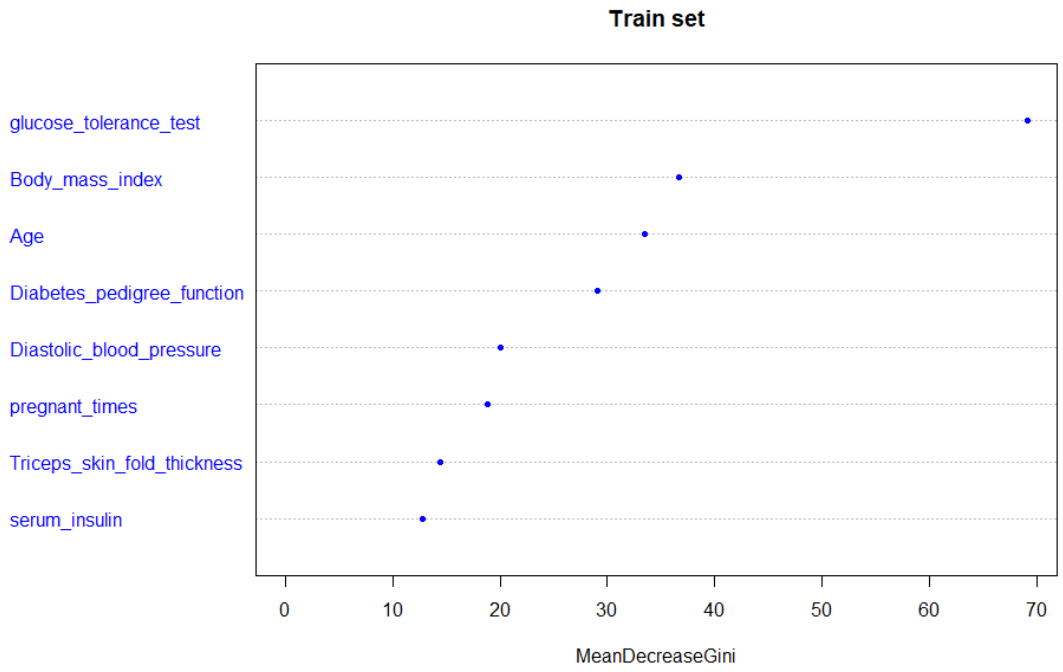

Figure 4 - Variable's importance

In figure 3, we can see the ranking predictors based on how much they influence the response: we can affirm that, according to the Gini index, the glucose tolerance test, body mass index, age and diabetes pedigree function are essential variables in our model.

We can plot one sample tree among 500 trees to get an idea about our model:

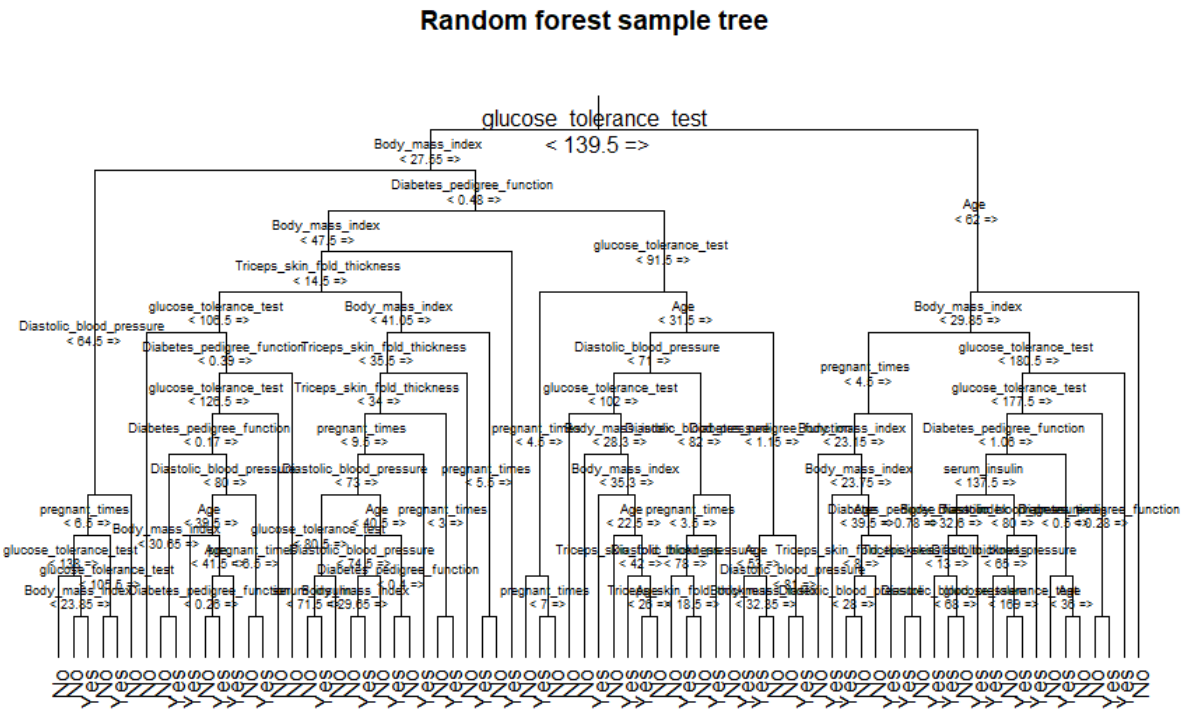

Figure 5 - Random forest sample tree

3.2.1 Goodness of fit

Once the optimal model has been detected, we used it to predict the probabilities for each observation to have diabetes or not. Then we compare the predicted results with the real results through a contingency matrix to understand how accurate our model is.

The optimal threshold point has been computed throw a function that detects the higher value between sensitivity and specificity.

	No	Yes	Sum
No	267	50	317
Yes	62	124	186
Sum	329	174	503

Table 8 - Random forest contingency matrix on train set

Table 8 shows a 22% error rate with an accuracy rate of about 77%.

To better understand the accuracy of our model, in this part, we predict the response variable on the validation set by using the previous random forest model.

	No	Yes	Sum
No	92	21	113
Yes	16	38	54
Sum	108	59	167

Table 9 - Random forest contingency matrix on the validation set

As we can see from table 9, most of the predictions matched the actual value, and the model performed an accuracy rate of 77.24% and an error rate of 22.75% on the validation set.

3.3 Neural Network

In this third section, we applied the neural network model to build an artificial neuron that allows us to predict the presence of diabetes throws a deep learning method. This approach is considered a parametric approach because of the estimation of weights in the function. However, first, as a neural network works better when observations are scaled, we build a matrix containing the scaled values to have a mean equal to 0 and standard deviation equal to 1 to compare predictors better.

We build a single layer perceptron model throw Keras that contains: the first input layer made up of 8 input neurons that correspond to the predictors, the hidden layer containing six neurons[5] in which ReLU function and a dropout rate of 0.3 has been applied to avoid the overfitting, the output layer containing one neuron in which sigmoidal function has been applied to allow the production of results from 0 to 1, that corresponds to our response value. About the compiler, we set the "Adam" optimizer with accuracy metrics and binary cross-entropy as loss measures as we deal with classification.

Layer (type)	Output Shape	Param #
Dense_1 (Dense)	(None, 6)	54
Dropout (Dropout)	(None, 6)	0
Dense (Dense)	(None, 1)	7

Table 10 - Sequential model

Table 10 shows the frame of our neural network model: the hidden layer contains 54 parameters represented by weights, and the output layer contains seven parameters, for a total amount of 61 parameters.

3.3.1 Goodness of fit

Before predicting the response and comparing them throw a confusion matrix in both the train and validation set, we fit the model with the mini-batch gradient descent to avoid overfitting: we set a batch size of 5 that iterates in 100 epochs.

[5] The number of neurons into the hidden layer has been computed by follow the formula: [(n. of input + n.of output) / 2] +1

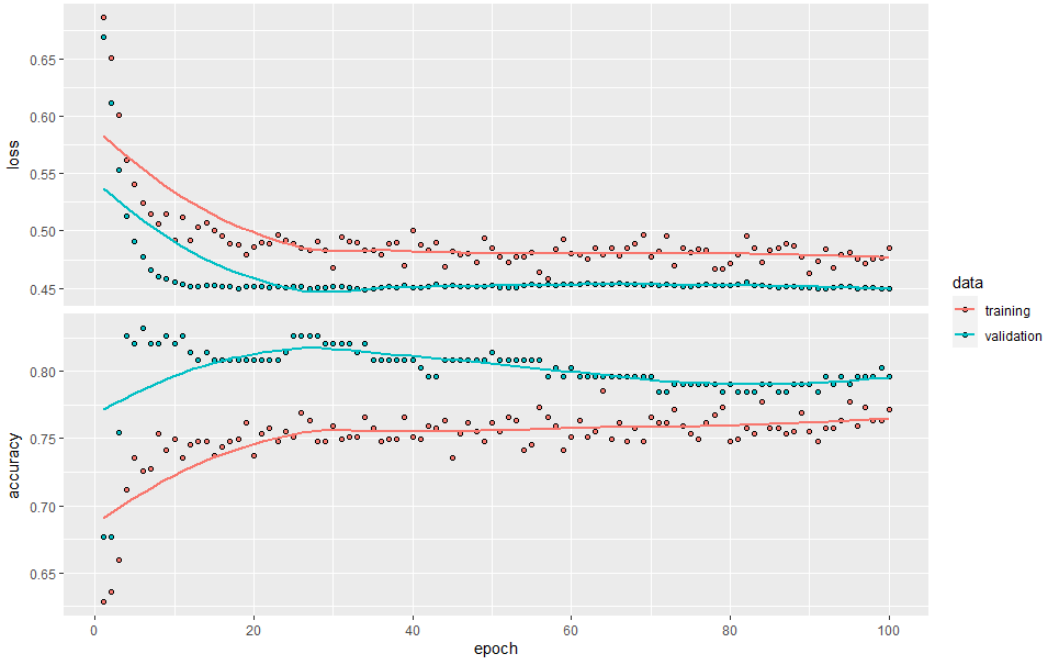

Figure 6 - Neural network history

As shown in figure 4, the loss rate tends to decrease for both the train and validation set and the accuracy rate tends to increase for both the train and validation set.

Now, we build the confusion matrix that allows us to compare the predicted value with the actual response of the train set by taking into consideration the optimal value of the threshold:

	No	**Yes**	**Sum**
No	237	33	270
Yes	80	153	233
Sum	317	186	503

Table 11 - Neural network contingency matrix on train set

The contingency matrix of the train set shows about 77% of accuracy rate and about 22% of error rate.

Now we compare prediction and actual response in the validation set to analyze how our model behaves in a new set of observations:

	No	Yes	Sum
No	93	12	105
Yes	20	42	62
Sum	113	54	167

Table 12 - Neural network contingency matrix on the validation set

The contingency matrix of the validation set shows about 80% of accuracy rate and about 19% error rate.

CONCLUSIONS[6]

In this last part of the analysis, we compare the ROC curves of our models on the validation set, and we choose the one with the highest area under the ROC curve (AUC value):

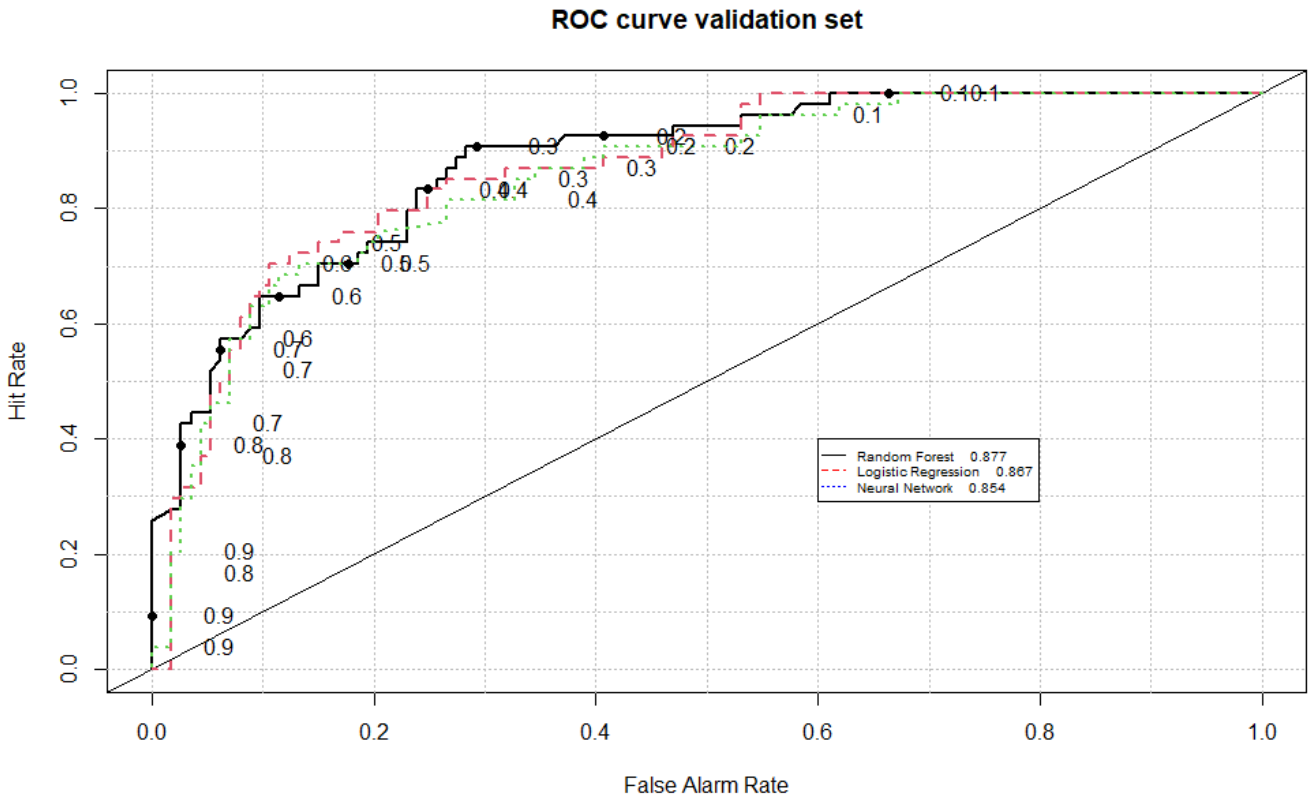

Figure 7 - ROC Curve validation set

According to the ROC Curve and AUC value of the validation set, we can see that the random forest (black straight line) appears to be the best model to predict diabetes. For this reason, we applied the random forest model to the test set to predict the presence of diabetes.

At this point, we can answer our business questions about the presence of diabetes:

Is there at least one of the measurements or a subset of them that could help predict the presence of diabetes?

[6] For technical details, see Appendix D

The random forest algorithm's most critical measurements to predict diabetes are glucose tolerance test, body mass index, age, and diabetes pedigree function.

How well does the final model, among different models, fit the data, and how accurate is our prediction?

In both the train and validation set, the accuracy rate is about 77%, with the higher AUC value of 0.87 among the three models on the validation set.

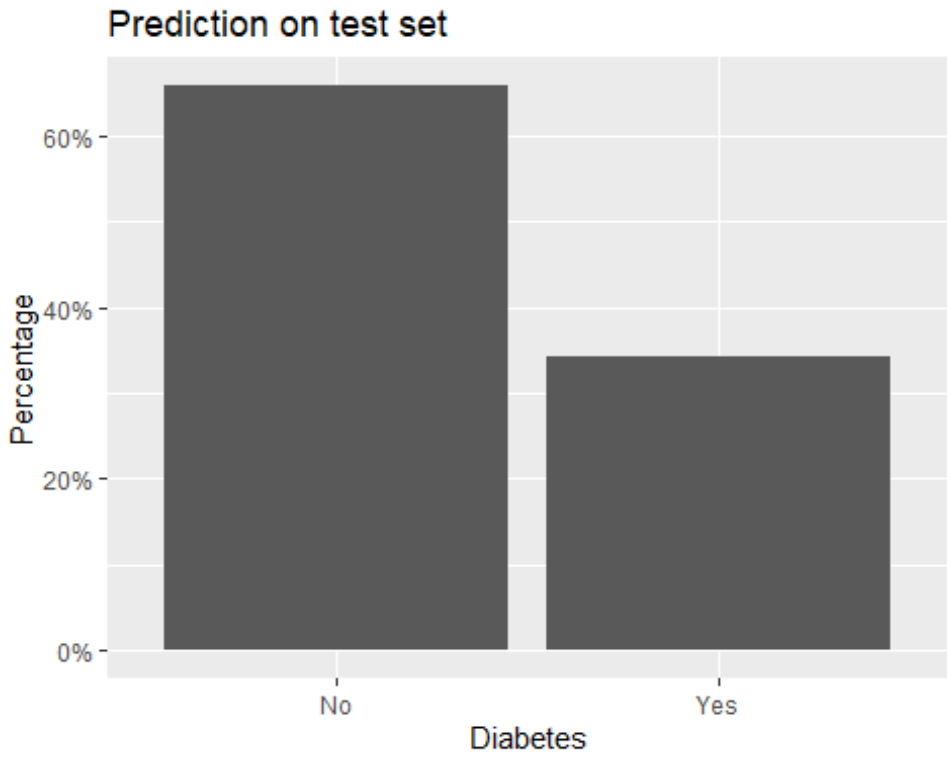

Figure 8 - Prediction barplot on the test set

Our test set contains about 60% of patients who did not have diabetes and about 30% of patients who have diabetes.

However, it is possible to improve further analysis by considering a mix of the three models instead of choosing a single one. For example, we can build a better neural network model by checking for linearity thanks to the logistic regression to apply the hidden layers and then select the essential variables thanks to the random forest.

APPENDIX A

Used library:

```
library(visdat)
library(skimr)
library(DataExplorer)
library(corrplot)
library(ggplot2)
library(GGally)
library(Deducer)
library(MASS)
library(randomForest)
library(keras)
library(verification)
library(dplyr)
```

Fixing dataset and structure:

```
dataset <- read.csv(file.choose())
row.names(dataset) <- dataset$X
dataset <- dataset[,2:10]
dataset$Class_variable <- factor(dataset$Class_variable, labels=c('No', 'Yes'))
attach(dataset)

str(dataset)
```

```
## 'data.frame':    503 obs. of  9 variables:
##  $ pregnant_times             : Factor w/ 17 levels "0","1","2","3",..: 7 2 2 1
 4 11 6 8 1 8 ...
##  $ glucose_tolerance_test     : int  148 85 89 137 78 139 166 100 118 107 ...
##  $ Diastolic_blood_pressure   : int  72 66 66 40 50 80 72 0 84 74 ...
##  $ Triceps_skin_fold_thickness: int  35 29 23 35 32 0 19 0 47 0 ...
##  $ serum_insulin              : int  0 0 94 168 88 0 175 0 230 0 ...
##  $ Body_mass_index            : num  33.6 26.6 28.1 43.1 31 27.1 25.8 30 45.8 2
9.6 ...
##  $ Diabetes_pedigree_function : num  0.627 0.351 0.167 2.288 0.248 ...
##  $ Age                        : int  50 31 21 33 26 57 51 32 31 31 ...
##  $ Class_variable             : Factor w/ 2 levels "No","Yes": 2 1 1 2 2 1 2 2
2 2 ...
```

APPENDIX B

Univariate analysis:

```
skim(dataset)

dataset$pregnant_times <- cut(dataset$pregnant_times, breaks=c(0,1,2,3,4,17), labe
ls= c('0','1','2','3','4+'), right=FALSE)

ggplot(dataset,aes(x=pregnant_times))+
  geom_bar(aes(y = (..count..)/sum(..count..)))+
  scale_y_continuous(labels=scales::percent)+
  ylab('Percentage')+
  xlab('Pregnant times')+
  ggtitle('Barplot of "pregnant_times" variable')
```

Multivariate analysis:

```
ggpairs(dataset, aes(colour=Class_variable), lower = list(continuous = wrap("point
s", size=0.6)), cardinality_threshold=17)
```

APPENDIX C

Logistic regression:

```
glm <- stepAIC(glm(Class_variable ~. + Age*pregnant_times, data = dataset, family
= binomial), direction='both')

glm.probs <- predict.glm(glm, dataset, type='response')
prdBln <- ifelse(glm.probs > threshold(glm.probs, dataset$Class_variable), 1 , 0)
dfrPlot <- mutate(dataset, glm.probs= glm.probs, POutcome = prdBln)
ggplot(dfrPlot, aes(x=glm.probs, y=POutcome)) + geom_point(shape=3, colour='red',
fill='red') + geom_smooth(method='gam', formula=y~s(log(x)), se=F) + labs(title='L
ogistic regression function') + labs(x='probability') + labs(y='Outcome')
```

Goodness of fit:

```
threshold <- function(predict, response) {
  perf <- ROCR::performance(ROCR::prediction(predict, response), "sens", "spec")
  df <- data.frame(cut = perf@alpha.values[[1]], sens = perf@x.values[[1]], spec =
 perf@y.values[[1]])
  df[which.max(df$sens + df$spec), "cut"]}[7]

N <- nrow(dataset)
glm.probs <- predict.glm(glm, type='response')
glm.pred <- rep('No', N)
glm.pred[glm.probs>.5]='Yes'
(confMat <- addmargins(table(glm.pred, dataset$Class_variable)))

(accuracy <- (confMat[1,1]+confMat[2,2])/N*100)

(Err <- 100-accuracy)

row.names(datasetV) <- datasetV$X
datasetV <- datasetV[,2:10]
datasetV$Class_variable <- factor(datasetV$Class_variable, labels=c('No', 'Yes'))
attach(datasetV)

nV <- nrow(datasetV)
glm.probsV <- predict.glm(glm, newdata=datasetV, type='response')
```

[7] data_analysis_codes/ABOD leave one out6192019.R at master · mr-neuroanatomy/data_analysis_codes · GitHub

```
glm.predV <- rep('No', nV)
glm.predV[glm.probsV>.5]='Yes'
(confMatV <- addmargins(table(glm.predV, datasetV$Class_variable)))

(accuracyV <- (confMatV[1,1]+confMatV[2,2])/nV*100)

(ErrV <- 100-accuracyV)
```

Random forest:

```
set.seed(2612)
(rf <- tuneRF(x = subset(dataset, select = -Class_variable), y = dataset$Class_var
iable, ntreeTry = 500, plot = F, mtryStart=8, doBest=T))

varImpPlot(rf, col="blue", pch=20, main="Variables' importance")

to.dendrogram <- function(dfrep,rownum=1,height.increment=0.1){

  if(dfrep[rownum,'status'] == -1){
    rval <- list()

    attr(rval,"members") <- 1
    attr(rval,"height") <- 0.0
    attr(rval,"label") <- dfrep[rownum,'prediction']
    attr(rval,"leaf") <- TRUE

  }else{
    left <- to.dendrogram(dfrep,dfrep[rownum,'left daughter'],height.increment)
    right <- to.dendrogram(dfrep,dfrep[rownum,'right daughter'],height.increment)
    rval <- list(left,right)

    attr(rval,"members") <- attr(left,"members") + attr(right,"members")
    attr(rval,"height") <- max(attr(left,"height"),attr(right,"height")) + height.
increment
    attr(rval,"leaf") <- FALSE
    attr(rval,"edgetext") <- paste(dfrep[rownum,'split var'],"\n<",round(dfrep[row
num,'split point'], digits = 2),"=>", sep = " ")}

  class(rval) <- "dendrogram"
```

```
  return(rval)}[8]

tree <- getTree(rf, k=1, labelVar = TRUE)

d <- to.dendrogram(tree)

plot(d, center=TRUE, edgePar=list(t.cex=.55, p.col=NA, p.lty=0), yaxt = "n", main
= paste("Random forest sample tree”))
```

Goodness of fit:

```
N <- nrow(dataset)
random.probs <- predict(rf, data=dataset, type='class')
(matrix <- addmargins(table(dataset$Class_variable, random.probs)))

(accuracy <-(matrix[1,1]+matrix[2,2])/N*100)

(error <- 100-accuracy)

randomforestV <- predict(rf, newdata = datasetV, type='class')
(matrix <- addmargins(table(datasetV$Class_variable,randomforestV)))

(accuracy <-(matrix[1,1]+matrix[2,2])/nV*100)

(error <- 100-accuracy)
```

Neural network:

```
train_data <- as.matrix(scale(dataset[1:8]))
train_labels <- dataset$Class_variable

model <- keras_model_sequential()

model %>%
  layer_dense(units = 6, activation = 'relu', kernel_initializer='uniform', input_
shape = c(8)) %>%
  layer_dropout(rate=0.3) %>%
  layer_dense(units = 1, activation = 'sigmoid', kernel_initializer='uniform')

model %>% compile(
  optimizer = "adam",
  loss = "binary_crossentropy",
```

[8] dendextend/as.dendrogram.randomForest.R at master · talgalili/dendextend · GitHub

```
  metrics = c("accuracy")
)
summary(model)
```

Goodness of fit:

```
val_data <- as.matrix(scale(datasetV[1:8]))
val_labels <- datasetV$Class_variable

set.seed(2612)
history <- model %>% fit(
  train_data, train_labels,
  epochs =100, batch_size = 5,
  validation_data = list(val_data, val_labels),
  verbose=0
)
plot(history)

nn.pred <- rep('No', N)
nn.pred[model %>% predict(train_data) > threshold(model %>% predict(train_data), d
ataset$Class_variable)]='Yes'
(confMat <- addmargins(table(nn.pred, dataset$Class_variable)))

(accuracy <- (confMat[1,1]+confMat[2,2])/N*100)

(Err <- 100-accuracy)

nn.predV <- rep('No', nV)
nn.predV[model %>% predict(val_data) >threshold(model %>% predict(val_data), datas
etV$Class_variable)]='Yes'
(confMat <- addmargins(table(nn.predV, datasetV$Class_variable)))

(accuracy <- (confMat[1,1]+confMat[2,2])/nV*100)

(Err <- 100-accuracy)
```

APPENDIX D

```
rocplotval <- roc.plot(x = (datasetV$Class_variable == "Yes"), pred = cbind(random
.probsV[,2], glm.probsV, predictnnV), main = "ROC curve", legend = T, leg.text = c
("Random Forest", "Logistic Regression", "Neural Network"))

rownames(datasetT) <- datasetT$X
datasetT <- datasetT[2:10]
datasetT$Outcome <- predict(rf, newdata = datasetT, type='response')

head(datasetT)

tail(datasetT)

ggplot(datasetT,aes(x=Outcome))+
  geom_bar(aes(y = (..count..)/sum(..count..)))+
  scale_y_continuous(labels=scales::percent)+
  ylab('Percentage')+
  xlab('Diabetes')+
  ggtitle('Prediction on test set')

write.csv(datasetT, 'testoutcome.csv')
```

TABLES AND FIGURES INDEX