01RLPOV - Data Spaces

Indian Liver Patient dataset analysis

Enrico Agrippino

April 2021

Contents

1 Data exploration

1.1 Introduction to dataset

The aim of this "tesina" is to analyze a dataset of 416 Indian liver patients and 167 non liver patients[1]. The data set was collected from north east of *Andhra Pradesh*, India. For each row of the dataset there are 11 attributes listed below:

- *Age*: Age of the patient, patients over 90 are all set to 90 (Numerical)
- *Gender*: Gender of the patient (Categorical)
- *TB*: Total Bilirubin (Numerical)
- *DB*: Direct Bilirubin (Numerical)
- *Alkphos*: Alkaline Phosphotase (Numerical)
- *Sgpt*: Alamine Aminotransferase (Numerical)
- *Sgot*: Aspartate Aminotransferase (Numerical)
- *TP*: Total Protiens (Numerical)
- *ALB*: Albumin (Numerical)
- *A/G Ratio*: Albumin and Globulin Ratio (Numerical)
- *Class*: Class 1 corresponds to liver patient while Class 2 to not-liver patient (Categorical, target variable)

The objective of this work is therefore to classify a person in liver/not-liver patient given the ten predictive variables listed above.

1.2 Data exploration

Since the vast majority of attributes are numerical, we can take a look at their distributions with boxplots. In Figure 1 six among the most significant boxplots are shown.

[1] Dataset avaible on UCI Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/ILPD+%28Indian+Liver+Patient+Dataset%29

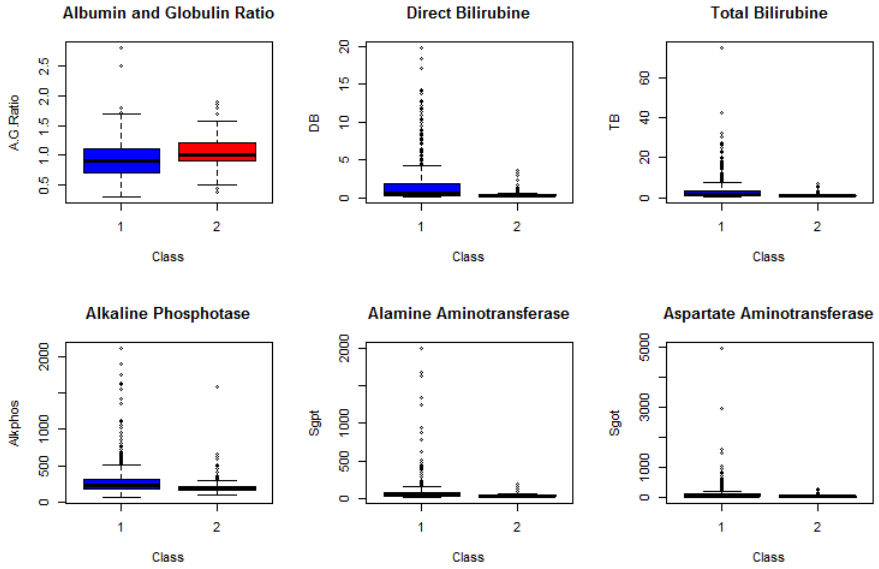

Figure 1: Boxplots for A.R. Ratio, DB, TB, Alkphos, Sgpt, Sgot

Looking at the figure some outliers can be spotted, in particular there are some extremely high values. However, not being a medical expert, I preferred to keep all values since some important information could have been lost.

Moreover, we can notice that the two classes are not so well separated. We can consequently expect our classification task to be challenging.

Let us now analyze the correlation among variables. There are some significant remarks to be done. As one could have expected, the correlation between Direct and Total Birilubine is very high, and that is not the only case, as can be seen from correlation matrix in Figure 2.

The number of predictors is not particularly high, however in order to avoid the so-called *Curse of dimensionality*, it could be better to work with less variables. One of the most frequently used technique to reduce number of predictors is the Principal Component Analysis (PCA). Neverthless, given the nature of data, interpretability of results has a great role in this case, so I decided not to apply PCA. High correlations among some predictors give us the opportunity of eliminating some variables, improving accuracy and interpretability of the results. I did not make a unique choice about which predictors to eliminate, it will depend on what method I will be using. In the next section more details are given about each method.

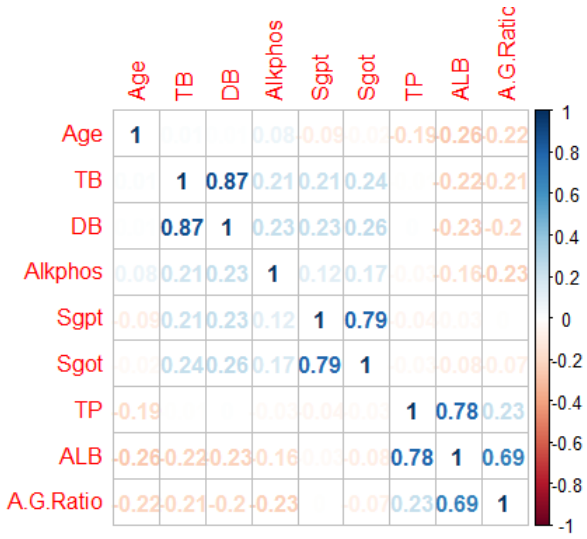

Figure 2: Correlation Matrix

1.2.1 Missing data

In the dataset there are only 4 missing data, all relative to the attribute *A.G.ratio*. Since the missing data are only the 0.68%, I simply decided to eliminate those 4 rows of dataset.

1.2.2 Splitting data

To test the results, I will apply the "Test on test data" method, so I split the original data with stratifying sample, i.e. keeping the proportion of the two class in the original data set. I obtained this partition:

- Training Set: 75% (429)
- Test Set: 25% (150)

Test data will be used only for the final testing, so from now on I will work on Training Data.

The two classes are not perfectly balanced: on 429 training data only 122 are of Class 2 (28.4%). In order to improve the results I decided to do a simple oversampling of *Class 2* data so that the two classes were perfectly balanced, for a total of 614 training data.

2 Classification methods

2.1 KNN: K-Nearest Neighbors

K-nearest neighbors (KNN) classifier can be considered the simplest classification method. Despite that, it often gives pretty good results. The idea is to assign a test observation to the class of the majority of its K neighbors in the training dataset. To define the concept of neighborhood in predictors space we have to specify a distance, in our case the *Euclidean* one. Given a positive K and a test observation x_0, the KNN classifier first identifies the K points in the training set that are closest to x_0, represented by $\mathcal{N}_0$. It then estimates the conditional probability for class j as the fraction of points in $\mathcal{N}_0$ whose response values equal j. Finally, KNN classifies the test observation x_0 to the class with the largest probability. In our case we have only two classes, so KNN classifies the test observation to the class with the probability > 0.5.

The crucial point of KNN is choosing the parameter K. When K is low KNN classifier has low bias but very high variance. This happens because the test observation x_0 is influenced only by its closest neighbors in the training dataset. As K grows, the method becomes less flexible because x_0 will be influenced by more neighbors. This corresponds to a low-variance but high-bias classifier. Therefore we have to deal with bias-variance trade off in order to choose the best value for K.

2.1.1 Confusion matrix and Accuracy

Accuracy index is the ratio between correctly predicted observation and the total observations. Accuracy is the most intuitive performance measure, but it is accurate only in case of symmetric datasets, i.e. where values of false positive and false negatives are almost same. For these reasons, there exist other indexes to evaluate performances, as shown in the so-called *Confusion Matrix* in the following Figure (3).

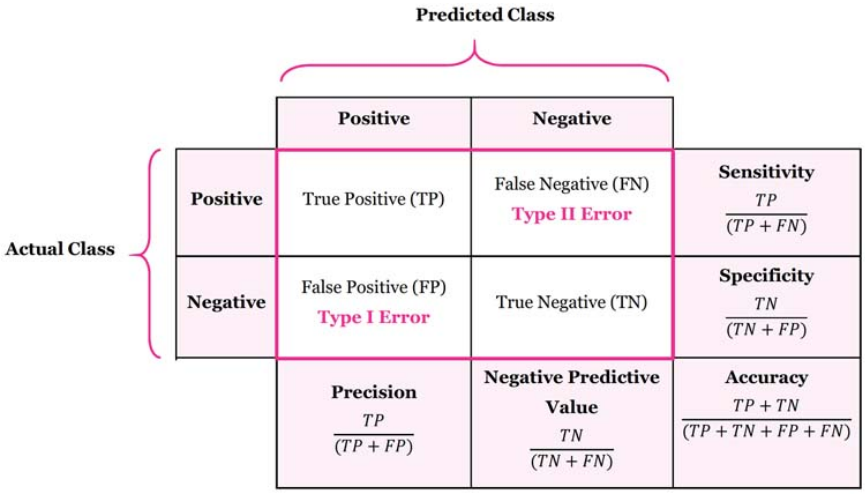

Figure 3: Confusion Matrix with index definitions

2.1.2 The choice of K

I used Accuracy index to determine which value of K is the best in our case. I computed the values of Accuracy for K going from 1 to 60, using all the predictors to build the model. The results are shown in Figure 4. As can be seen, the optimal value of Accuracy would correspond to $K = 1$, but Negative predictive Value is unaccettable in this case (only 2 test data were correctly assigned to Class 2 over the actual 43). I therefore decided to use $K = 36$, that led to an Accuracy = 0.653 and accettable Precision and NPV.

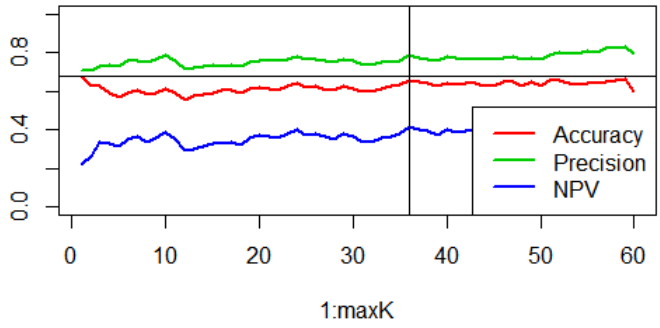

Figure 4: Plot representing Accuracy, Precision and Negative Predictive Value for different values of K. The horizontal line corresponds to the maximum value for accuracy, obtained for K=1. The vertical line corresponds to K=36, my choice.

Figure 5 shows the Confusion Matrix for KNN method with $K = 36$.

```
                pred_class
real_class  1  2
         1 77 30
         2 23 20
```

Figure 5: Confusion Matrix for KNN with $K = 36$

2.2 Logistic Regression

The next method used is the so-called *Logistic Regression*. In a nutshell, given our predictors (attributes) X, Logistic Regression assumes a linear relationship between X and the logit of the probability $p(X)$ that an observation belongs to class 2 (not-liver patient). Namely:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta X$$

where β_0 is the intercept and β is a vector of coefficients, one for each predictor.

I first tried to use all the attributes to estimate the model. That gave the results of an accuracy index equal to 0.693 and the coefficients reported in the Figure 6.

```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.9007493  1.3425764   4.395 1.11e-05 ***
ALB          3.1201530  0.7902429   3.948 7.87e-05 ***
A.G.Ratio   -4.1664338  1.2411637  -3.357 0.000788 ***
TP          -1.4998702  0.3910433  -3.836 0.000125 ***
TB          -0.1570279  0.4275873  -0.367 0.713439
Sgpt        -0.0029846  0.0035862  -0.832 0.405261
Age         -0.0153246  0.0059942  -2.557 0.010570 *
Alkphos     -0.0017713  0.0007861  -2.253 0.024231 *
DB          -0.5259129  0.7498178  -0.701 0.483061
GenderMale  -0.0207107  0.2111882  -0.098 0.921879
Sgot        -0.0063538  0.0028763  -2.209 0.027171 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 6: Coefficients of Logistic Regression model with all variables, R output

As we can seen from the R output, there are some coefficients that are not significant. That can happen because an attribute is poorly correlate with the target attribute class or because it is highly correlated with other attributes. Looking back at Figure 2, we can see that there are attributes highly correlated among each other. For these reasons, I decided not to use attributes *DB, Sgpt* and *Gender*. Doing so, I obtained the coefficients shown in Figure 7

Now the coefficients are all almost significant. Despite the elimination of 3 attributes over 10, accuracy has not changed! In fact, it remains equal to 0.6933.

Confusion matrix for this choice of predictors can be seen in Figure 8.

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.7188711  1.2946049   4.417 9.99e-06 ***
ALB          3.0117900  0.7643593   3.940 8.14e-05 ***
A.G.Ratio   -4.0231710  1.2040537  -3.341 0.000834 ***
TP          -1.4566836  0.3805234  -3.828 0.000129 ***
Age         -0.0150160  0.0059479  -2.525 0.011584 *
Alkphos     -0.0018747  0.0007882  -2.378 0.017385 *
DB          -0.8214453  0.2110848  -3.892 9.96e-05 ***
Sgot        -0.0079223  0.0022549  -3.513 0.000443 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 7: Coefficients of Logistic Regression model without DB, Sgpt and Gender attributes, R output

```
          pred_class
real_class  1  2
         1 67 40
         2  9 34
```

Figure 8: Confusion matrix for Logistic Regression model without DB, Sgpt and Gender attributes, R output

2.2.1 ROC Curve

In Figure 9 ROC Curve of the last model descripted is shown. ROC curve is a graphic for simultaneously displaying the two types of errors: True Positive Rate= $\frac{TP}{TP+FN}$ and False Positive Rate = $\frac{FP}{FP+TN}$, for all possible thresholds. The overall performance of a classifier, summarized over all possible thresholds, is given by the area under the curve (AUC). An ideal ROC curve will hug the top left corner, so the larger the AUC the better the classifier (an AUC below 0.5 means that the model is performing worse than hazard predictions). Here AUC is equal to 0.78.

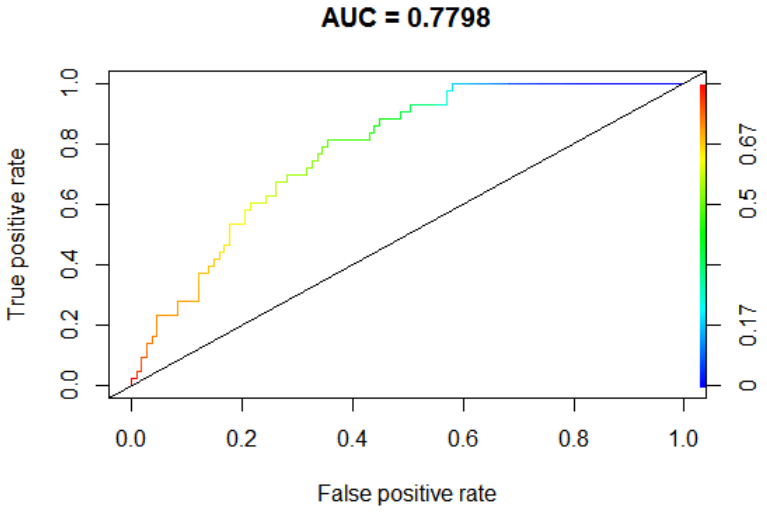

Figure 9: ROC Curve for Logistic Regression model without DB, Sgpt and Gender attributes. Above is reported the value of AUC. R plot

2.3 Support Vector Machine

SVM or Support Vector Machine is a linear model for classification and regression problems. Here we use it as a classification method. The idea behind SVM is simple: at first approximation, the algorithm creates in the predictor space a hyperplane which separates data into classes. Since data are not always linearly separable, a natural relaxation of the algorithm is to allow the constraints to be violated for some of the examples in the data set, associating a cost to these violations (coefficient *Cost*) . This is called "Soft-SVM".

The method very briefly descripted above assumes that the boundary between the two classes is linear. What if we are faced with non-linear class boundary? It turns out that the SVM solution involves only the inner products of the observations. The inner product of two observations $x_i, x_{i'}$, is given by

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{1}$$

where p is the number of features. Therefore we can replace (1) with $K(x_i, x_{i'})$, a generalization of the inner product where K is some function called *Kernel*. If we take

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

we are in the aforementioned situation of a *linear* Kernel.
While the following choice for K

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right) \tag{2}$$

is called *radial kernel*. Notice that here we introduced a hyperparameter γ.

I implemented linear and radial SVM on my data set. In (2) I fixed $\gamma = 1$. In both cases the tuning for *Cost* coefficients for the violations gave me the value of *Cost*$= 1$.
Moreover, after some tries I decided not to use the attributes TB, Sgot, Alb for the SVM computation, improving a little the results. In fact, these are three among the most correlated variables as one can notice from the Correlation Matrix in Figure 2.

In Figure 10 the Confusion Matrices for the two SVM methods and their Roc Curves are shown. Which of the two is the best? Accuracy for Linear SVM is 0.607, while for Radial is better: 0.693. Linear kernel has a very high precision for Class 1: 96%, on the other hand has a bad Sensitivity, only 46%. The opposite happens if we are interested in Class 2. Confusion matrix for Radial kernel is a more balanced one, but also this method fail to predict test observations of Class 2. Indeed, only 24 are correctly assigned to Class 2, leading to a Specificity of 56% and a Negative Predictive Value of 47%[2]. Saying that, I am note suggesting that Radial kernel performs better than Linear kernel: confusion matrices are obtained assuming a threshold probability of 0.5, but one could choose a different threshold. It depends on what we are more interested in predicting: liver or non-liver patients?

A better comparison between the two methods can be done looking at the ROC curves in Figure 11. Here the AUC index suggests that Linear SVM performs better than Radial.

```
> table(real,pred_Lin)
    pred_Lin
real  1  2
   1 50 57
   2  2 41
> table(real,pred_Rad)
    pred_Rad
real  1  2
   1 80 27
   2 19 24
```

Figure 10: Confusion matrices of Linear and Radial SVM Classification methods applied to the dataset without TB, Sgot and Alb features. R output

[2]Here I used the definition of indexes given in Figure 3

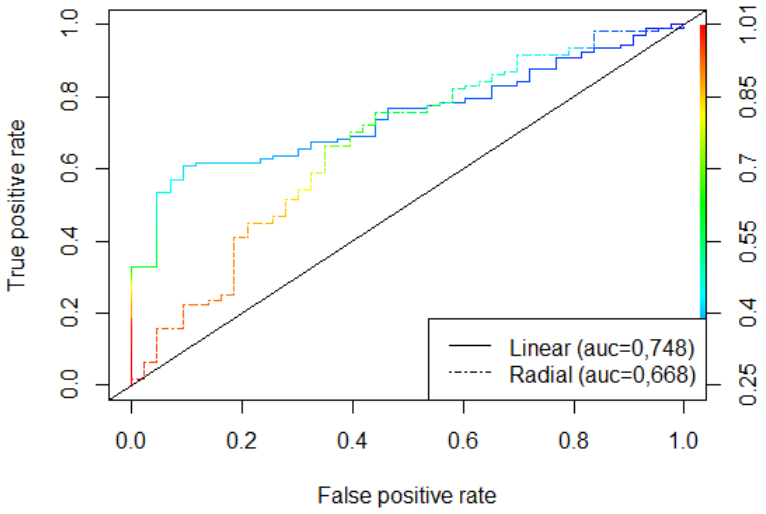

Figure 11: ROC Curve (setting Class 1 as target class) for Linear and Radial SVM Classification methods applied to the dataset without TB, Sgot and Alb features. R plot

2.4 Tree-based methods

Tree-based methods for classification involve finding the best partition of the predictor space with respect to some loss measure. In order to make a prediction for a given observation, we assign it to the most commonly occurring class in the region to which it belongs. Since the set of splitting rules to segment the prediction space can be summarized in a tree, this approach is called *Decision tree*. The loss measure aforementioned is theoretically the *classification error rate*: the fraction of observation in a given region that are misclassified. However, it turns out that classification error is not sufficiently sensitive, and in practice two other measures are preferable: *Gini index* and *Cross-Entropy*.

One of the greatest advantages of Decision-tree is then its interpretability. In Figure 12 the classification tree for our training data is shown. Here I setted the parameter "minSplit" equals to 10, i.e. no splitting takes place if less than 10 training observations belongs to that region.

Let us see now the Confusion Matrix for the test set in Figure 13. The Accuracy index is pretty low: 0.62. In particular, tree method fails to classify Class 2: for this class Precision is only 0.38 and recall 0.51.

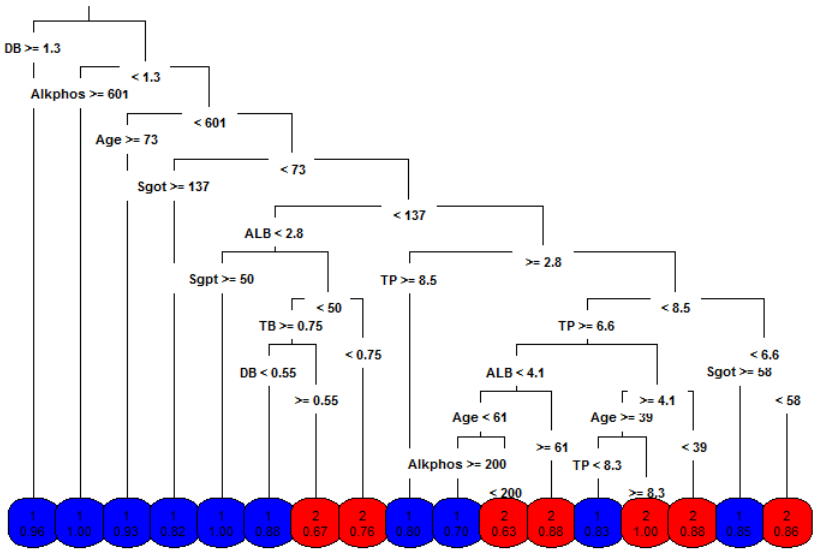

Figure 12: Classification tree for training data. Blue is the color for Class 1, red for Class 2. In the leaves the probability of the fitted class can be read. R plot

```
           pred_class
real_class  1  2
         1 71 36
         2 21 22
```

Figure 13: Confusion matrix for Tree classification method on test set. R output.

2.4.1 Random Forest

One of the main disadvantages of trees is that they can be very non-robust: a small change in data can cause a large change in the final estimated tree. In other words they have too much variability. Random forest idea comes from the consideration that *averaging a set of observation reduces variance*. We could calculate B separate trees using B training sets and then assign the overall prediction for a given observation to the most commonly occurring class among the B predictions. How to obtain B different training sets? Using *Bootstrap* technique, i.e. sampling with replacement from the original data set B training sets. Moreover, when building a decision tree, each time a split in a tree is considered only a random selection of m predictors from the full set of p predictors is chosen as split candidates. Then at the next split, a fresh selection of m predictors is taken. This helps to decorrelates the trees. If we set $m = p$, so that decision trees are built as usually, the method is called *Bagging*. A typical choice for m is $\sqrt{p}$.

Here I tuned the parameter m: results can be seen in Figure 14. I chose

accuracy as target value for deciding which value of m was the best. I obtained $m = 3$ with an accuracy index equals to 0.72. However, as one can notice from the plot, there is not such an high difference among different values of m.

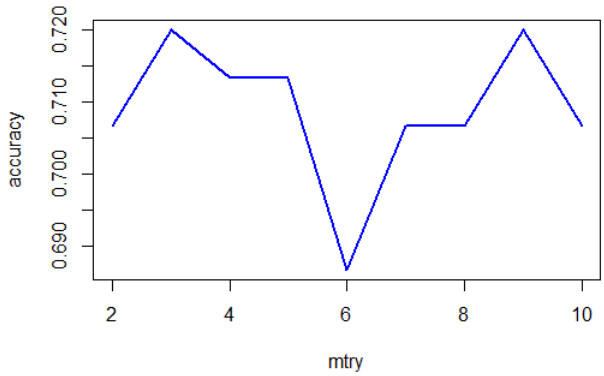

Figure 14: R plot for tuning the best value of the number of predictors among which tree can choose a each split. The highest value for Accuracy is reached for $m = 3$

The confusion matrix for testing set is displayed on Figure 15.

```
          pred_class
real_class  1  2
         1 87 20
         2 22 21
```

Figure 15: Confusion matrix of test set for Random Forest method.
Accuracy= 0.72, Precision= 0.798, Recall= 0.813, Specificity= 0.48, Negative Predictive Value= 0.51.

Given its construction, Random forest can not be visualized easily like Decision trees. However, we can view (Figure 16) the importance of each predictor. *Mean Decrease Accuracy* is computed from permuting OOB data: for each tree, the prediction error on the out-of-bag portion of the data is recorded. Then the same is done after permuting each predictor variable. The difference between the two are then averaged over all trees, and normalized by the standard deviation of the differences. In Figure 17 the values of Mean Decrease accuracy are displayed for each variable.

```
     Sgot   Alkphos      Sgpt       Age        TB       ALB        DB        TP A.G.Ratio    Gender
 51.41226  48.58013  46.64558  41.60045  40.16114  39.67223  37.90642  37.51206  30.92947  20.00980
```

Figure 17: Mean Decrease accuracy values for Random Forest, R output

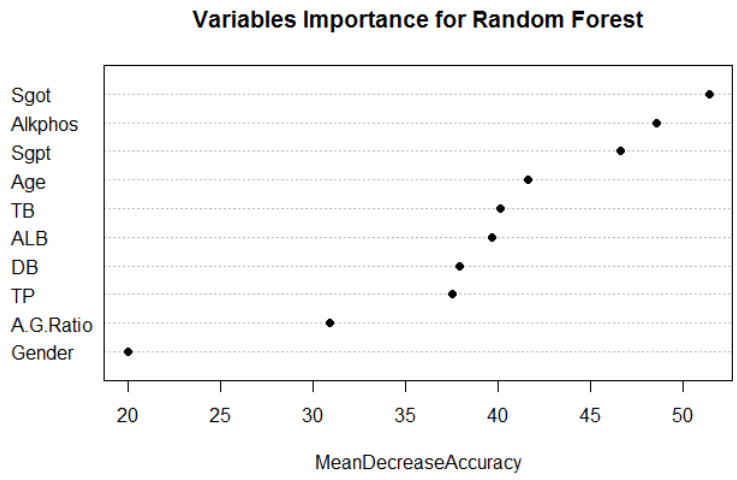

Figure 16: R Plot of Mean Decrease Accuracy

2.4.2 Tree vs Random Forest

Tree has the advantage of being easily visualized and interpreted. On the other hand, its level of predictive accuracy do not reach other classification approaches such as Logistic regression or SVM. Instead Random forest, despite its impossibility to be visualized, overperforms decision trees and yet some consideration about feature importance are possible as said in the paragraph above.

In order to give a better comparison between the two methods, in Figure 18 ROC curves are shown. As can be seen, AUC for Random forest is higher than Tree: that confirms our previsions about Random Forest improving accuracy of predictions.

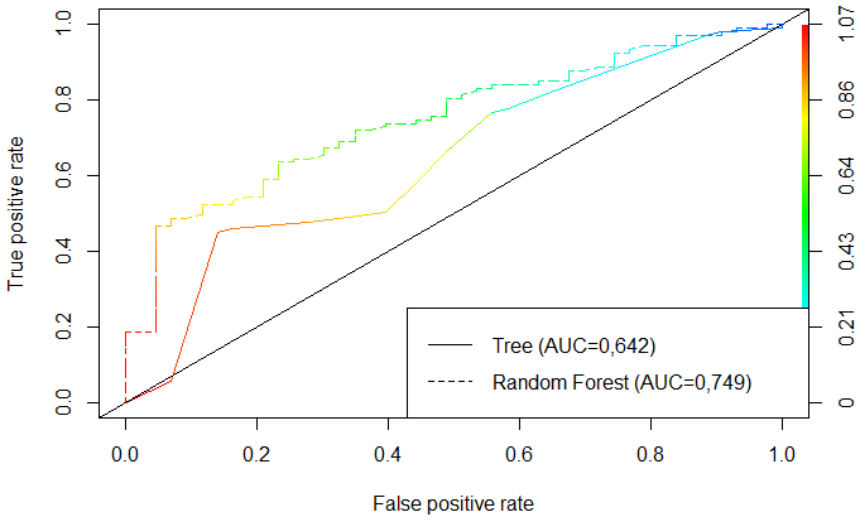

Figure 18: ROC Curves for Decision tree (plain line) and Random Forest (dotted line), where Class 1 is set to be the target class. R plot

3 Comments about the overall results

As said in section 1.2 at the beginning of this work, classifying instances of the original dataset into Liver/Not-liver patient revealed to be a challenging task. Results shown in previous pages suggest that not all methods have acceptable performances. The best learning methods have proven to be Logistic Regression (Accuracy= 0.693, AUC= 0.78) and Random Forest (Accuracy= 0.72, AUC= 0.749). Both methods are pretty easily interpretable: in Logistic Regression, Coefficients displayed in Figure 7 give a precise measure of how and how much each predictor influences the assignation of a given observation to the Class, while in Random Forest we can see the predictors importance as shown in Figure 16.