Larry Linares
Nicolas Garcia

Machine Learning, I

Assignment 1:

Classification

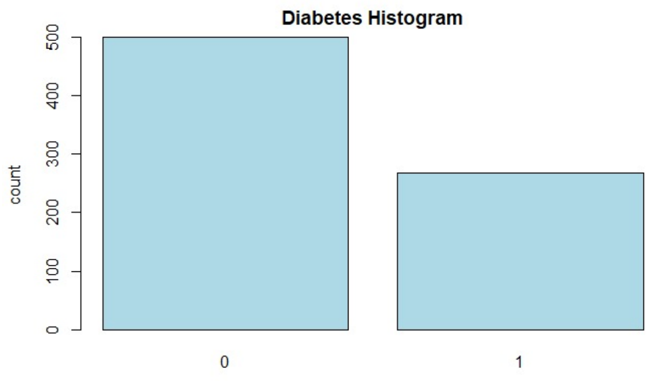
Explanatory Data Analysis (EDA):

First of all, we have loaded our data on diabetes into R. Once we have our data, we begin with the EDA. This particular dataset contains data from 768 21-year-old or older Pima Indian women. The goal of this analysis is to determine, given certain independent variables, whether the Pima Indian women has diabetes or not.

In the loaded dataset, we have 9 different variables:

- PREGNANT: Number of times the women has gotten pregnant
- GLUCOSE: Glucose level 2 hours after an oral glucose tolerance test
- BLOODPRESSURE: Diastolic blood pressure (mm Hg)
- SKINTHICKNESS: Triceps skin fold thickness (mm)
- INSULIN: Insulin level 2 hours after a serum insulin test (mu U/ml)
- BMI: Body mass index (weight (kg)/ height (m)^2)
- DIABETESPEDIGREEFUNCTION: a function which scores likelihood of diabetes based on family history
- AGE: Age when the data was taken
- OUTCOME: 0 = Non-diabetic, 1 = Diabetic

To start the EDA, we first look at the variable outcome, to see the distribution and skewness. First of all, we have converted the outcome variable into a factor, as it groups the population into two different subsets, those who have diabetes, and those who don't. Looking at the bar plot, we can conclude that there is a right tail skewness, as more than 65% of the data is on the left.



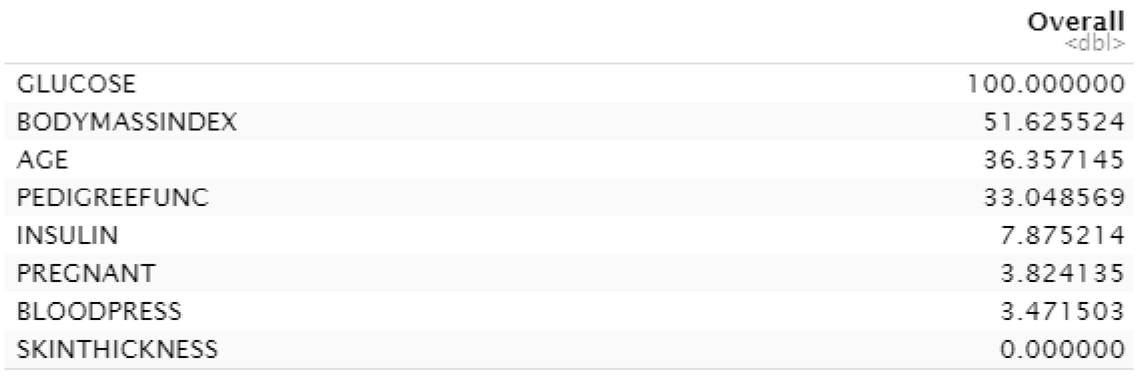
We check for any missing values (NAs), but we do not find any, although, we do find that there are many values that are 0. More specifically, there are 5 for GLUCOSE, 374 for INSULIN, 35 for BLOODPRESSURE, 227 for SKINTHICKNESS, and 11 for BMI. Given that there are only 11 for BMI, we can afford to eliminate all rows which contain a 0. On the other hand, for the rest of the variables that have 0s, we can't afford to simply delete them, as they constitute a large amount of our data set and would cost too much when developing the model. The 374 0 values for INSULIN also do make sense or are at least not outliers. According to medical data, the range for INSULIN levels in women can vary between 1.6 mu U/ml and 13 mu U/ml (Maryam Tohidi, 2014), so it is entirely possible to have values close to or equal to zero.

On the other hand, the variables BLOODPRESSURE and SKINTHICKNESS do not make any sense to have values equal to 0. If a persons BLOODPRESSURE is equal to 0, it would mean that there is no blood flowing, suggesting that the person is not alive. A similar thing happens when

SKINTHICKNESS is equal to 0, it is physically impossible to have a SKINTHICKNESS of 0, as it would mean they have no skin.

To fix this problem, we have decided to substitute the values (0), for their respective median, as it is not influenced by outliers, and gives a better representation of the true values for the variables. The median value for the SKINTHICKNESS is 29, and 72 for BLOODPRESSURE.

Once we have cleaned and prepared our data, we calculate which features are most important for the classification of diabetes. To do this we use an xg boost tree, as, after the boosted trees are constructed, the process to retrieve importance scores is straightforward, compared to other models. The variables we use in the models depend on what scores the xg boost model returns. In our case, the following results were given, suggesting that any variable with a lower importance of 30 will not be used.



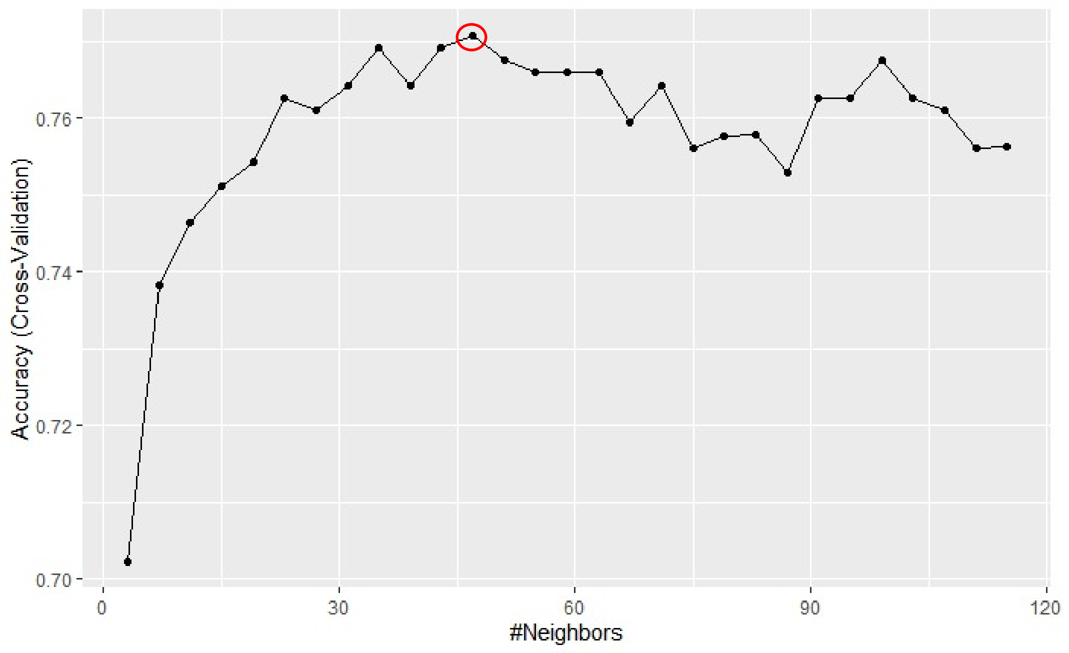
	Overall <dbl>
GLUCOSE	100.000000
BODYMASSINDEX	51.625524
AGE	36.357145
PEDIGREEFUNC	33.048569
INSULIN	7.875214
PREGNANT	3.824135
BLOODPRESS	3.471503
SKINTHICKNESS	0.000000

KNN – Model:

KNN model, or K-Nearest Neighbours, is a machine learning algorithm that classifies unlabelled examples. It does so by considering its nearest data points (neighbours). If a new example is introduced, and is close to, or in a cluster of points, it is very likely that they both should be classified into the same category. This is essentially how KNN works, the closer the data example is to a point, the more likely it should have the same classification.

There is one hyperparameter for a KNN model; K. If K=1, then the algorithm classifies the new example depending on the nearest neighbour. It is very hard to guess what number K should be, therefore we let the algorithm decide the optimal K.

For this specific model, the performance was lower when INSULIN and PREGNANT were considered, therefore we have discarded these variables, along with those that don't have a significant importance on the output variable.



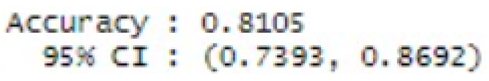
As can be seen in the above graph, the ideal number of neighbours (K) is equal to 47, as it has the highest accuracy, around 81%, and a Kappa value of 0.56, which according to Joseph L. Fleiss, is between fair and good.

To train this model we used cross-validation, as it would give us a better idea of the accuracy the model has. Cross-validation is a resampling method that ensures that the model is trained on different portions of the data set, not just one. This helps with the over-fitting problem, as many different sets of data are used, making over-fitting extremely difficult.

Regarding the accuracy, we want to make sure that the difference between train and test models is not too large, as a large difference would imply that there is some over-fitting in the model.

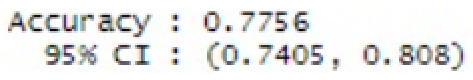
In our case, our test accuracy came to 81%, with most accuracies (due to cross validation) between 73% (strong) and 86% (very strong), while our training set had around 77% accuracy (also strong). Given that our test set had higher accuracy, we can conclude that there does not seem to be any over-fitting in this model.

TEST:



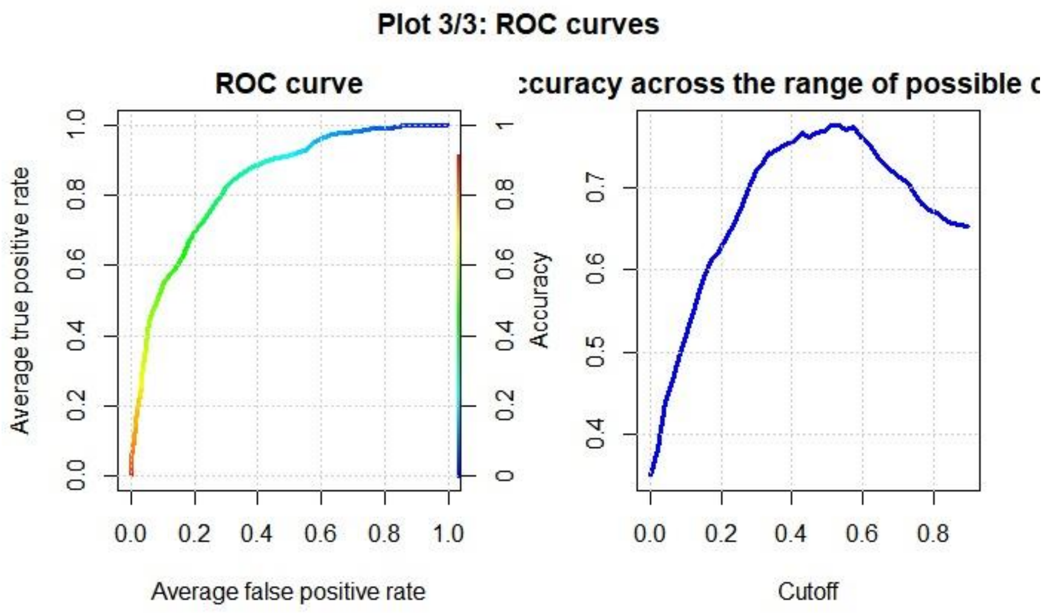
```
Accuracy : 0.8105
  95% CI : (0.7393, 0.8692)
```

TRAIN:



```
Accuracy : 0.7756
  95% CI : (0.7405, 0.808)
```

The ROC curve evaluates the class probabilities for the model across a continuum of thresholds. The curve has two different parameters, True Positive Rate and False Positive Rate. If we lower the threshold, the model classifies more data points as positive, thus increasing both False Positives and True Negatives. The higher the area under curve (0 =<AUC =< 1), the better the performance of the model at distinguishing between the positive and negative classes. In our case,


Plot 3/3: ROC curves
ROC curve
Average true positive rate
Average false positive rate
:curacy across the range of possible c
Accuracy
Cutoff