Heart Disease
Machine Learning 1

Jose Pérez Cano and Álvaro Ribot Barrado

GCED – FIB – UPC

June 3, 2020

Contents

1 Introduction

1.1 Problem

The database we have chosen to work with is about the presence or absence of heart disease in a sample of the population. The dataset consists of 75 attributes, one of them is the presence of heart disease in a scale from 0 to 4. It also contains missing values and dummy variables created to replace some sensible data.

1.2 Reason

We wanted to work with a medical dataset because it is the type of dataset we feel is more practical. At first we have a pool of several problems from which this one got more variables to work with although less patients. This way we got to practice not only the machine learning algorithm but the problem of having too many dimensions and very few patients, which seems more real for us.

1.3 References

We have obtained the data from UCI Machine Learning Repository (more information can be found in this link). It has a long list of papers which cite this dataset.

2 Preprocessing

2.1 Format

The data provided by the UCI repository is not in a nice format to be read directly using R. So we have done a previous preprocessing step converting .data, where each patient was split into several lines, to .csv format, in which the 75 attributes of each patient were altogether in one line, using C++. The repository consists of different datasets, corresponding to different locations but all of them with the same variables (75). This databases are called Cleveland, Hungarian, Switzerland and Long Beach VA. We have begun our study using only the Cleveland database, since it is the one other researchers used. The target variable in this project is $V58$, corresponding to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4 and we will deal with it as a factor.

2.2 Missing values

There are 20 variables with all missing values, so we have deleted them from our Data Set. We have deleted two more variables, one with more than 90% missing values that we also have deleted, and another one with 69 missing values, corresponding to the time where ST measure depression was noted.

Missing values are encoded with -9. Before encoding them with NA we have imputed the remaining missing values using the k nearest neighbour technique, using $k = 7$.

After doing this, we have changed several variables types from integer to factor, according to the dataset description. Also, we've removed some dummy variables which had only one value for all. The number of remaining variables is 45, from the initial 75.

2.3 Data visualization

Initially, we start by doing boxplots and histograms of the different variables to see possible outliers and if the distributions are more or less symmetric. This is the result, it contains symmetric and skewed variables.

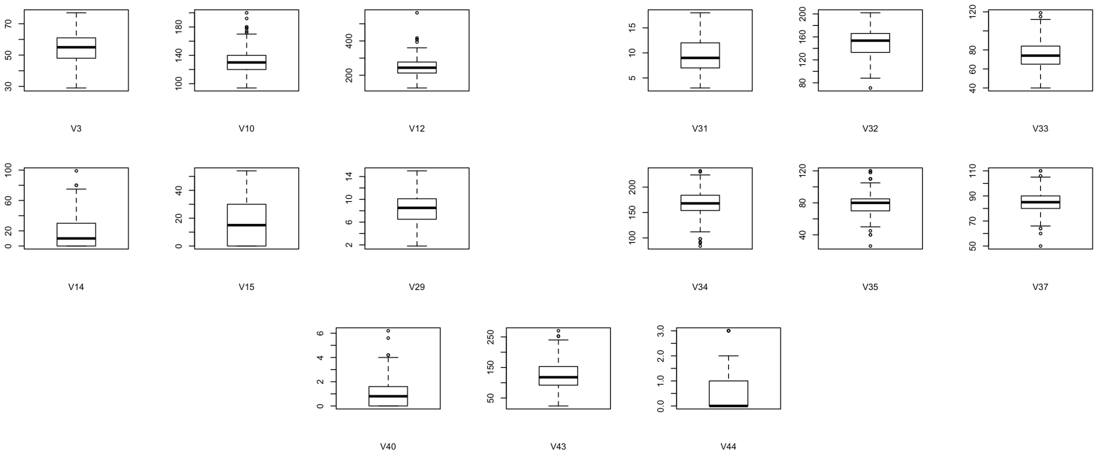


Figure 1: Boxplots of numeric variables

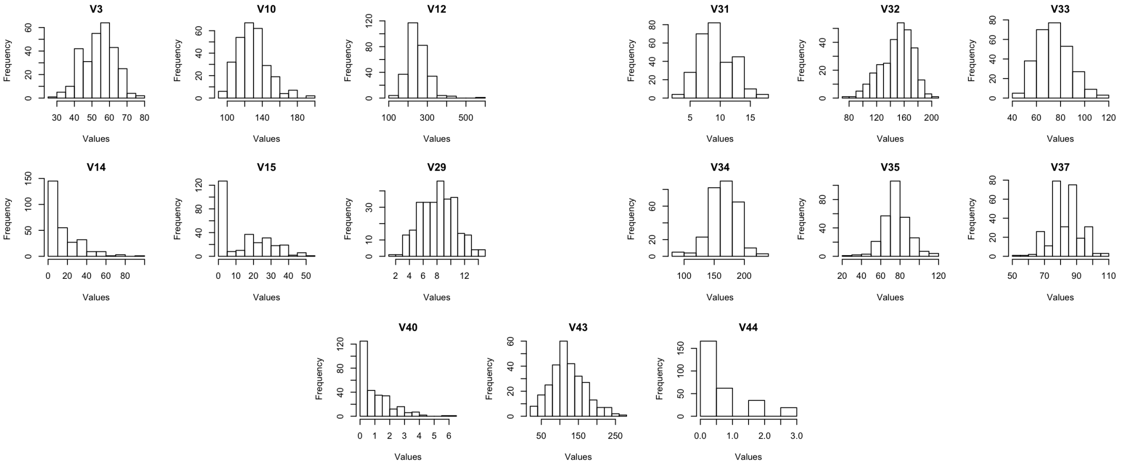


Figure 2: Histograms of numeric variables

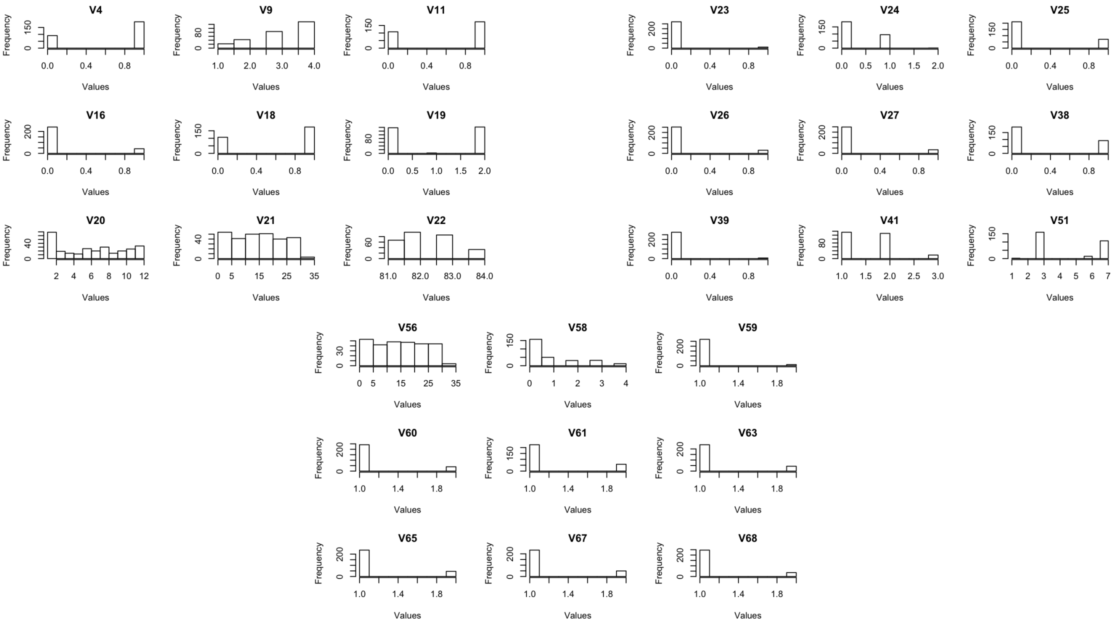


Figure 3: Histograms of factors

2.4 Correlation of variables

The dataset has many variables which are related between them. This insight will be helpful in a possible reduction of variables later on. After looking their description every correlation found seems reasonable. The more correlated pair was the duration of a test exercise and a MET score in a test, with a correlation of 0,93. MET means Metabolic Equivalent of Task, which is simply a measure of how much exercise is done, as is the duration, thus the relation. There are other pairs of moderately correlated variables $(0,5 < cor < 0,7)$, like cigarettes per day and years as a smoker. Also peak exercise blood pressure and resting blood pressure, and maximum heart rate achieved and METs achieved.

2.5 Modification of values

As it was seen previously many variables aren't normal, and most of our models will rely partly on this assumption. Therefore we are going to apply a Box-Cox transformation to approximate the distribution to a Gaussian as much as possible. Furthermore, data is standardised in order to obtain better results, since the ranges of the variables vary substantially across variables.

Some of the variables are already normal and in the Box-Cox the estimated lambda was statistically 1, or the Q-Q plot was a straight line. These were the age of the patient, the maximum heart rate, the peak blood pressure exercise, the resting blood pressure and the duration of the exercise. The estimated transformation for the other variables is a square root transformation.

However, there are three special variables which are years as smoker, cigarettes per day and ST depression induced by exercise relative to rest. They have two groups, one with all zeros, and the other with a variable quantity. We have decided to transform this variables considering the zeros apart, this way, the estimated transformation to cigarettes per day and ST depression is a square root, and years as a smoker is normal if we ignore the zeros.

Since there are non-positive values we decided to subtract the minimum plus some epsilon to be able to use Box-Cox. Another approach could have been to use the Manly transformation which is exponential and it allows to use non-positive values.

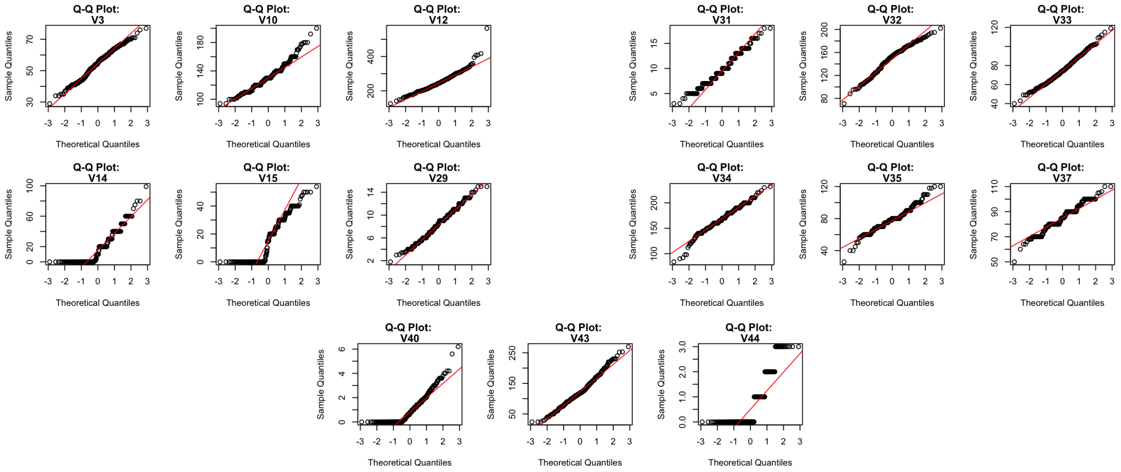


Figure 4: QQ-Plot Before transformation

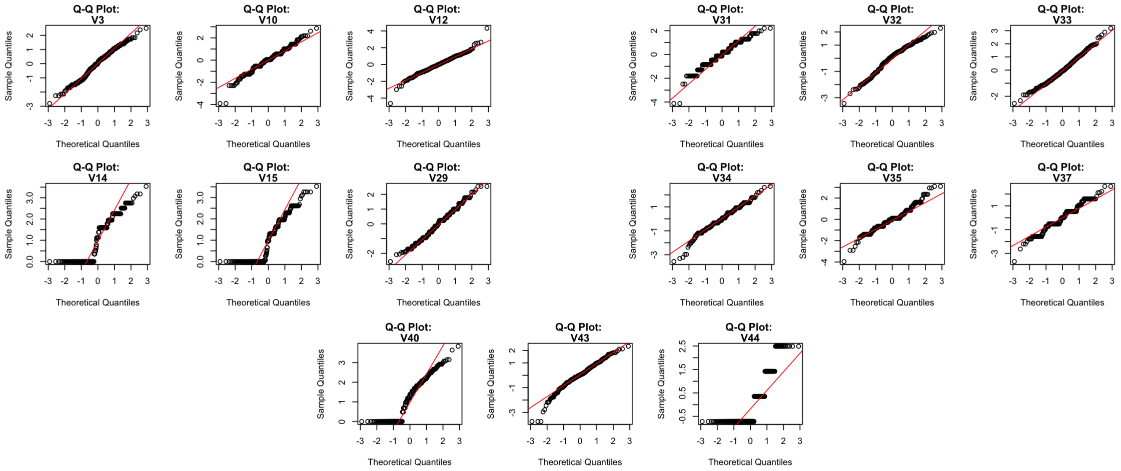
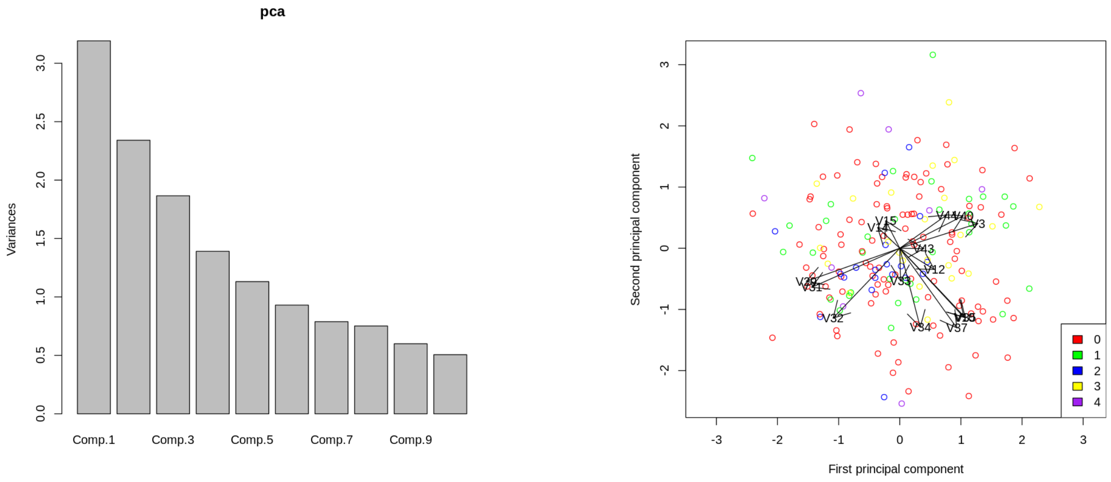


Figure 5: QQ-Plot After transformation

2.6 Feature extraction

2.6.1 PCA

After doing the PCA we have obtained the following screeplot of the variance explained by the principal components. With 2 components we can explain 38% of the total variance, and if we want to explain more than 80% we would need 8 components. The Figure 3 represents the screeplot, where we can see the variance explained by each principal components. We have also computed a biplot to visualize the data in two dimensions. As we expected, it is quite difficult the interpretation of this biplot and we cannot distinguish the different levels of the target variable. Because of that we will not use the PCA in our models.



(a) Screeplot

(b) Biplot

Figure 6: PCA plots

2.6.2 FDA

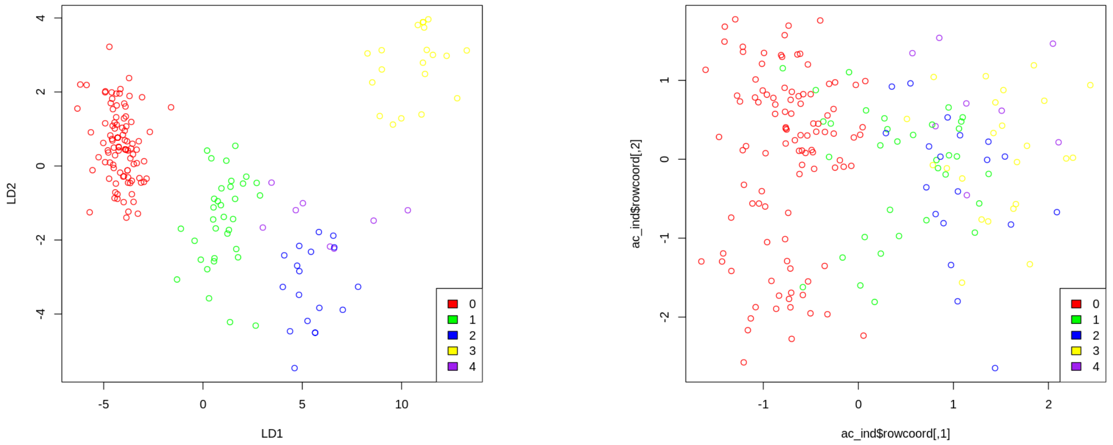
To compute the Linear Discriminant Analysis we have removed three problematic variables from the dataset. This variables are

- $V1$: the patient ID,
- $V57$: year of cardiac (which can be expressed as a linear combination of the other variables),
- $V59$: left main coronary trunk. It takes only two values. 1 is taken by 270 patients and 2 by only 12 patients. So it cannot be used to discriminate the data properly.

In Figure 4a) we can see the plot obtained after FDA. We can clearly distinguish different levels of the target variables (each one is represented by one colour). We also can note that the variables obtained seem to follow a multivariate normal distribution.

2.6.3 MCA

We have studied the relation between factor variables performing a Multivariate Correspondence Analysis. But as we can see in Figure 4*b*), it is not very helpful.



(a) FDA

(b) MCA

Figure 7: FDA/MCA plots

3 Initial models

We are going to use the following models:

- Regularized Discriminant Analysis
- k-Nearest Neighbour
- Naive Bayes
- Generalized Linear Models (Multinomial)

We cannot perform a Linear and Quadratic Discriminant Analysis because there are some groups with little representation and it gives numerical errors in QDA, and in LDA when doing cross-validation sometimes one group is misrepresented and numerical errors also occurs.

The target variable is an ordinal, and so one of our model will be ordinal regression. However if we treat the variable as a factor we could use the multinomial glm.

The formula used for the error is simply $1 - \frac{\text{Correctly classified observations}}{\text{total number of observations}}$

3.1 Cross-Validation

Validation error obtained for RDA is 1.05%, for Naive Bayes 11.3%. In Figure 5 we plot the validation error for different values of k for the k-NN. The minimum one is 20.98%, achieved for $k = 1$, much larger than the other ones.

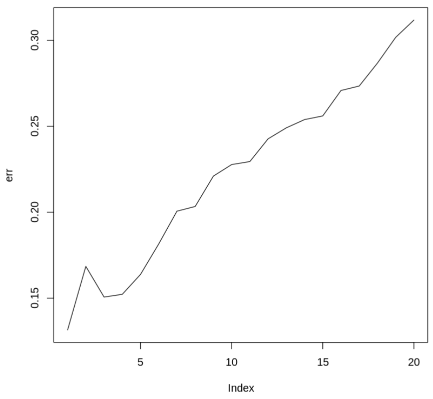


Figure 8: k-NN training error depending on the value of k

We have performed a Multinomial regression after computing the step function and we get a cross-validation error of 25.9%. Before applying the criteria of AIC to reduce variables the error was 38.5%. Without the FDA features the error is 40.6% before step and 26.07% after the step.

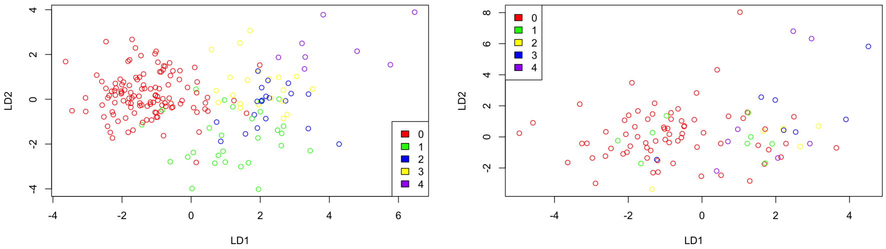
3.2 Test error

In light of the above, we have selected the RDA model as our final model. We have computed the error with the test data obtaining an error of 23.4%. We may note that the model is a little bit overfitted to the training data.

4 Hungarian dataset

We are going to continue the project using the hungarian dataset. It has 294 observations of the same variables as before. We find this dataset more interesting because many investigators have chosen the cleveland dataset instead of this one and so it is less worked, although the result are similar for both datasets.

Since the type of data is essentially the same as before, we are going to skip the preprocessing work because it is nearly the same we have done with cleveland dataset. The code is in the files Preprocessing.R and Visualizations.R. One thing we want to emphasise is the difference in the FDA projections between train and test. This imply that this features won't be useful, however we will keep them to see what happens when adjusting models to them.



(a) Training set

(b) Test set

Figure 9: FDA extracted features (2 components)

4.1 Initial models

We have used the same models as in section 2, obtaining the following cross-validation errors (using 10x10 cross-validation).

4.1.1 Regularized Discriminant Analysis

Validation error for Regularized Discriminant Analysis is 35%, but if we add the extracted variables from FDA we get 12.1%.

We have also tried to perform LDA and QDA without success since there are small groups in some factor variables (small in comparison with the whole training set), that is because there are variables with only 3 or 4 subject in one group and others with many levels like the variables that represent the year, we could have erased them, but we preferred to keep the maximum information possible.

4.1.2 Naive-Bayes

Using Naive-Bayes we get a validation error of 29%.

4.1.3 K-Nearest Neighbour

We have performed a cross-validation to obtain the best k for K-Nearest Neighbour and, as we can see in the image below, the optimum is reached for k=4 and with value 28 %.

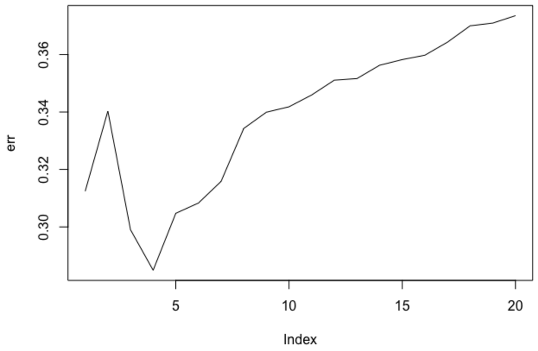


Figure 10: k-NN training error depending on the value of k

4.1.4 Multinomial

Finally, we have performed a Multinomial regression obtaining a cross-validation error of 52%, which turns out to be 48% after applying the step function. However, if we use the FDA components we get a 47% validation error and after applying the step routine to reduce variables it lowers to 29%.

4.1.5 Neural Networks

The main problem with neural networks here is the time it requires to do cross-validation. Since we have 32 variables, many of which are factor which are one-hot encoded, the number of weights get high with very few neurons. In fact, the limit of weights by default of the nnet function gets achieved with only 9 hidden neurons. To tackle this problem we tried two different approaches which didn't end well, so finally we just decided to do 1x5 cross-validation instead of 10x10.

The first approach we tried was to work with the PCA components instead of all the components. But it resulted that we only had 16 numeric variables so with the 16 components there wasn't enough information to train a classifier because the information was in the factors.

The second approach was to use only the FDA components, it achieved low validation error, but the training error was incredibly low so we suspected it was overfitting. Moreover, since the components of FDA were so mixed in the test set we concluded that the information of this components was misleading.

Finally with all the data we tried to train a neural network with many coefficients and some decay, following the advice of our teacher. Nine neurons were enough to achieve 0 training error on the whole dataset (now we are not taking into account any extracted features, just the dataset). We chose 20 neurons because it seemed high enough to fit well the data and it wasn't extremely expensive (computationally) to tune the hyperparameter of the regularization.

The result was that the best lambda was near 0.6, using this value we achieved a validation accuracy of 64%. Note this accuracy is measured according to the probabilities, not the predicted values, this can mean many things, either the net is fitting poorly or it isn't quite sure of its predictions. The training error, measured with the predictions, was of 22%. It seems high, but taking into account that the test error on cleveland with the best model was bigger than that it could mean it is well adjusted.

Since all of the previous work didn't give a very good result, we decided to slightly modify to problem in order to compare to previous results on this dataset.

4.1.6 Simplifying the problem

We have converted the target variable to a binary one in order to obtain a simple model. Now the target variable is 0 if the patient does not have a heart disease and 1 otherwise.

With one hidden layer of 30 neurons we already achieve a 0 % training error, so we tried to fit the decay with this configuration. After several computations of 10 fold cross-validation we obtained that the optimum decay was 0.55, obtaining 80 % cross-validation accuracy and 11 % training error.

5 Conclusions

In general, we have been unable to achieve a small validation error. Moreover, the experts that had been working with hungarian dataset achieved around a 77 % of accuracy. So we suspect that the Bayes error is big enough that we can not expect to obtain better results.

As expected, we obtain lower validation errors with neural networks than with initial models. So we decided to use this methods to compute an estimation of the test error.

As mentioned before, the best model we got was a neural network with one hidden layer of 20 neurons and a regularization parameter of 0.68, so we will be using this one on the test set. The result is a test error of 29%.

On the other hand, we used a neural network with one hidden layer of 30 neurons and a regularization parameter of 0.55 for the binarized problem (only 2 target classes) Using the test set to validate the model we get around 17 % test error.

There is an interesting question, is it better to join the groups before or after the training of the net? It seems reasonable to think that if the net has access to more information it will do better. So we used the same predictions as with the first net and classified all the predictions that were greater than 1 to 1. This way the test error is 14%. We could state that our hypothesis is true, but this result could be just random so we will compute some confidence interval to see if it really is smaller (using De Moivre-Laplace theorem). The interval with a confidence of 5% is (0.10,0.25) for joining levels before, and (0.07,0.21) for joining them after. Therefore, we can accept that both test errors are statistically equal.

References

[1] UCI Machine Learning Repository
`http://archive.ics.uci.edu/ml/datasets/Heart+Disease`