Heart failure classification

Luca Bartolomei

10 aprile, 2022

Contents

1 Introduction

This project has the aim to analyze the Heart failure dataset to build a classifiers to predict whether people have heart disease or not.

The dataset used for project comes from Kaggle website: https://www.kaggle.com/datasets/andrewmvd/heart-failure-clinical-data

Attribute Information:

1. age: The person's age in years
2. anaemia: Decrease of red blood cells or hemoglobin
3. creatinine_phosphokinase: Level of the CPK enzyme in the blood (mcg/L)
4. diabetes: If the patient has diabetes
5. ejection_fraction: Percentage of blood leaving the heart at each contraction
6. high_blood_pressure: If the patient has hypertension
7. platelets: Platelets in the blood (kiloplatelets/mL)
8. serum_creatinine: Level of serum creatinine in the blood (mg/dL)
9. serum_sodium: Level of serum sodium in the blood (mEq/L)
10. sex: The person's sex
11. smoking: If person smoking
12. target: Death
    - 0 = no
    - 1 = yes

In this project "target" attribute is going to be the dependent variable. The others attributes are going to be the predictors.

This project will make a comparison between different machine learning algorithms in order to to assess the correctness in ***classifying*** data with respect to efficiency of each algorithm in terms of ***sensitivity*** and ***specificity***.

Sensitivity and ***specificity*** are widely used in medicine for binary classification test.

We are going to project a ***Machine Learning Classifiers***, more specifically a ***Binary classifiers*** where the two classes are:

1. yes: Death
2. no: No Death

2 Methods

2.1 Key concepts

Classification Classification is the process of predicting the class of given data points. When there are only two classes the problem is known as ***statistical binary classification***.

Accuracy and ***Kappa*** These are the default metrics used to evaluate algorithms on binary and multi-class classification datasets in caret.

Accuracy is the percentage of correctly classifies instances out of all instances. It is more useful on a binary classification than multi-class classification problems because it can be less clear exactly how the accuracy breaks down across those classes.

Sensitivity is the true positive rate also called the recall. It is the number instances from the positive (first) class that actually predicted correctly.

Specificity is also called the true negative rate. Is the number of instances from the negative class (second) class that were actually predicted correctly

Confusion Matrix Confusion Matrix is represented by the following table

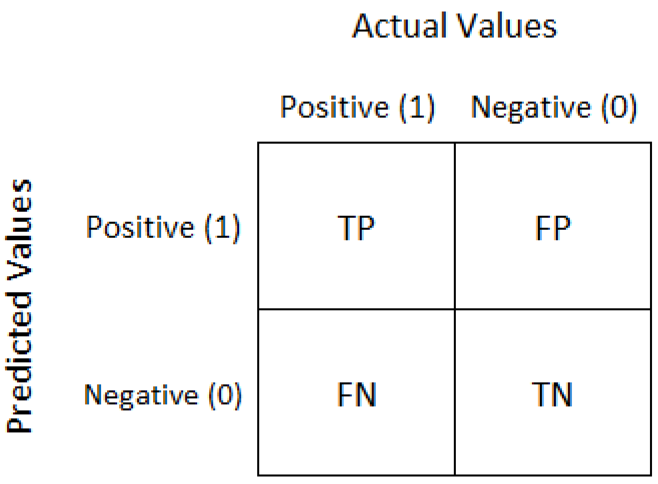


Figure 1: Confusion matrix

1. TP: True positive
2. FP: False positive
3. FN: False negative
4. TN: True negative

A true positive is an outcome where the model correctly predicts the positive class.

A false positive is an outcome where the model incorrectly predicts the positive class.

A false negative is an outcome where the model incorrectly predicts the negative class.

A true negative is an outcome where the model correctly predicts the negative class.

Now we consider these rates

$$\text{TPR} = \frac{TP}{TP + FN}$$

$$\text{FPR} = \frac{FP}{TN + FP}$$

TPR (True Positive Rate, that correspond to Sensitivity) means how often does the classifier correctly predict positive.

FPR (False Positive Rate) means how often does the classifier incorrectly predict positive. FPR also correspond to: 100% - Specificity, where Specificity is the proportion of true negative who are correctly identified.

ROC (Receiver Operating Characteristic) Curve is a way to visualize the performance of a binary classifier.

The ROC metric is strictly connected with Confusion Matrix

The ROC metric is better explained by ROC curve where TPR is on y axis and TFR on x axis.

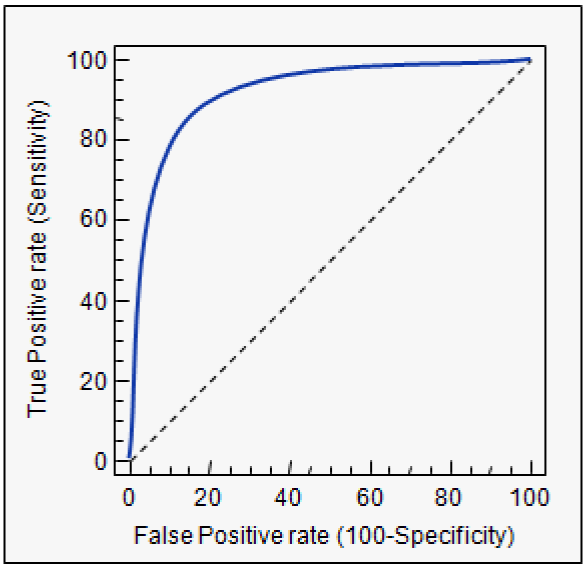


Figure 2: ROC curve

The dotted line correspond to the random classification, so a good classification is showed by any curve (like the blue one) up the dotted line.

The area under the ROC curve is also called: ***AUC*** (Area Under Curve)

Covariance of two statistical variables is a number that provides a measure of how much the two vary together.

Covariance matrix is a generalization of ***covariance*** to the case of dimension greater than two.

Homoscedasticity Homoscedasticity describes a situation in which the error term is the same across all values of the independent variables.

Normal distribution The normal distribution, also known as the bell curve and as the Gaussian distribution, is one of the most famous mathematical concepts in history. A reason for this is that approximately normal distributions occur in many situations, including gambling winnings, heights, weights, blood pressure, standardized test scores, and experimental measurement errors.

The fact that the distribution is defined by just two parameters implies that if a dataset is approximated by a normal distribution, all the information needed to describe the distribution can be encoded in just two numbers: the average and the standard deviation. We now define these values for an arbitrary list of numbers.

Shapiro- Wilk test The Shapiro–Wilk test is a test of normality in frequentist statistics. It was published in 1965 by Samuel Sanford Shapiro and Martin Wilk. The null-hypothesis of this test is that the population is normally distributed. Thus, if the p value is less than the chosen alpha level, then the null hypothesis is rejected and there is evidence that the data tested are not normally distributed.

Supervised learning is a machine learning technique that aims to train a computer system in order to allow it to autonomously make predictions on the output values of a system versus an input based on a series of ideal examples, consisting of pairs input and output, which are initially provided.

Unsupervised learning is a type of machine learning that looks for previously undetected patterns in a dataset with no pre-existing labels and with a minimum of human supervision. Unlike supervised learning which usually makes use of human-labeled data, unsupervised learning allows for the modeling of probability densities on inputs.

The Bayes theorem describes the probability of an event, based on the preliminary knowledge of the conditions that could be related to the event. It serves as a way to understand ***conditional probability***.

The conditional probability of an event A with respect to an event B is the probability that A will occur, knowing that B has occurred. This probability, indicated $P(A|B)$ or $P_B(A)$, expresses a "correction" of expectations for A, dictated by observation of B.

Naive Bayes is a ***supervised learning algorithm*** suitable for solving binary (two-class) and multi-class classification problems. The main peculiarity of the algorithm, in addition to making use of ***the Bayes theorem***, is that it is based on the fact that all the characteristics are not related to each other.

Ensemble model is a combination of single simple models (called weak learners) that together create a new, more powerful model (called strong learner).

Boosting is an ***ensemble method*** for improving model predictions of a given learning algorithm. The idea behind it is to train weak learners sequentially, each trying to correct its predecessor.

Loss function is a method for estimating the quality of an algorithm in modeling the data provided. If the forecast deviates too much from actual results, the loss function produces a very large number.

Bagging is an ***ensemble model***. Is a general procedure that can be used to reduce the variance of those algorithms that have a high variance.

Collinearity is a linear association between two predictors. ***Multicollinearity*** is a situation where two or more predictors are highly linearly related that it can lead to misleading results when attempting to predict the dependent variable.

2.2 Model evaluation

2.2.1 Metrics To Evaluate Machine Learning Algorithms

The metric used for these project is ROC.

AUC is the area under the ROC curve and represents a models ability to discriminate between positive and negative classes. An area of 1.0 represents a model that made all predicts perfectly. An area of 0.5 represents a model as good as random.

ROC can be broken down into sensitivity and specificity. A binary classification problem is really a trade-off between sensitivity and specificity.

To calculate ROC information, in our trainControl we must set the summaryFunction
to twoClassSummary.

2.2.2 Principal component analysis

Principal component analysis (PCA) is a technique that, starting from a set of numerical variables, obtains a smaller set of “artificial” orthogonal variables. The reduced set of linear orthogonal projections (known as “principal components” or “principal components”, “PC”) is obtained by linearly combining the original variables in an appropriate manner.

In PCA, the term “information” indicates the total variability of the original input variables, ie the sum of the variances of the original variables. The central point of the PCA is the so-called spectral decomposition (also called the decomposition into eigenvalues and eigenvectors, or eigendecomposition) of the sample variance/covariance matrix. This decomposition returns the eigenvalues and eigenvalues of the ***covariance matrix***. The eigenvalues (in decreasing order of value) represent the amount of the total variability observed on the original variables, explained by each main component; the eigenvectors instead represent the corresponding (orthogonal) directions of maximum variability extracted from the principal components.

The hope in the application of the PCA is that the sample variances of the first Main Components (PC) are large, while the variances of the other components are small enough to consider the corresponding PCs negligible. A principal component variable that has little variability (relative to other variables) can be treated roughly as a constant. Omitting PCs with low sample variability and placing all attention on PCs with higher variance can be seen as a simple and “sensible” way to reduce the dimensionality (number of columns) of the dataset.

2.2.3 Adaptive Boosting

AdaBoost, acronym for “Adaptive Boosting” is a ***supervised learning algorithm*** proposed by Freund and Schapire in 1996. It was the first highly successful ***boosting*** algorithm developed for ***binary classification***.

It represents a popular boosting technique that combines multiple “weak classifiers” into one “strong classifier”.

2.2.4 Support Vector Networks

Support Vector Networks or SVM are classification algorithms used in supervised learning to analyze labeled training data. SVM can classify features in a training set into categories that use either a linear or non-linear model. The linearity of the classifier is determined by the kernel function of the data set. For non-linear classification in SVM using the kernel trick.

2.2.5 Stochastic Gradient Boosting

Stochastic Gradient Boosting is another ***supervised learning algorithm*** for regression and classification problems.

It represents an ***ensemble model***. In each training cycle, or iteration, the weak learner is built and its predictions are compared with the correct result we expect.

The distance between observation and prediction represents the error rate of our model. These errors are defined using a ***loss function***.

The purpose of the algorithm is to minimize this ***loss function***, using a tool that is called "***gradient***", which basically represents the partial derivative of the ***loss function***.

2.2.6 Classification and Regression Trees

Classification Trees is another ***supervised learning algorithm*** for regression and classification problems. Is based on concept of ***decision tree***.

A tree is a collection of entities called nodes connected to each other by arrows or lines. Each node contains a value and may or may not have child nodes, while arrows indicate the decisions/rules a tree may or may not make.

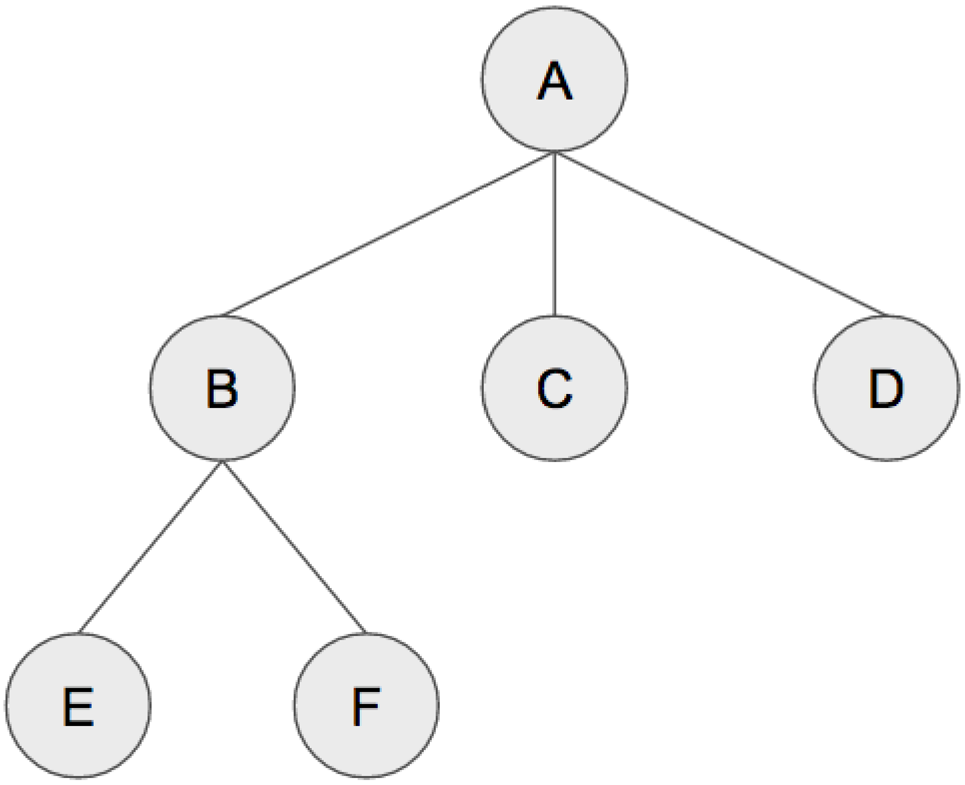


Figure 3: Decision tree

The nodes are A, B, C, D, E and F and are connected by lines that indicate kinship relations between the various nodes.

Node A is called the root node and is the starting point of the tree. It consists of three children: node B, node C and node D. The only node without a parent is the root node.

A tree is a ***binary tree*** when each parent has at most 2 child nodes. Decision trees are ***binary trees***.

An example of a binary tree is shown in the following image.

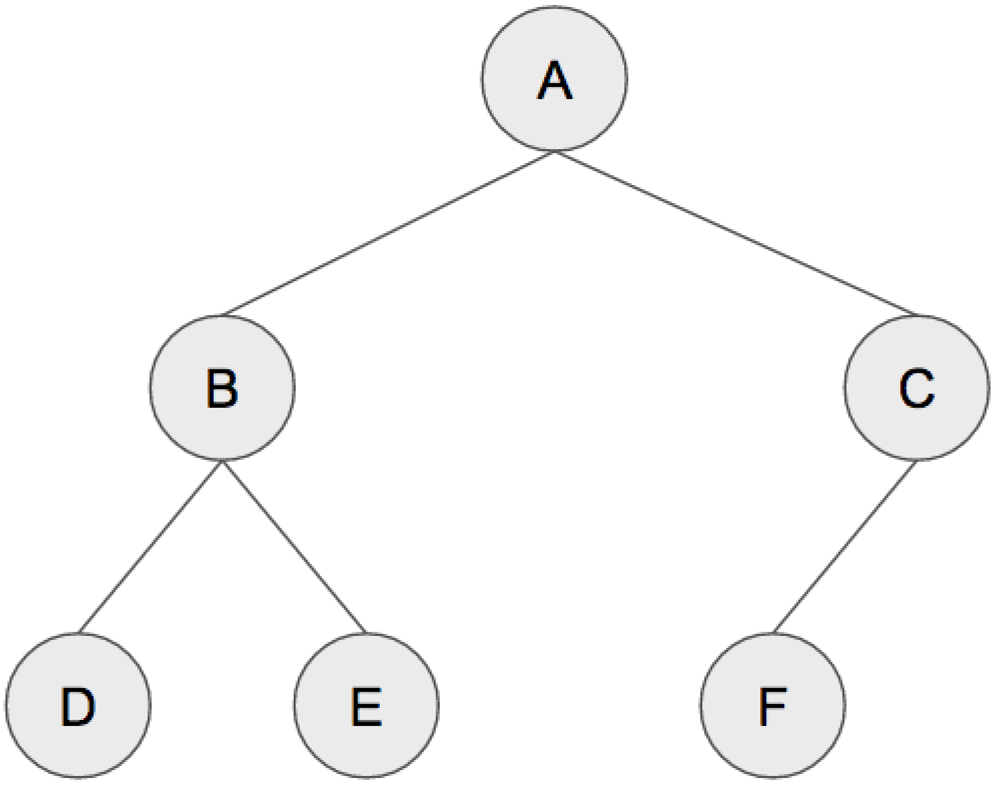


Figure 4: Binary tree

Classification and Regression Trees, also called CART, it is basically a binary tree.

2.2.7 Random Forest

Random Forest is another ***supervised learning algorithm***

It represents a type of ***ensemble model***, which uses ***bagging*** as an ensemble method and the decision tree as an individual model.

This means that a random forest combines many decision trees into one model. Individually, the predictions made by the decision trees may not be accurate, but combined together, the predictions will on average be closer to the outcome.

2.2.8 K Nearest Neighbor (KNN)

KNN is a ***supervised learning algorithm***, whose purpose is to predict a new instance by knowing the data points that are separated into different classes.

Its operation is based on the similarity of characteristics: the closer an instance is to a data point, the more the knn will consider them similar.

Similarity is usually calculated by Euclidean distance. The shorter the distance, the greater the similarity between the data point and the instance to be predicted.

In addition to the distance, the algorithm provides for setting a parameter k, chosen arbitrarily, which identifies the number of closest data points. The algorithm evaluates the k minimum distances thus obtained. The class that obtains the greatest number of these distances is chosen as the prediction.

2.2.9 Neural Network

A neural network is a model composed of artificial "neurons", vaguely inspired by the simplification of a biological neural network.

This model consists of a group of information interconnections made up of artificial and computational neurons processes. In most cases, an artificial neural network is an adaptive system that changes its own structure based on external or internal information flowing through the network itself during learning phase.

An artificial neural network receives external signals on a layer of input nodes (processing units), each of which is connected with numerous internal nodes, organized in several layers. Each node processes the received signals and transmits the result to subsequent nodes.

A neural Network model it can be both supervised and unsupervised. In this project we will use the ***supervised model***.

2.3 Data exploration

Let's explore the data structure

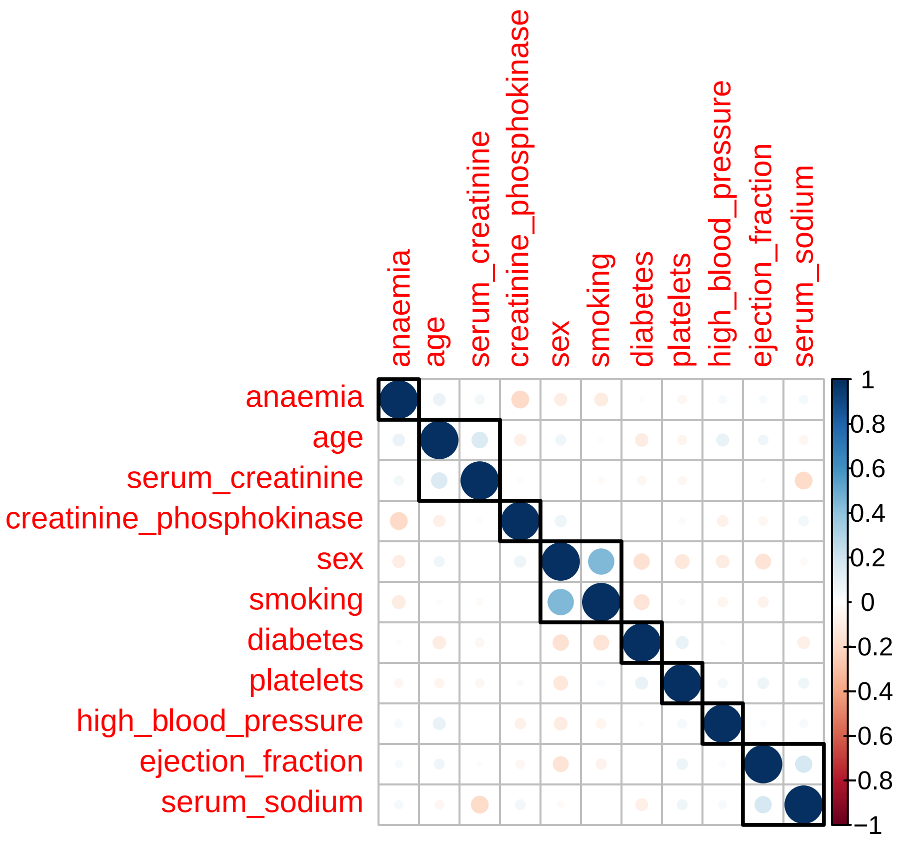
```
## 'data.frame':    299 obs. of  13 variables:
##  $ age                     : num  75 55 65 50 65 90 75 60 65 80 ...
##  $ anaemia                 : int  0 0 0 1 1 1 1 1 0 1 ...
##  $ creatinine_phosphokinase: int  582 7861 146 111 160 47 246 315 157 123 ...
##  $ diabetes                : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ ejection_fraction       : int  20 38 20 20 20 40 15 60 65 35 ...
##  $ high_blood_pressure     : int  1 0 0 0 0 1 0 0 0 1 ...
##  $ platelets               : num  265000 263358 162000 210000 327000 ...
##  $ serum_creatinine        : num  1.9 1.1 1.3 1.9 2.7 2.1 1.2 1.1 1.5 9.4 ...
##  $ serum_sodium            : int  130 136 129 137 116 132 137 131 138 133 ...
##  $ sex                     : int  1 1 1 1 0 1 1 1 0 1 ...
##  $ smoking                 : int  0 0 1 0 0 1 0 1 0 1 ...
##  $ time                    : int  4 6 7 7 8 8 10 10 10 10 ...
##  $ DEATH_EVENT             : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
##       age           anaemia       creatinine_phosphokinase    diabetes
##  Min.   :40.00   Min.   :0.0000   Min.   :  23.0           Min.   :0.0000
##  1st Qu.:51.00   1st Qu.:0.0000   1st Qu.: 116.5           1st Qu.:0.0000
##  Median :60.00   Median :0.0000   Median : 250.0           Median :0.0000
##  Mean   :60.83   Mean   :0.4314   Mean   : 581.8           Mean   :0.4181
##  3rd Qu.:70.00   3rd Qu.:1.0000   3rd Qu.: 582.0           3rd Qu.:1.0000
##  Max.   :95.00   Max.   :1.0000   Max.   :7861.0           Max.   :1.0000
##  ejection_fraction high_blood_pressure   platelets      serum_creatinine
##  Min.   :14.00     Min.   :0.0000      Min.   : 25100   Min.   :0.500
##  1st Qu.:30.00     1st Qu.:0.0000      1st Qu.:212500   1st Qu.:0.900
##  Median :38.00     Median :0.0000      Median :262000   Median :1.100
##  Mean   :38.08     Mean   :0.3512      Mean   :263358   Mean   :1.394
##  3rd Qu.:45.00     3rd Qu.:1.0000      3rd Qu.:303500   3rd Qu.:1.400
##  Max.   :80.00     Max.   :1.0000      Max.   :850000   Max.   :9.400
##   serum_sodium        sex            smoking            time
##  Min.   :113.0   Min.   :0.0000   Min.   :0.0000   Min.   :  4.0
##  1st Qu.:134.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 73.0
##  Median :137.0   Median :1.0000   Median :0.0000   Median :115.0
##  Mean   :136.6   Mean   :0.6488   Mean   :0.3211   Mean   :130.3
##  3rd Qu.:140.0   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:203.0
##  Max.   :148.0   Max.   :1.0000   Max.   :1.0000   Max.   :285.0
##   DEATH_EVENT
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.3211
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

From the analysis of the observations we can understand that there are no missing values or duplicated observations

2.3.1 Correlation between predictors

We analyze the correlation between variables through the use of graphs showing the correlation matrix



The correlation is below 0.60, therefore well below the cut-off of 0.70 below which it can be said that it does not exist high ***collinearity*** between predictors, but it is not enough to be confident in using ***Naive Bayes*** algorithm.

The percentages of observations for the two possible target values are as follows:

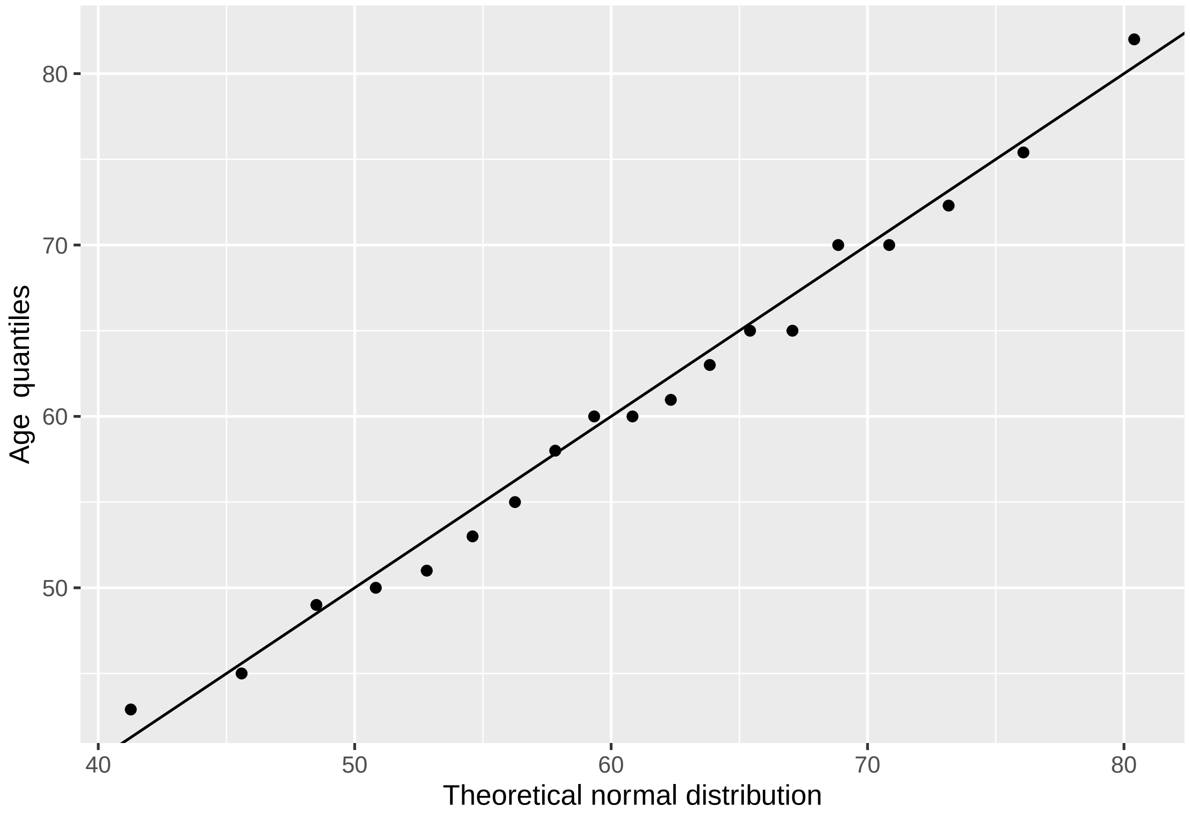
1. No: 67.9%
2. Yes: 32.1%

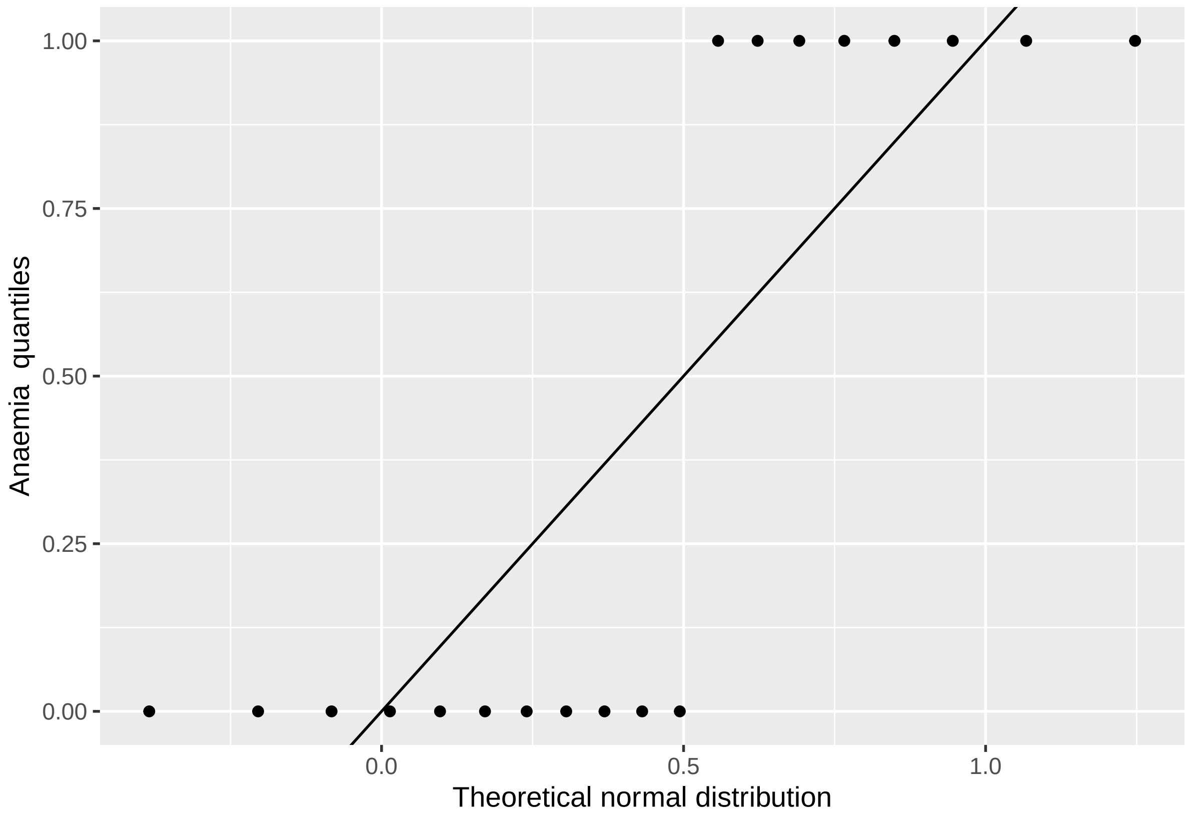
Check proportion of data

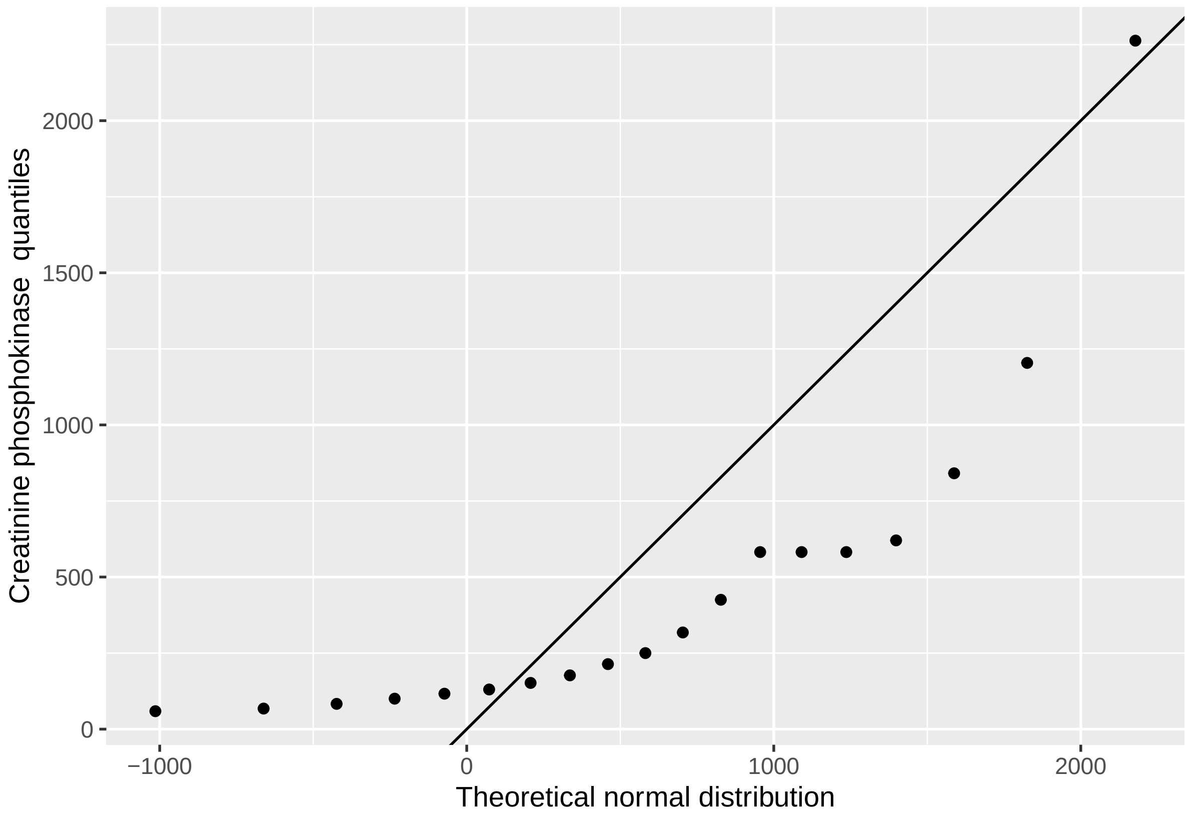
```
##
##    no   yes
## 0.679 0.321
```

2.3.2 Data distribution

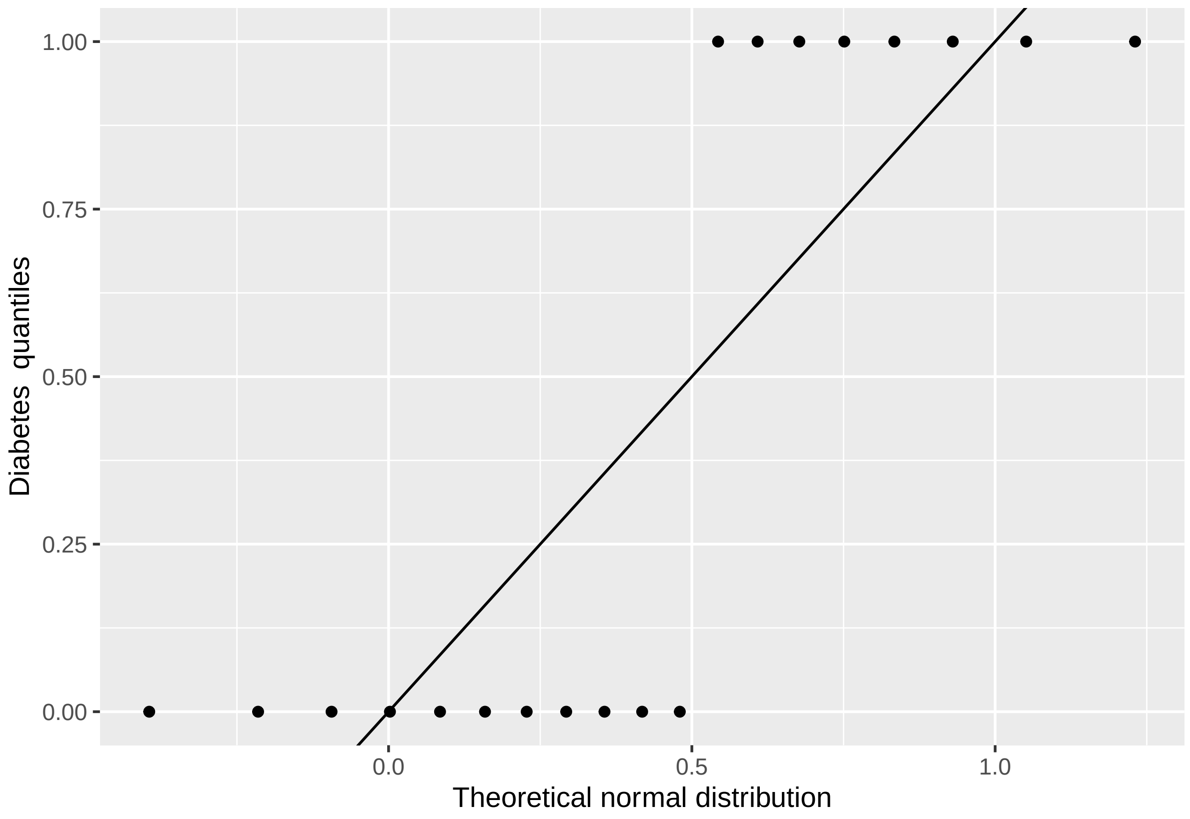
We investigate the individual variables to understand if they are normally distributed. For this purpose, we use graphs and then implement the Shapiro-Wilk test.



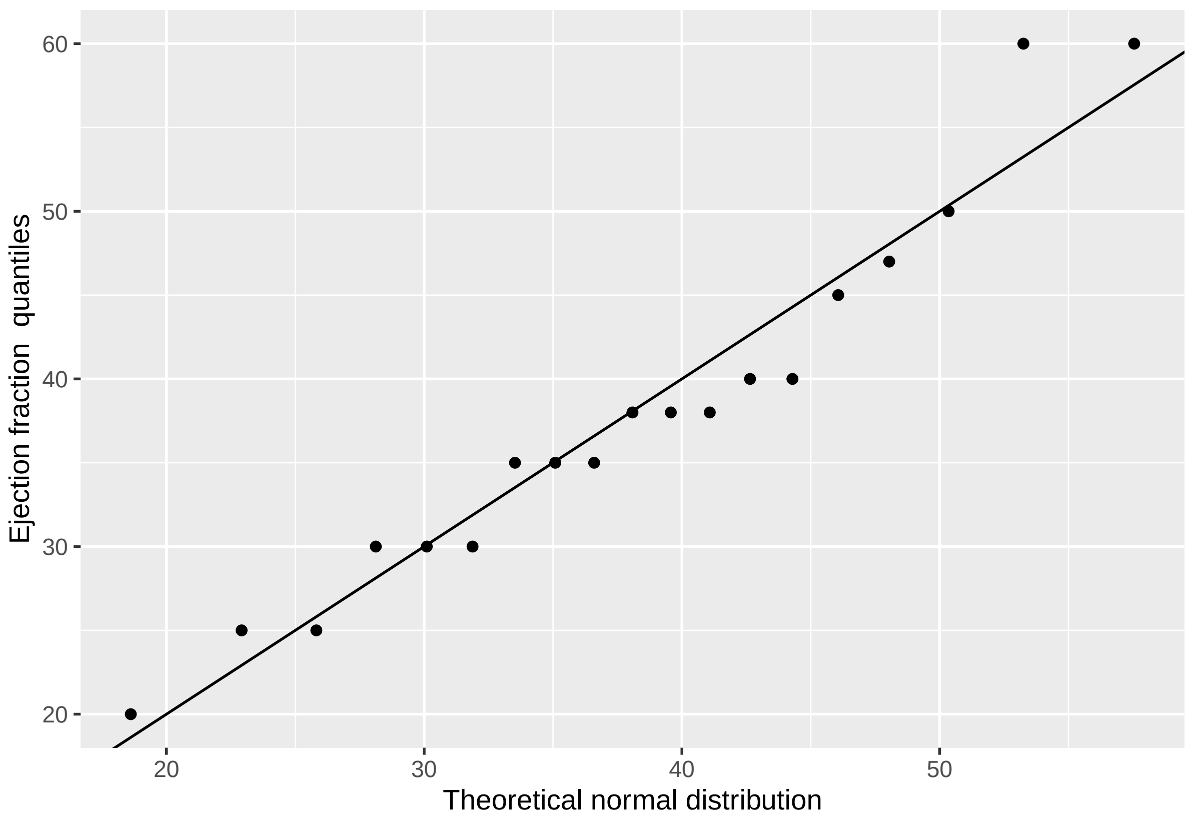




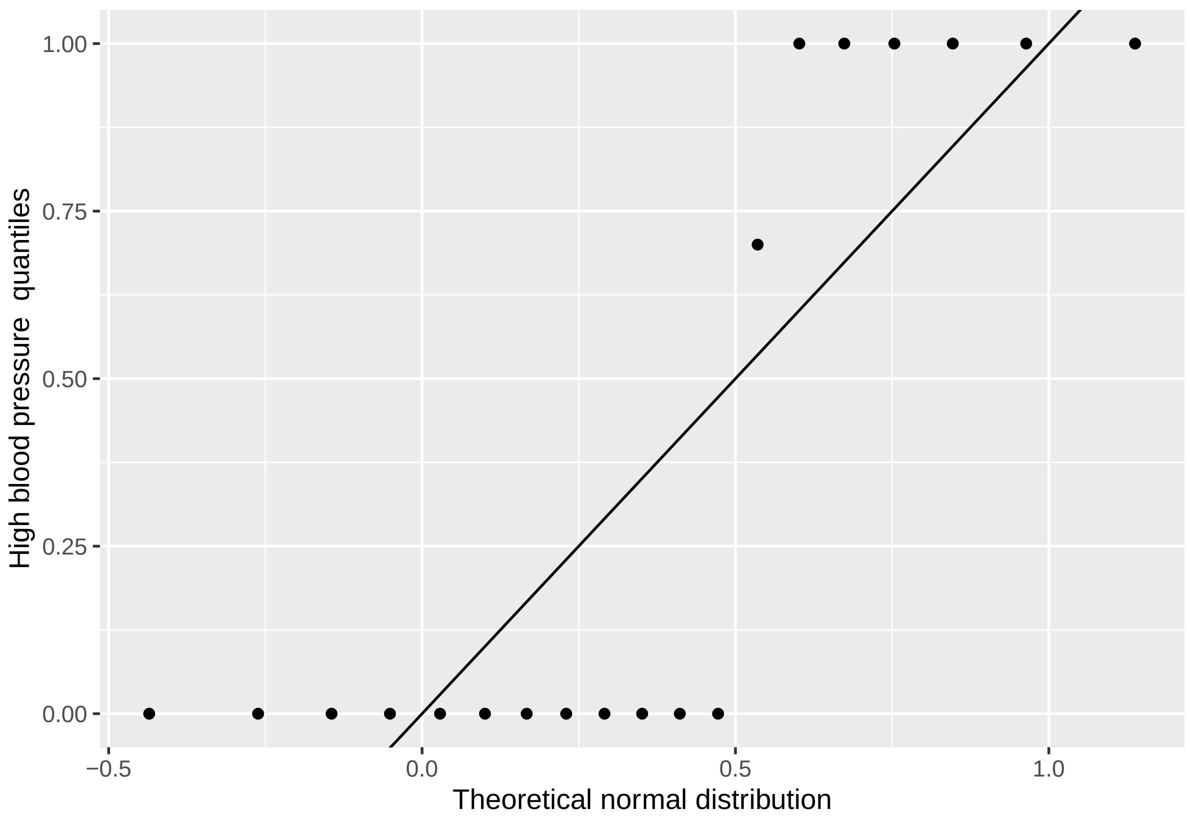
Creatinine phosphokinase quantiles
Theoretical normal distribution
2000
1500
1000
500
0
−1000
0
1000
2000



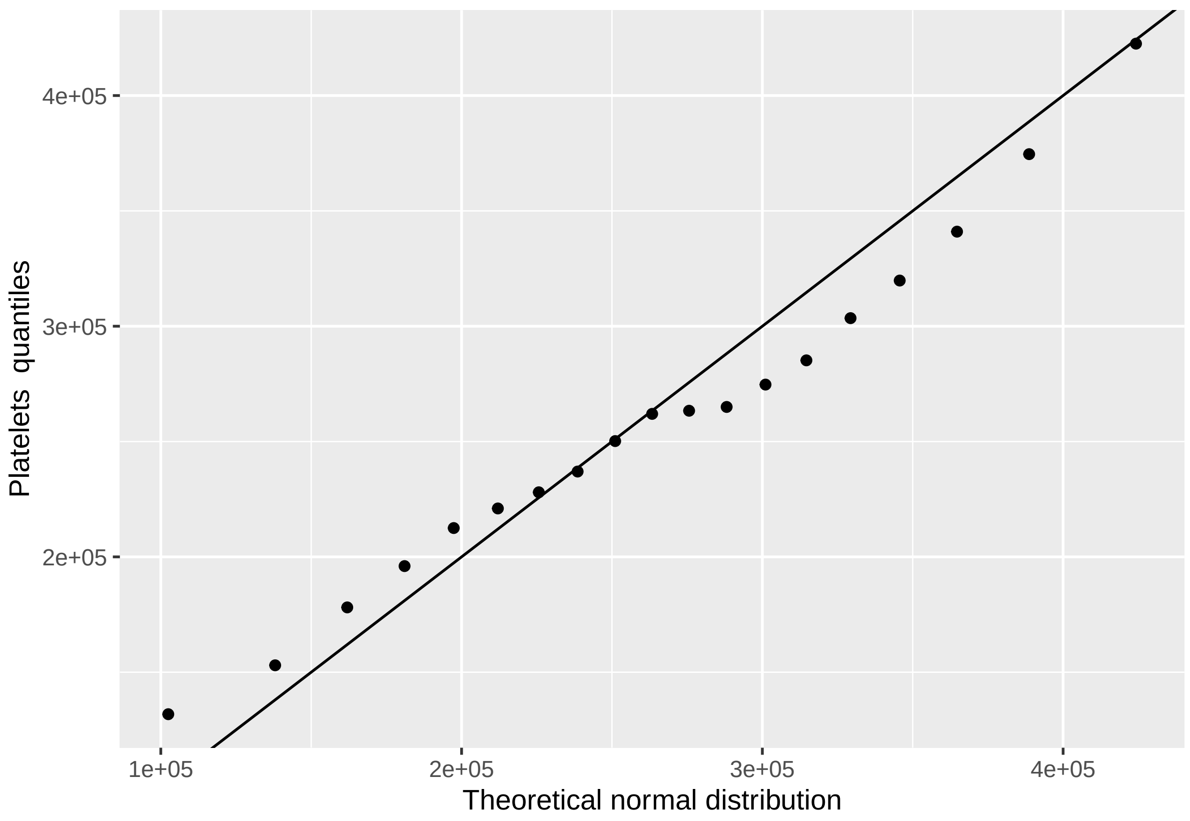
Diabetes quantiles
Theoretical normal distribution
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0



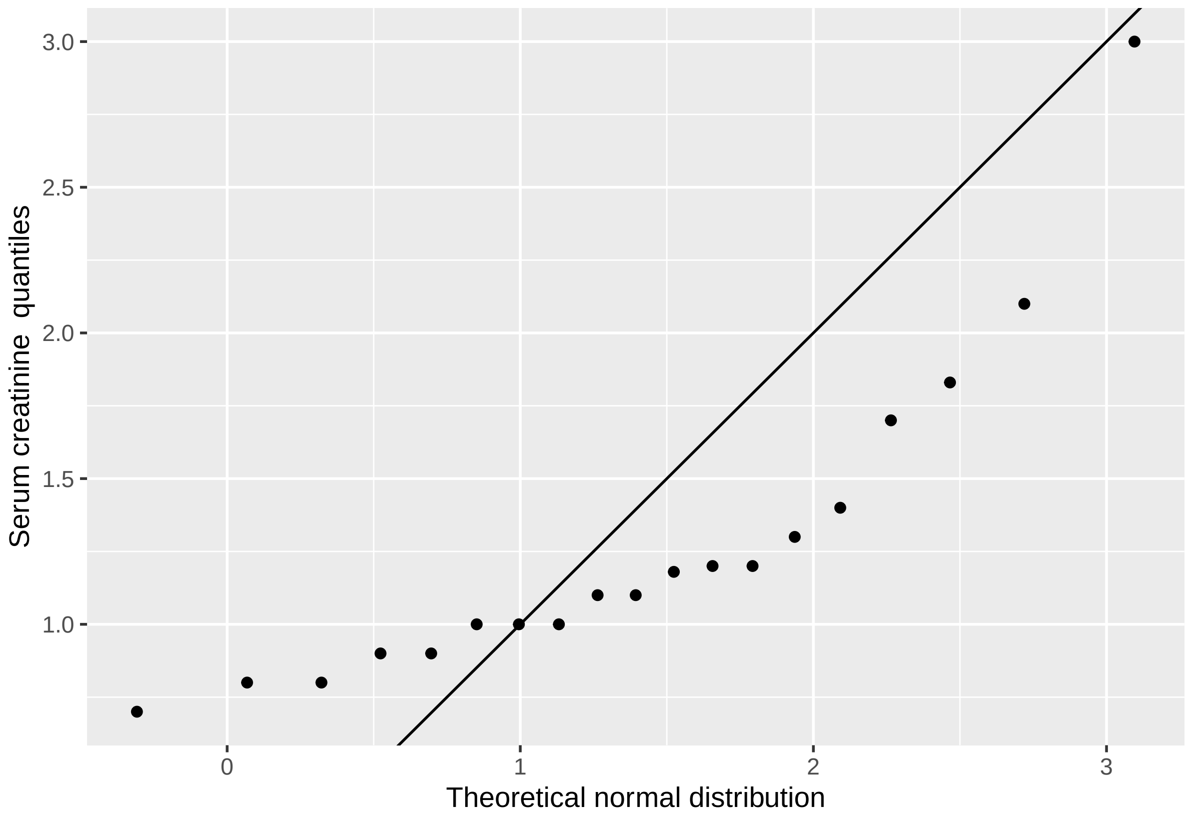
Ejection fraction quantiles
Theoretical normal distribution
20
30
40
50
60



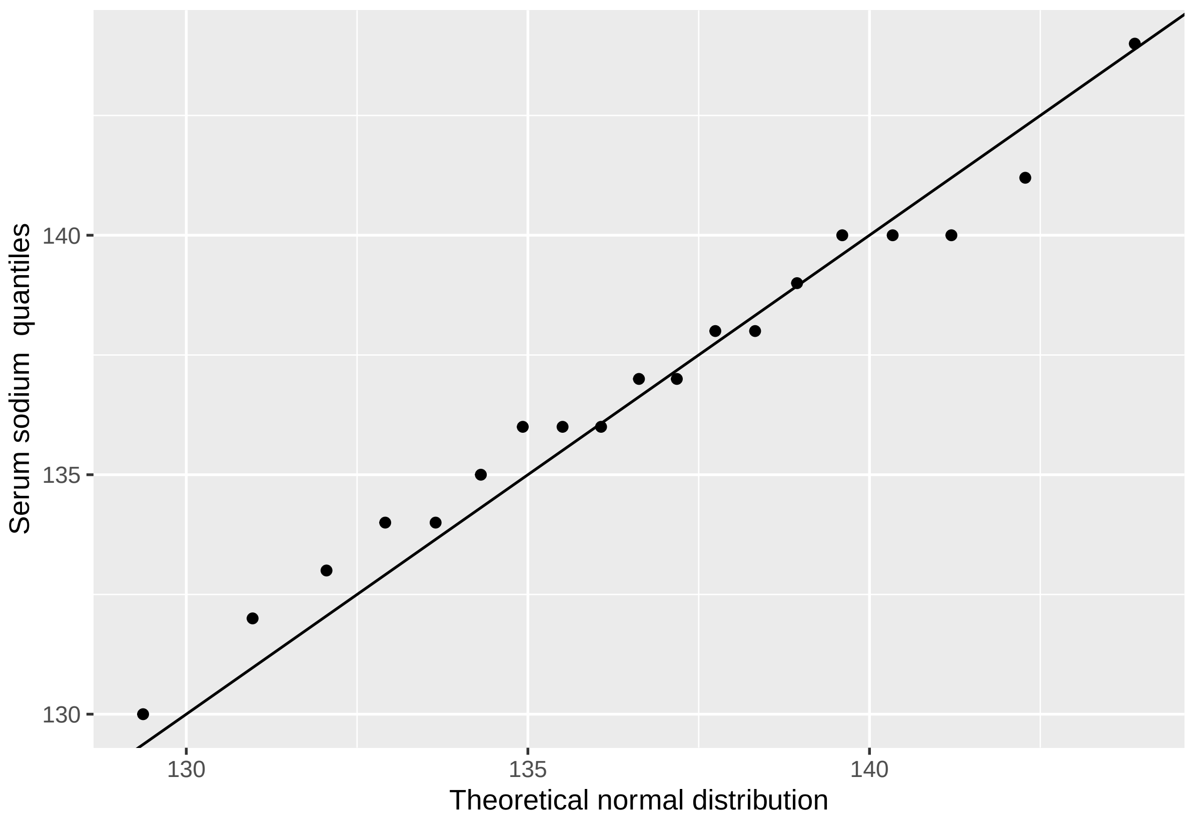
High blood pressure quantiles
Theoretical normal distribution
0.00
0.25
0.50
0.75
1.00
−0.5
0.0
0.5
1.0

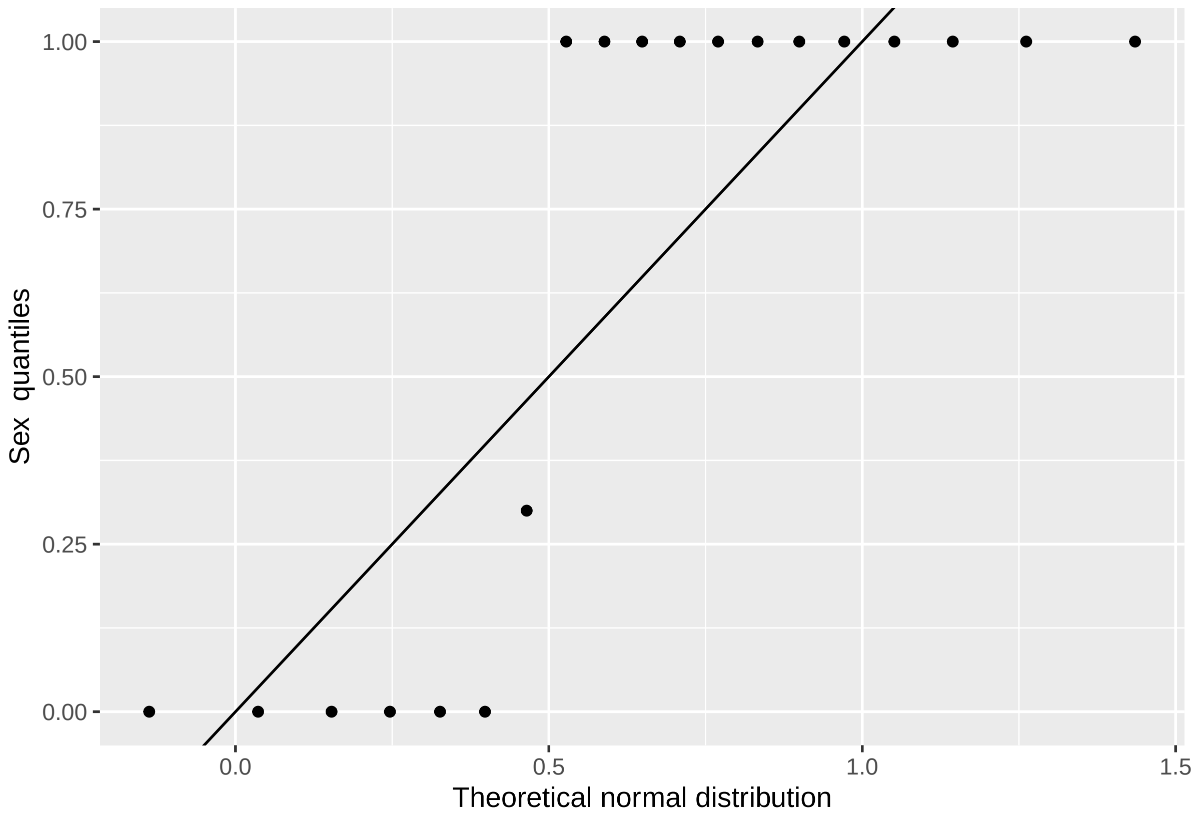


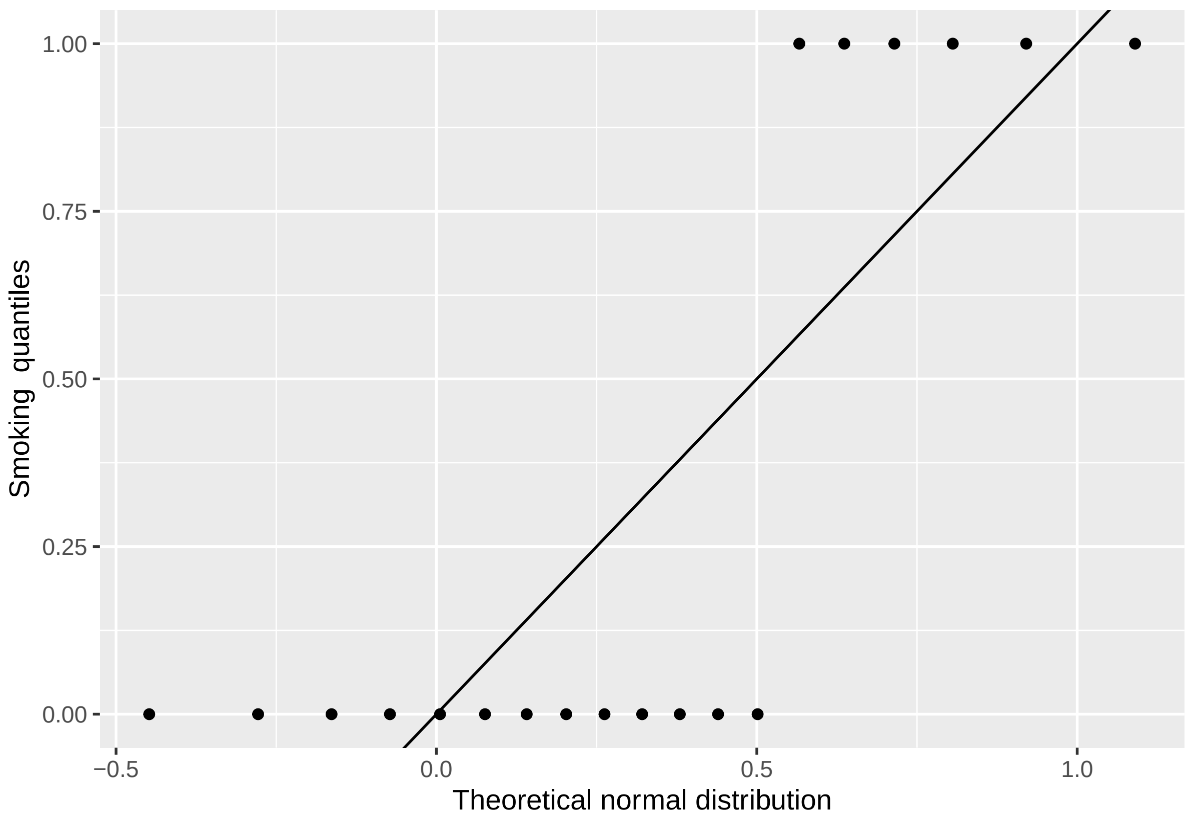
Platelets quantiles
4e+05
3e+05
2e+05
1e+05
2e+05
3e+05
4e+05
Theoretical normal distribution



Serum creatinine quantiles
3.0
2.5
2.0
1.5
1.0
0
1
2
3
Theoretical normal distribution


Serum sodium quantiles
130
135
140
Theoretical normal distribution


Sex quantiles
0.00
0.25
0.50
0.75
1.00
0.0
0.5
1.0
1.5
Theoretical normal distribution



Only Age and Serum sodium they appear to have a normal distribution

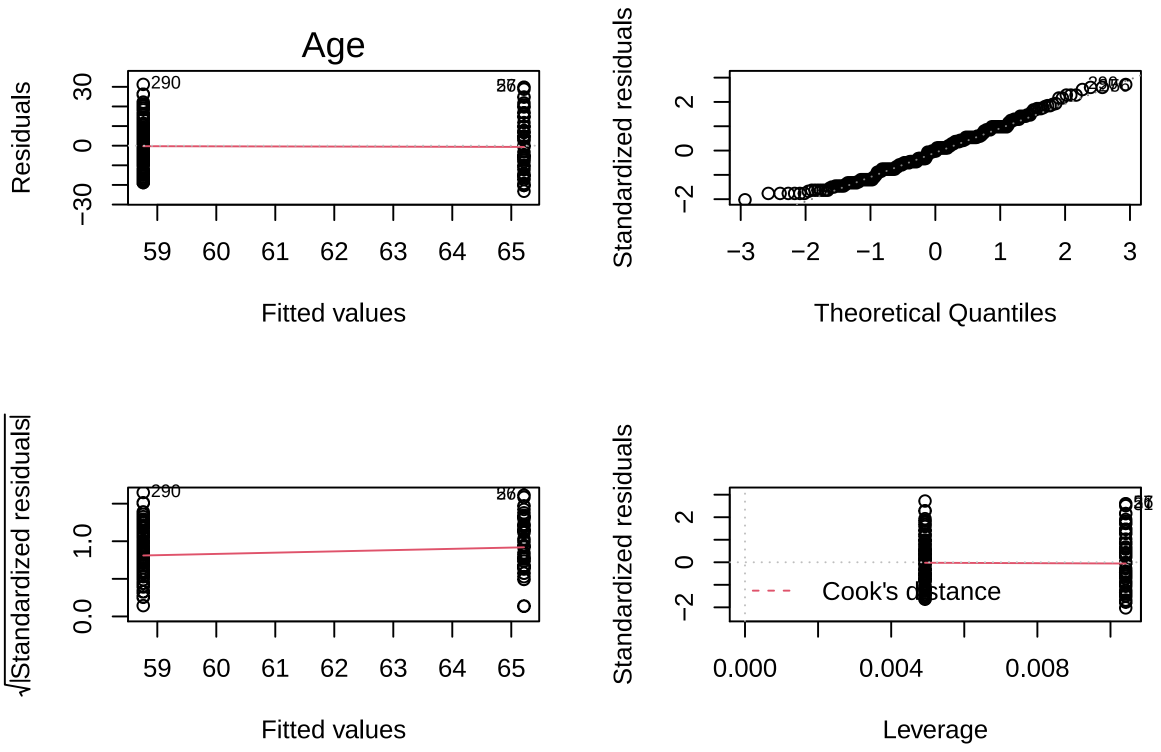
To understand how much each variable differs from the normal distribution we implement the Shapiro-Wilk test

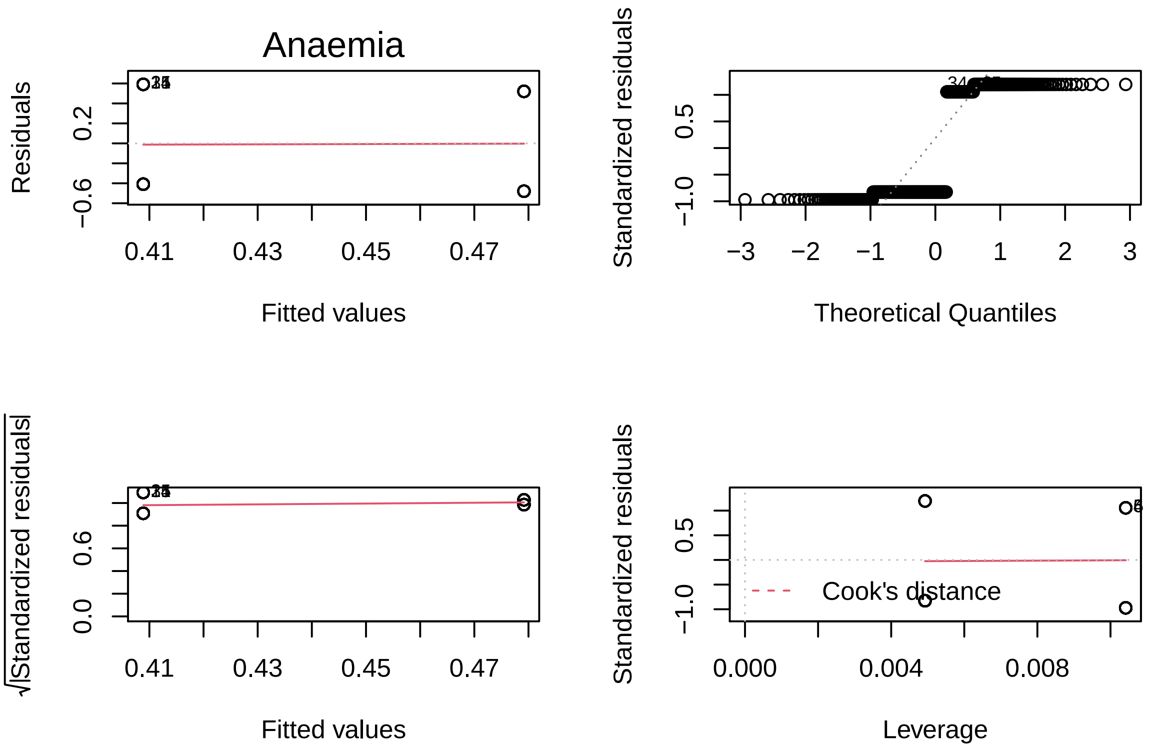
Variable	P-VALUE
age	5.35e-05
anaemia	6.21e-25
creatinine phosphokinase	7.05e-28
diabetes	5.12e-25
ejection fraction	7.22e-09
high blood pressure	1.17e-25
platelets	2.88e-12
serum creatinine	5.39e-27
serum sodium	9.21e-10
sex	1.17e-25
smoking	4.58e-26

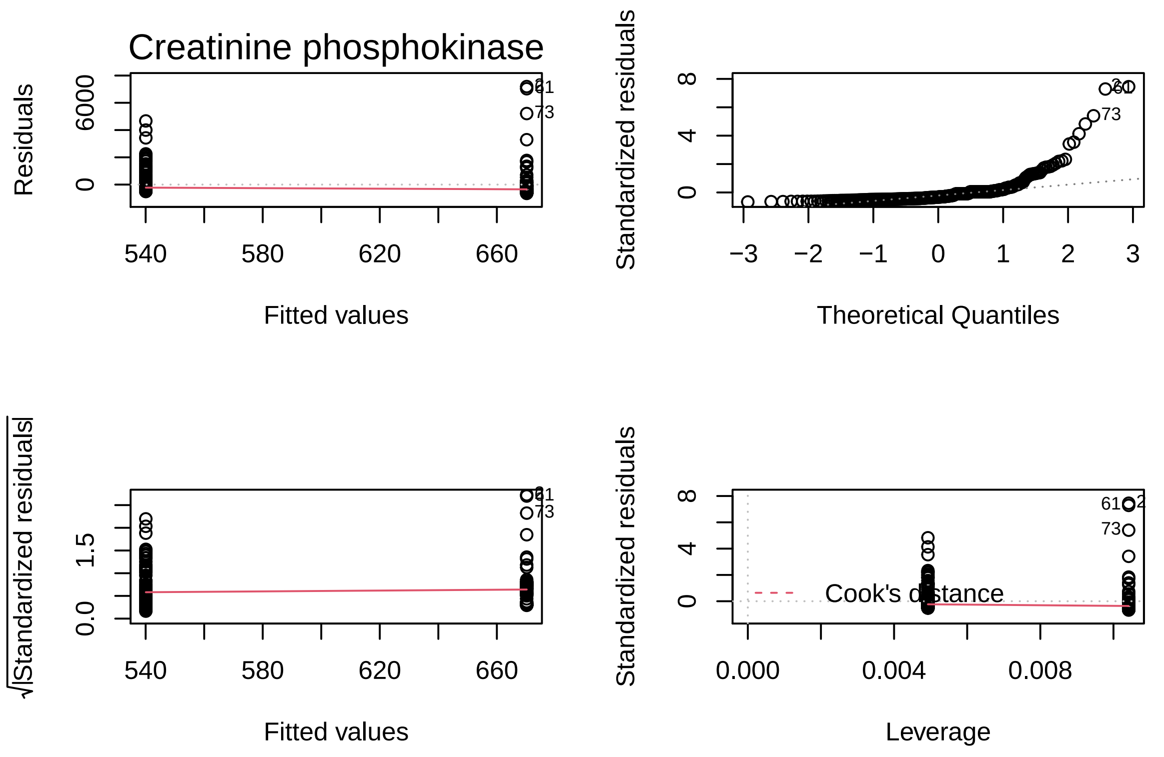
By analyzing the p-value we understand that no variable follow a normal distribution

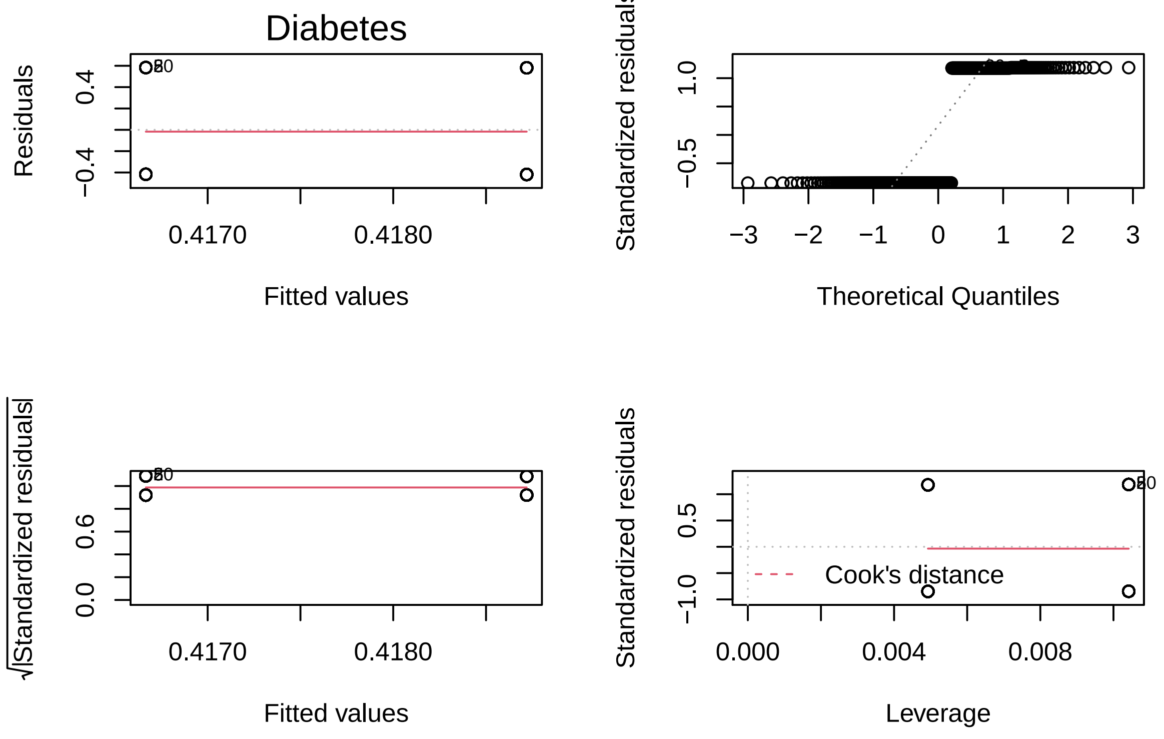
2.3.3 Homoscedasticity analysis

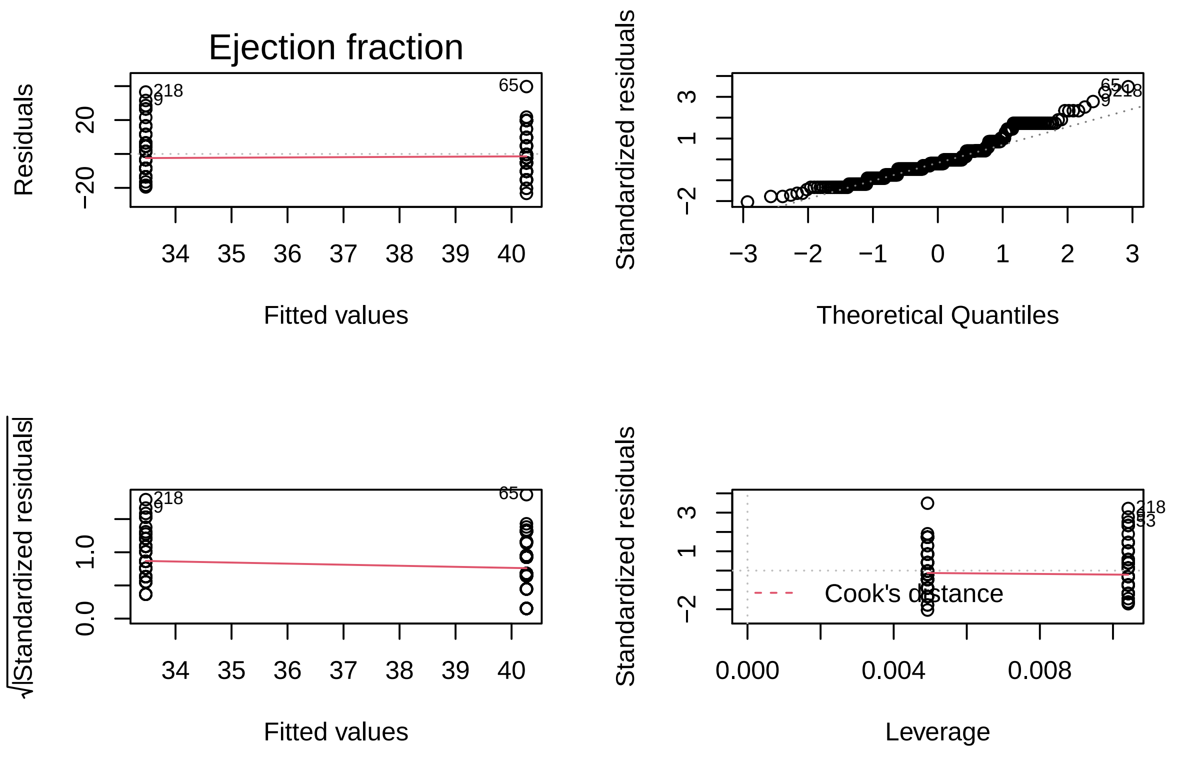
Now we want to understand if is respected the hypothesis of homoscedasticity

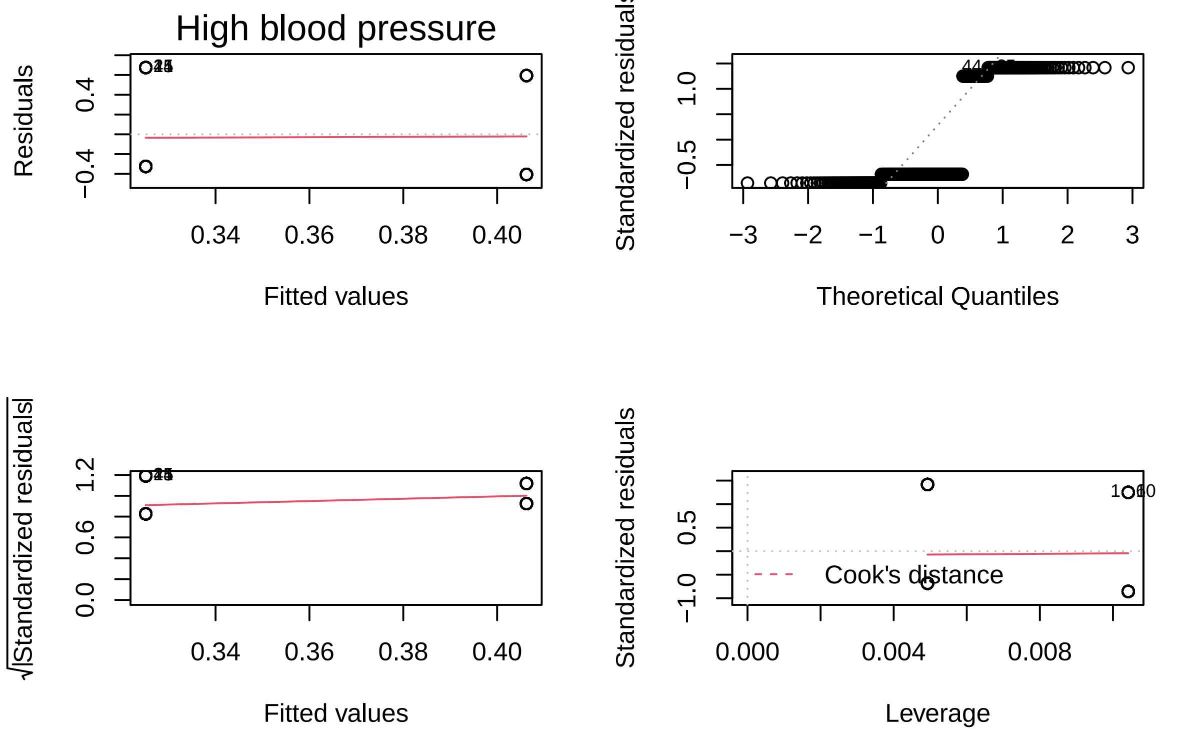


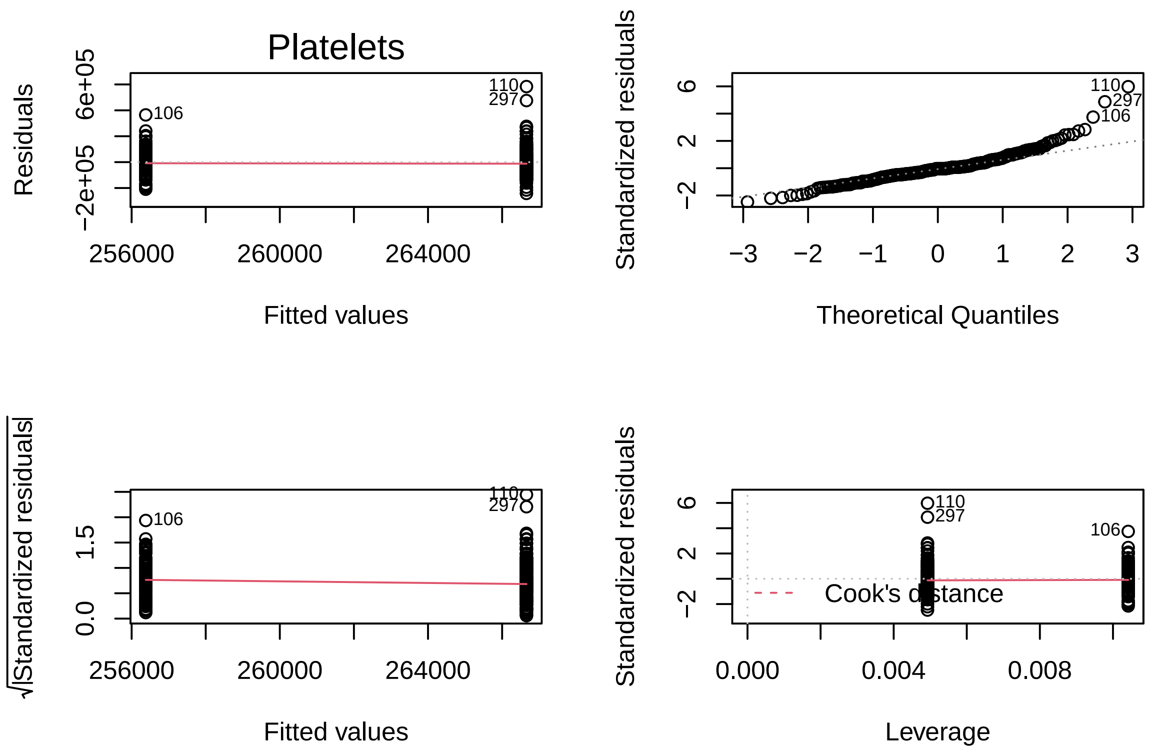


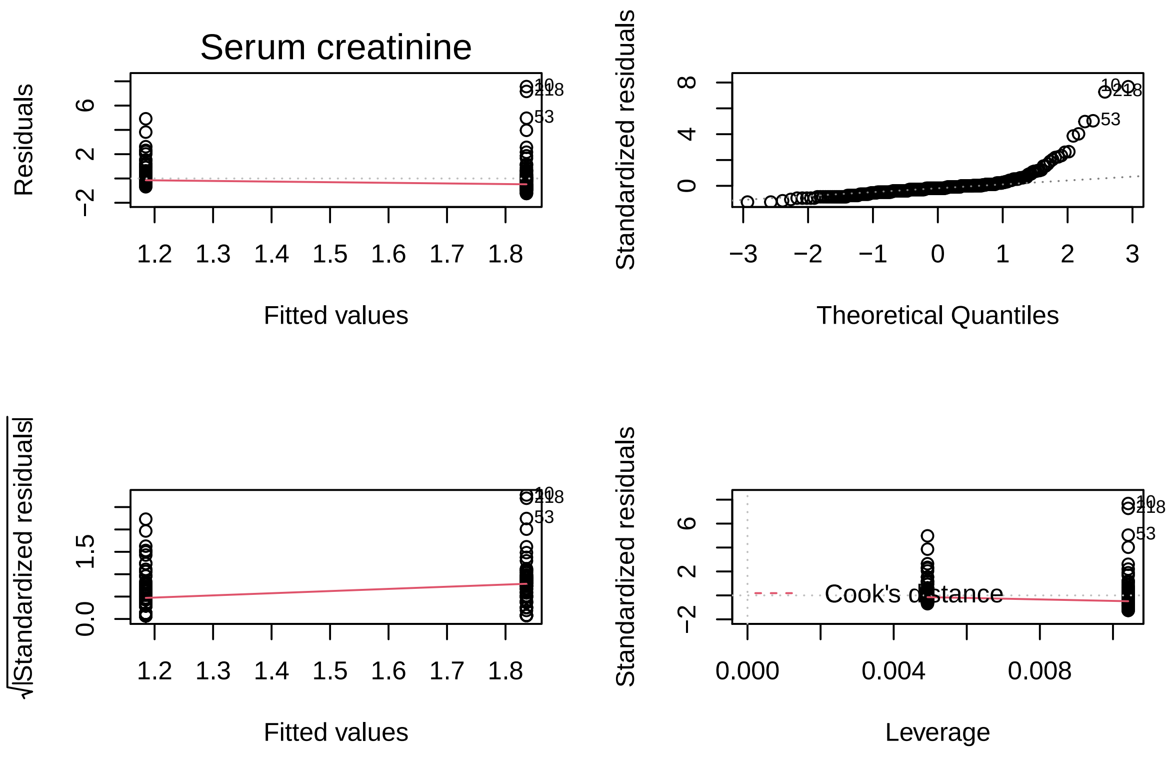

Creatinine phosphokinase
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance
61
73
61
73
61
73

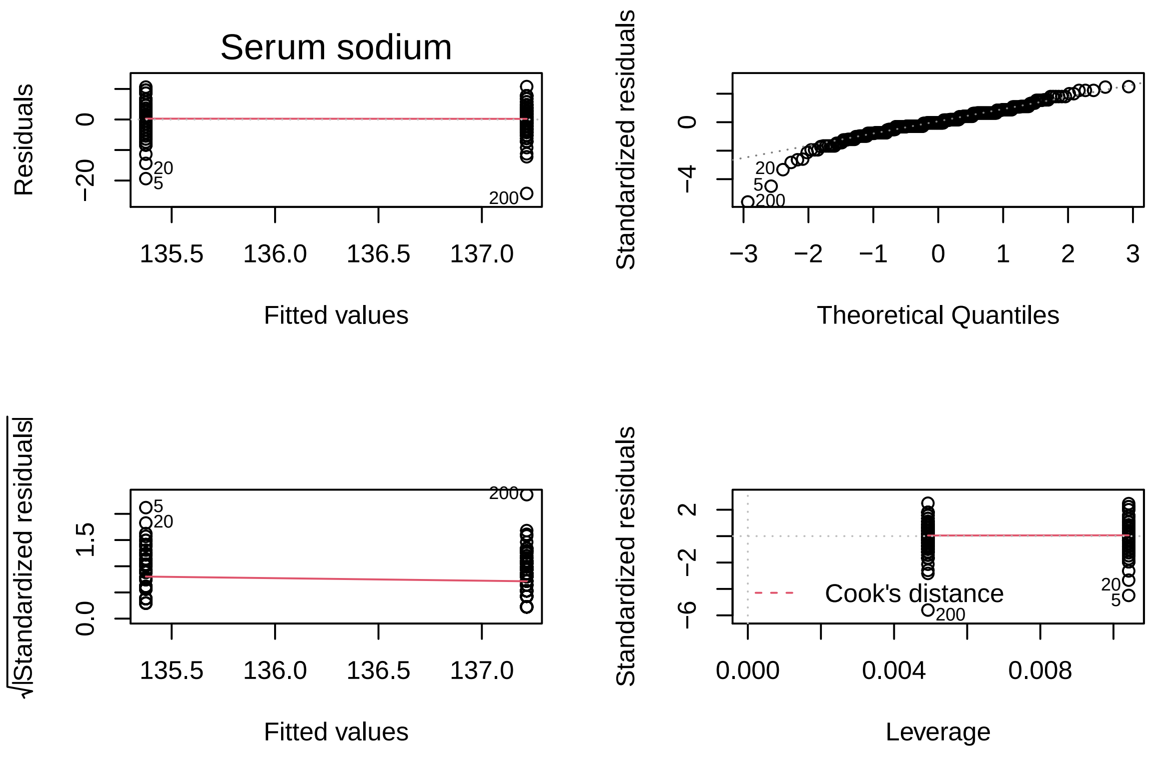

Diabetes
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance
80
80
80


Ejection fraction
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Fitted values
Standardized residuals
Cook's distance
Leverage
218
9
65

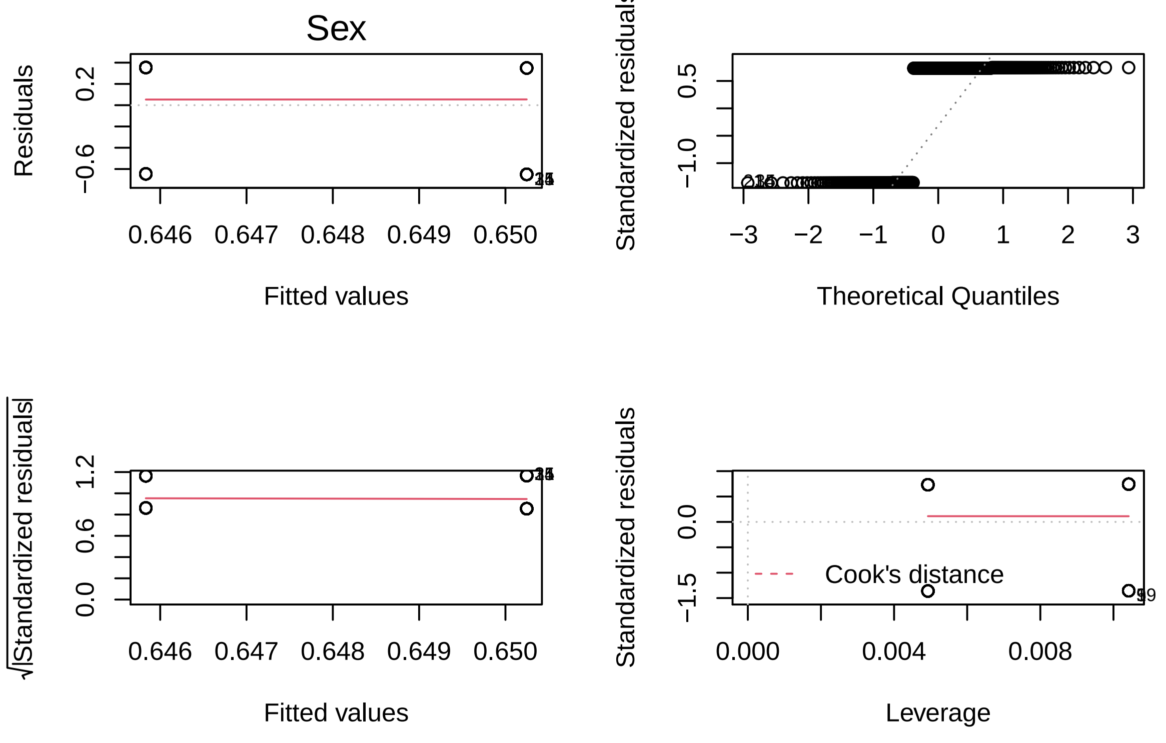

High blood pressure
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Fitted values
Standardized residuals
Cook's distance
Leverage


Platelets
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance
106
110
297

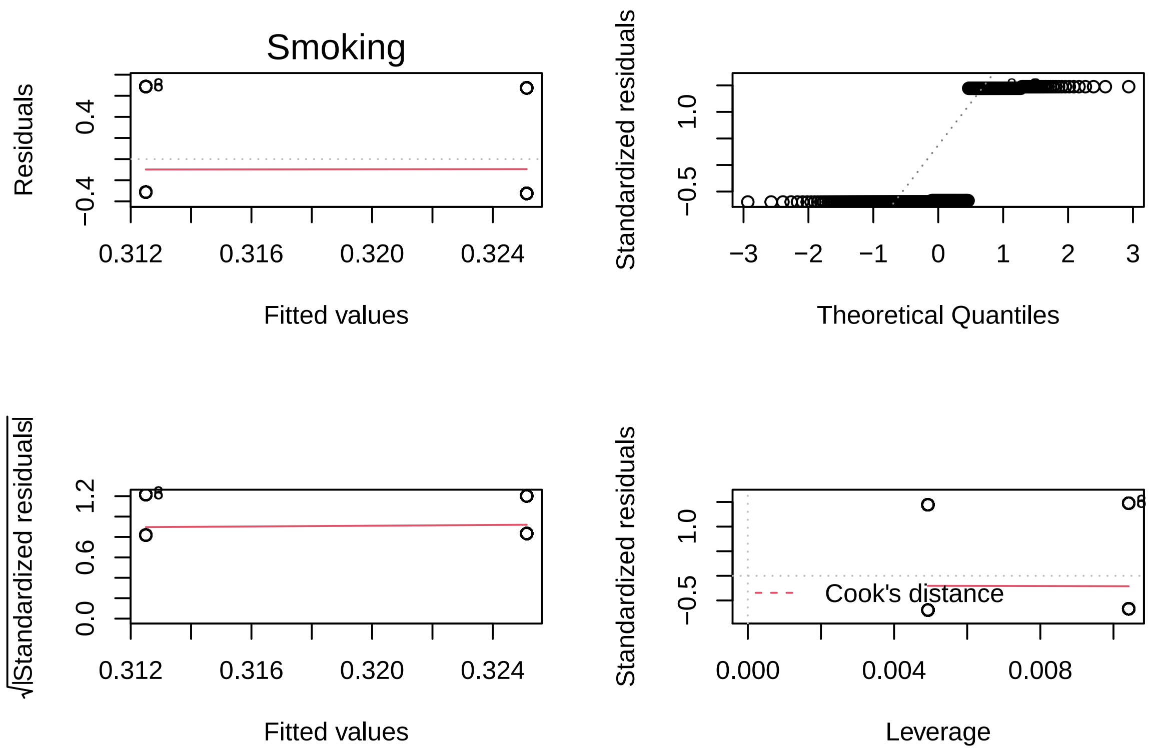

Serum creatinine
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance
10
218
53



Serum sodium
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance



Sex
Residuals
Fitted values
Standardized residuals
Theoretical Quantiles
√|Standardized residuals|
Leverage
Cook's distance



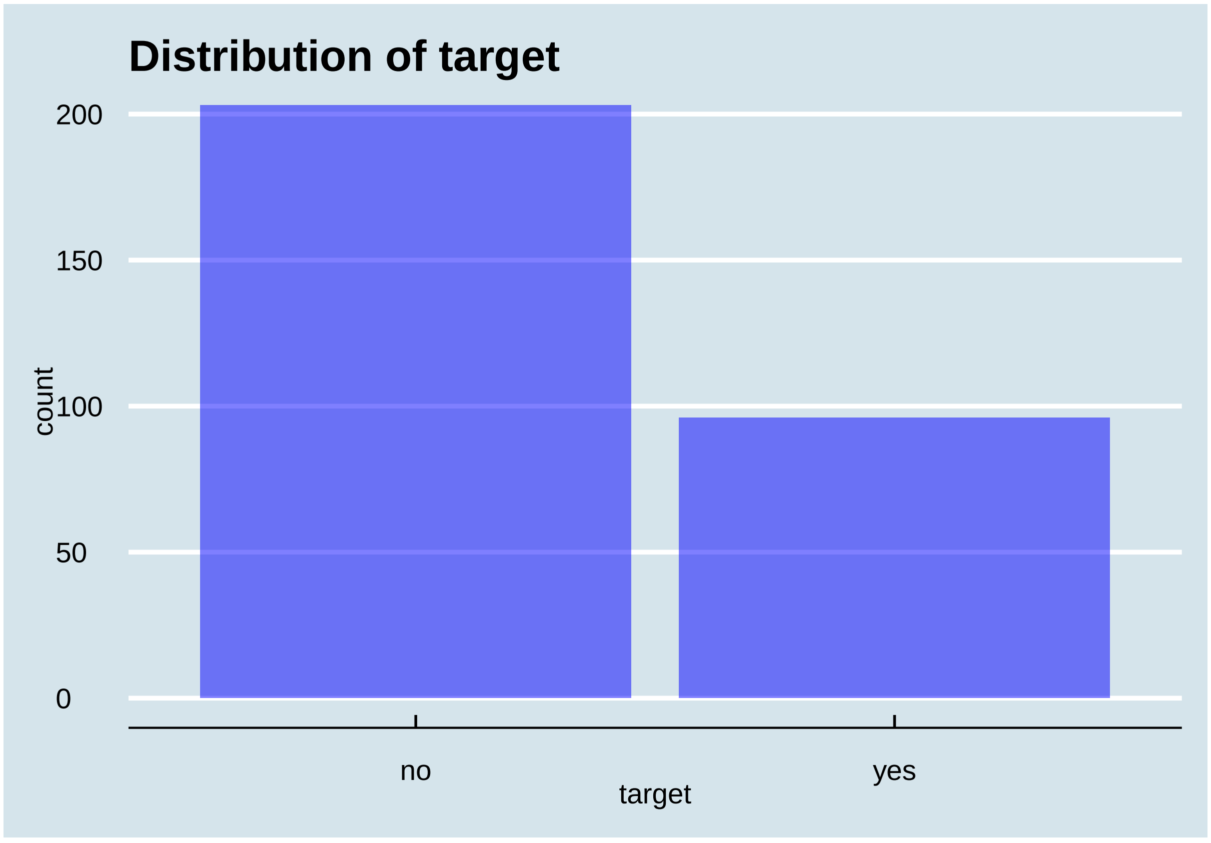
By analyzing the plot of Residuals (top-left) and the one of square root of Standard residuals (bottom-left), we see that no variable respects the hypothesis of homoscedasticity

2.4 Visualization

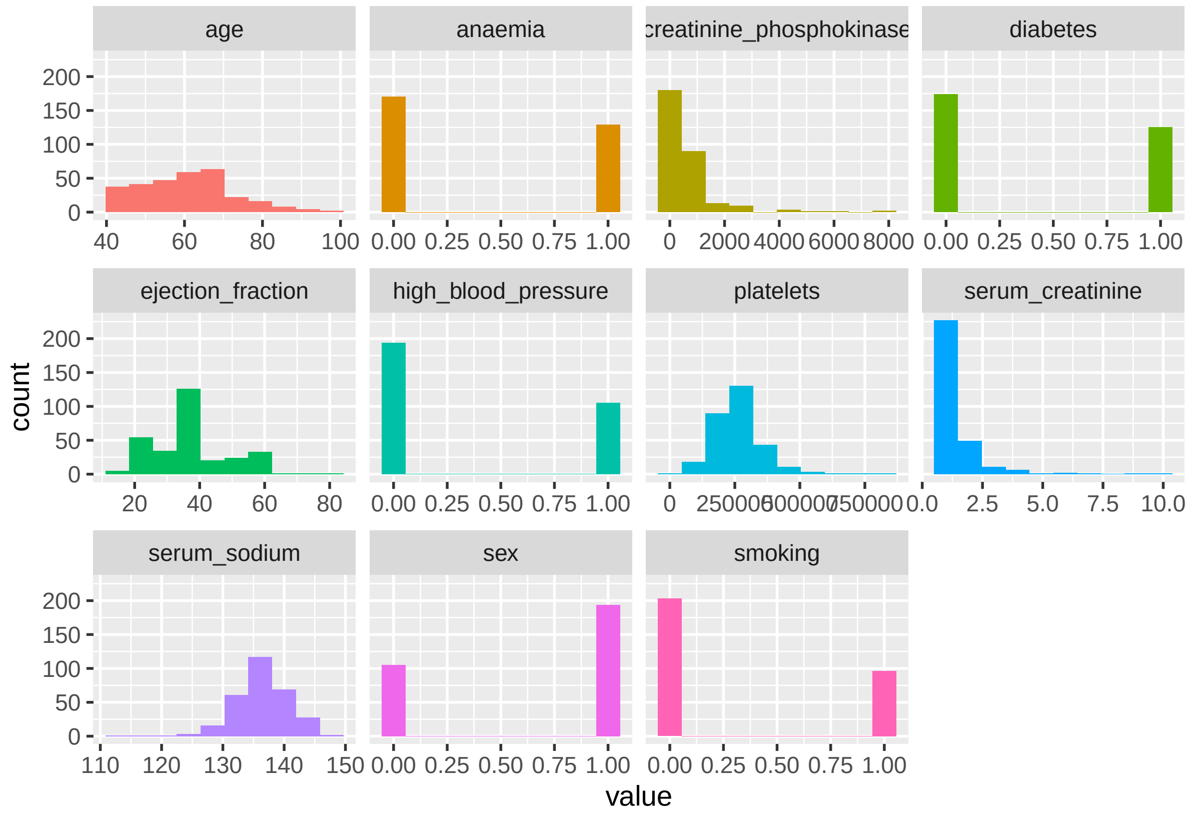
The next graphs show the following distributions:

1. Distribution of target
2. Distribution of variables
3. Distribution of variables for each target value

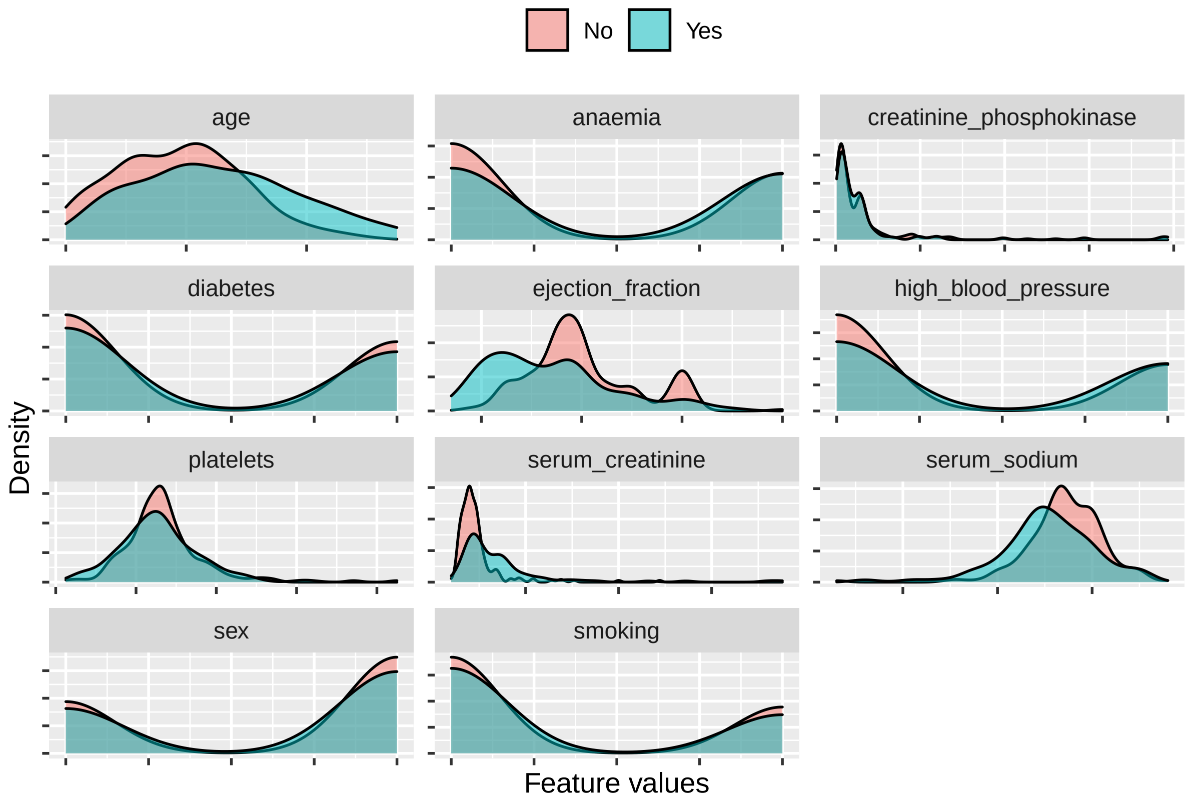
Distribution of target



Distribution of variables

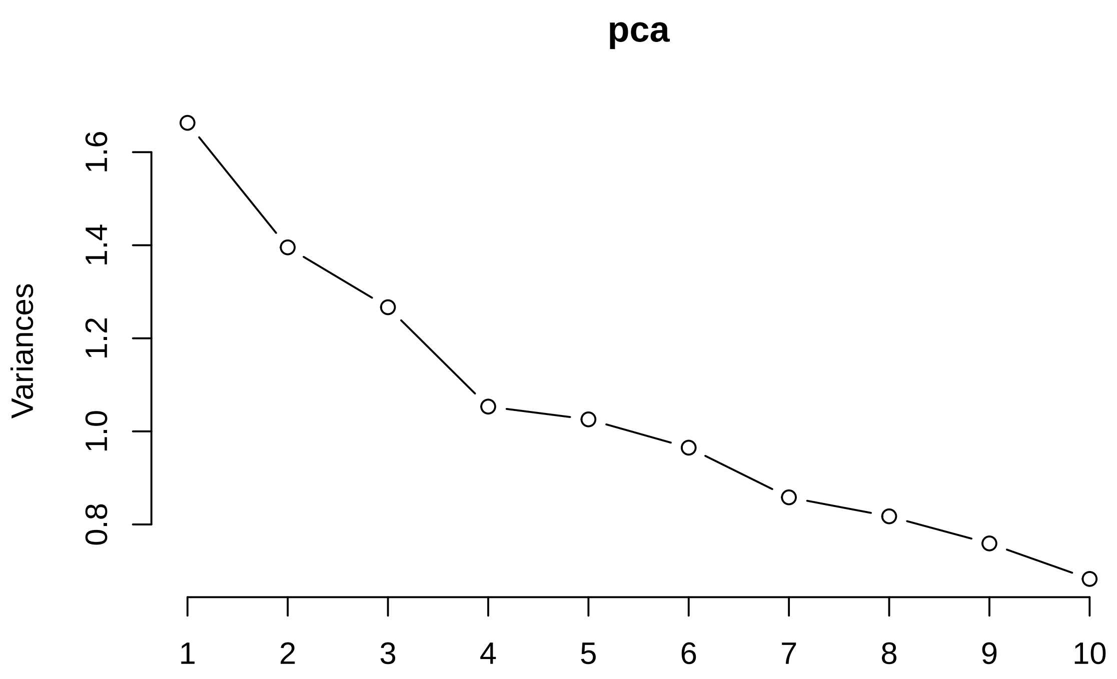


Distribution of variables for each target value

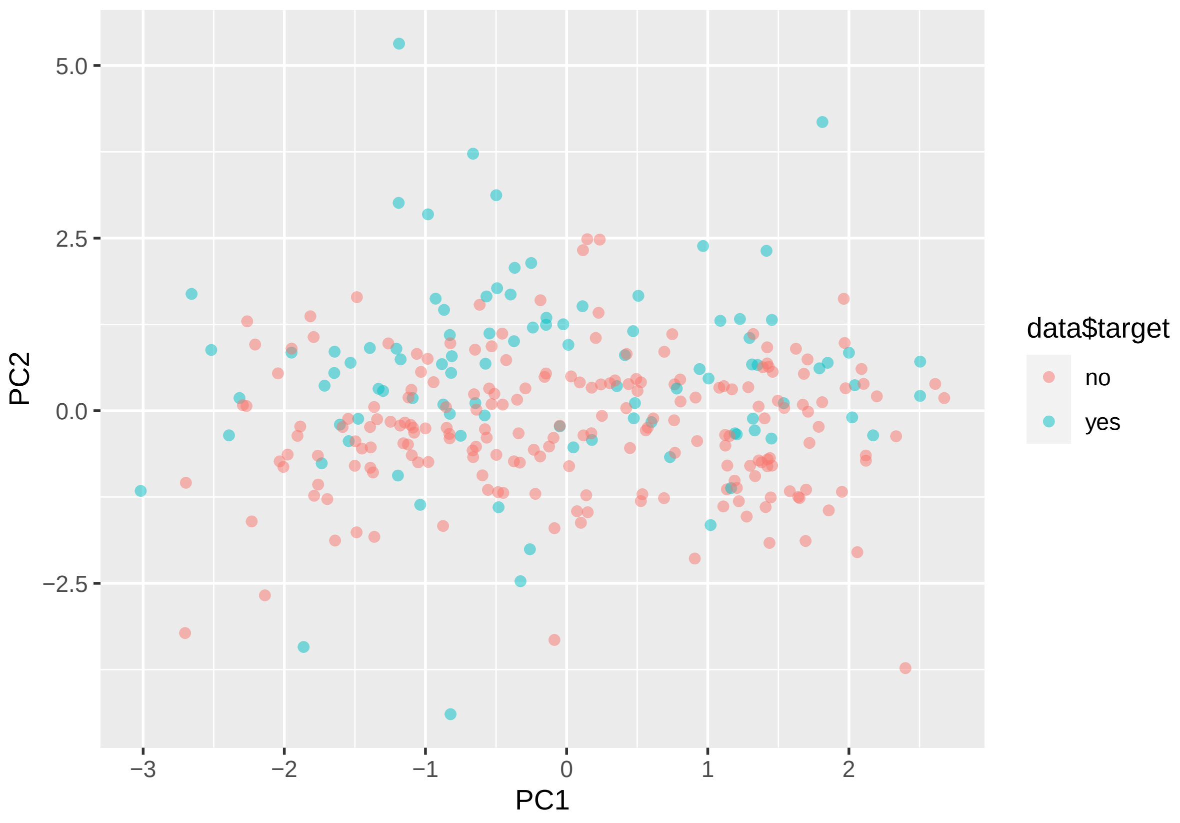


2.4.1 Principal Component Analysis

Now we apply the Principal Component Analysis to understand if it is possible to reduce the number of predictors



```
## Importance of components:
##                          PC1   PC2   PC3    PC4    PC5    PC6   PC7    PC8
## Standard deviation     1.290 1.181 1.125 1.0263 1.0128 0.9823 0.926 0.9041
## Proportion of Variance 0.151 0.127 0.115 0.0958 0.0932 0.0877 0.078 0.0743
## Cumulative Proportion  0.151 0.278 0.393 0.4889 0.5822 0.6699 0.748 0.8223
##                          PC9  PC10   PC11
## Standard deviation     0.871 0.8264 0.7162
## Proportion of Variance 0.069 0.0621 0.0466
## Cumulative Proportion  0.891 0.9534 1.0000
```



The data of the first 2 components cannot be easly separated into two classes.

We need 11 variables to reach 95% of the variance, so there is no point to implement PCA to reduce the number of predictors.

We are going to use all predictors on dataset.

3 Results

We begin the implementation of the various algorithms described above.

For the implementation of the algorithms we will use the Caret package

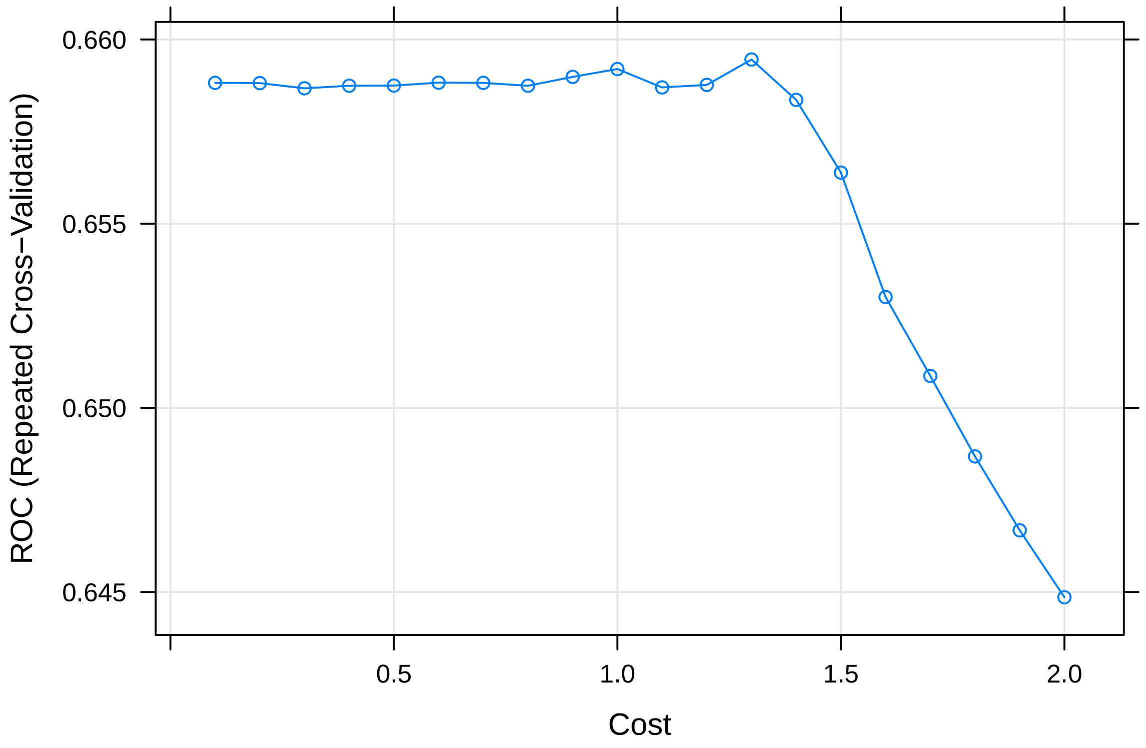
We start creating the partition 80% and 20% to create respectively the training dataset and test dataset.

For each algorithm, the following operations will be performed:

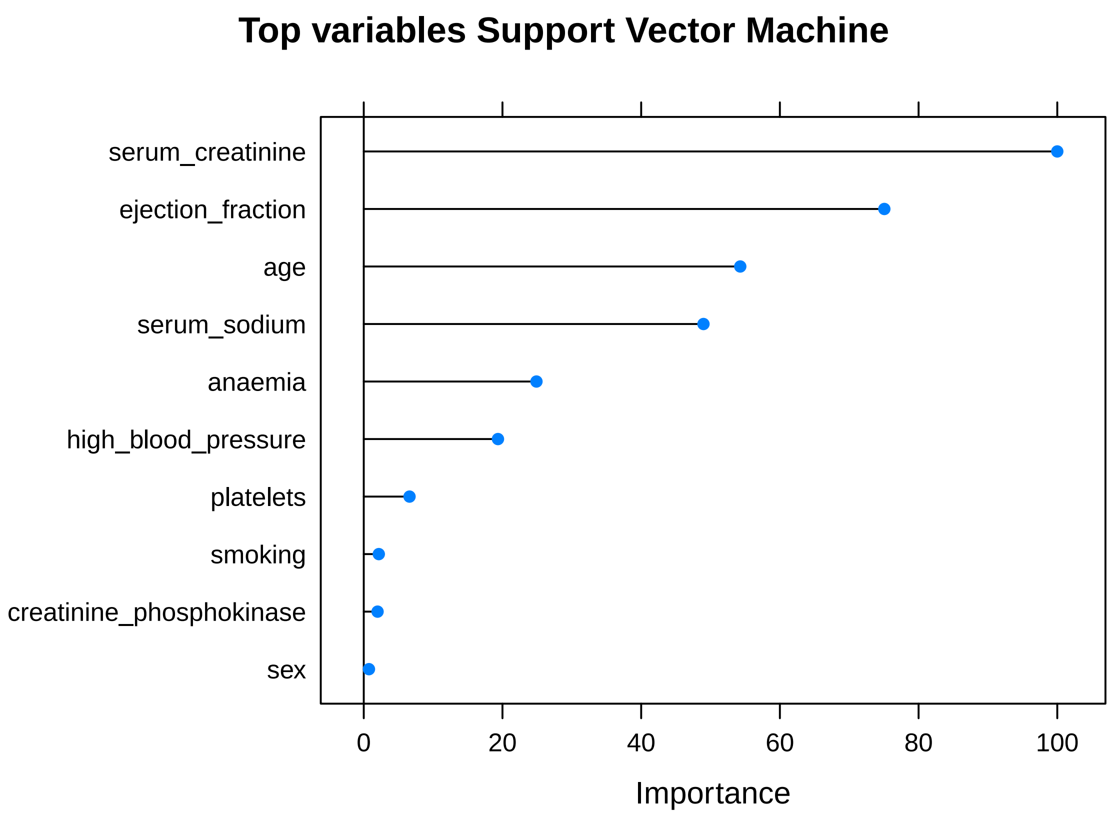
1. set up a grid of adjustment parameters which, by adapting to the model and calculating its performance, will allow us to determine the values that provide optimal performance;
2. Train model;
3. Exploration and visualization of the model, with indication of the best values for tuning parameters;
4. Perform the prediction;
5. Evaluation of confusion matrix will show the best values for parameters and the relative performance obtained;
6. Show the 10 most important predictors;
7. Show the ROC cure with highlighted the best value for Specificity and Sensitivity.

3.1 Support Vector Machine

Tuning parameters: 1. sigma (Sigma) 2. C (Cost)

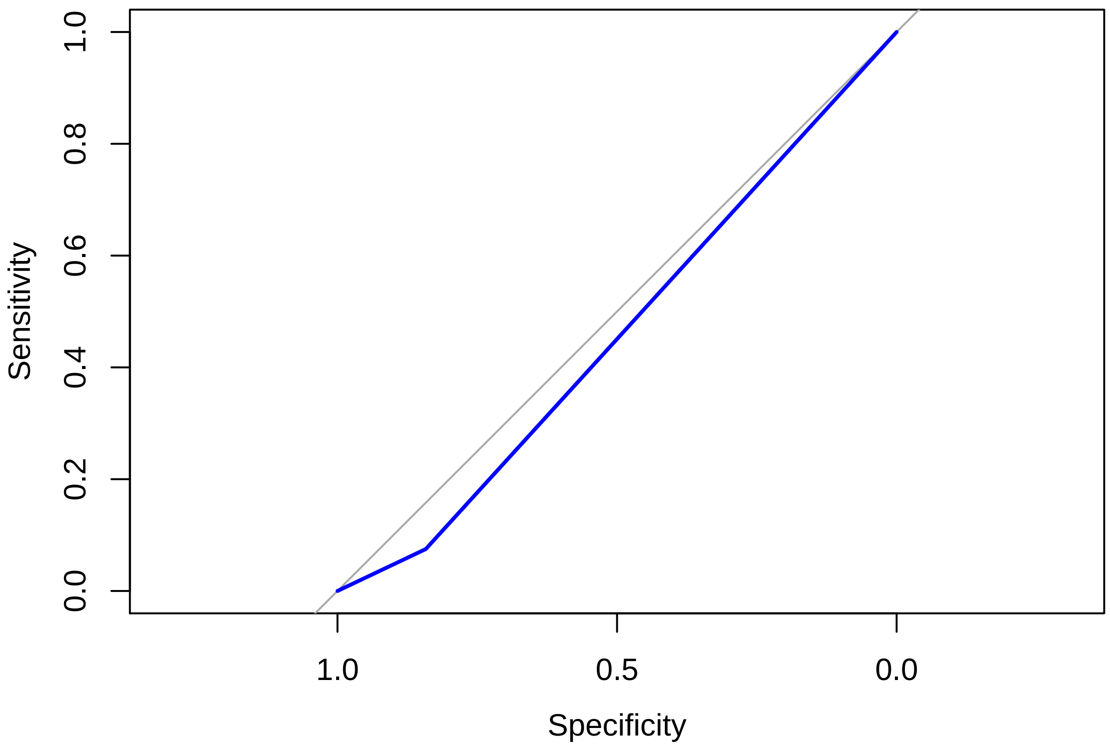


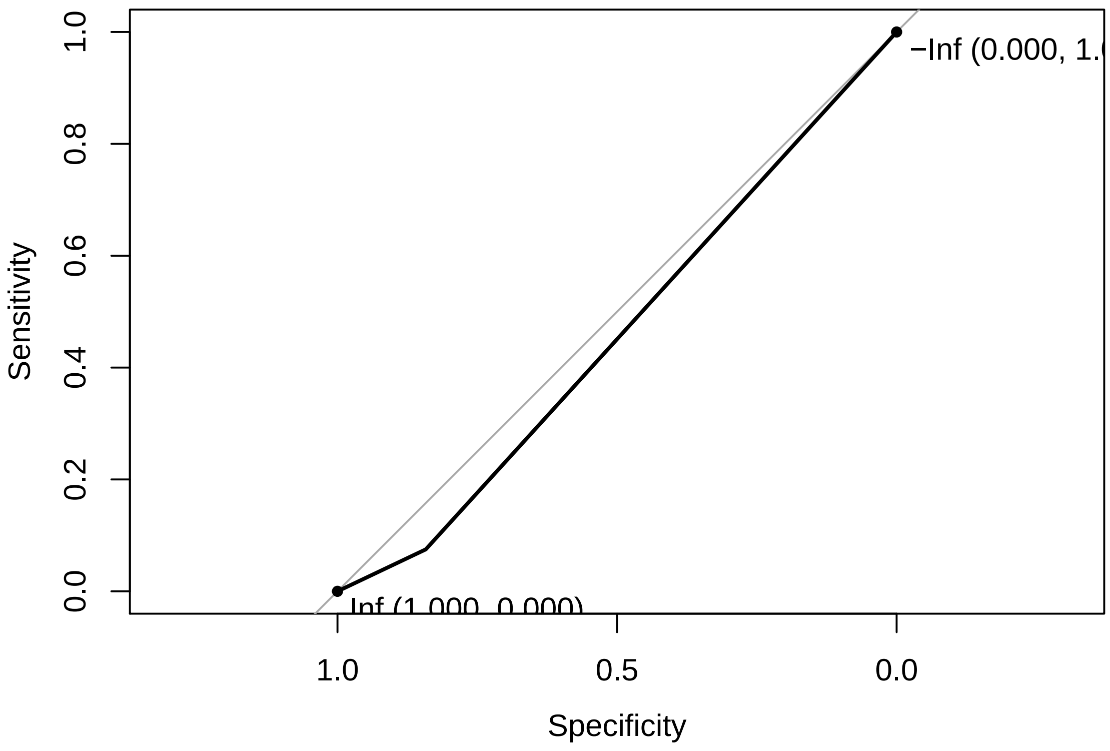
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  37  16
##        yes  3   3
##
##                Accuracy : 0.678
##                  95% CI : (0.544, 0.794)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.56176
##
##                   Kappa : 0.101
##
##  Mcnemar's Test P-Value : 0.00591
##
##             Sensitivity : 0.925
##             Specificity : 0.158
##          Pos Pred Value : 0.698
##          Neg Pred Value : 0.500
##              Prevalence : 0.678
##          Detection Rate : 0.627
##    Detection Prevalence : 0.898
##       Balanced Accuracy : 0.541
##
##        'Positive' Class : no
##
```



serum creatinine, ejection fraction and age are the most important

ROC curve for Support Vector Machine

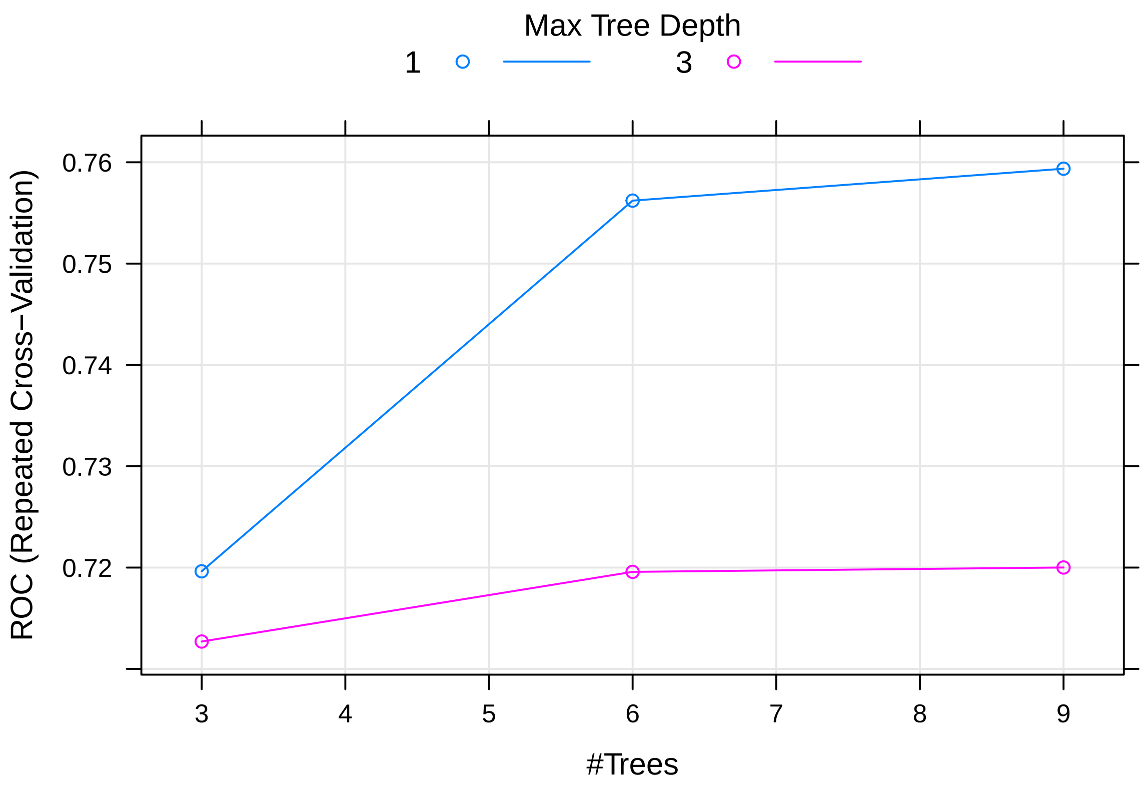




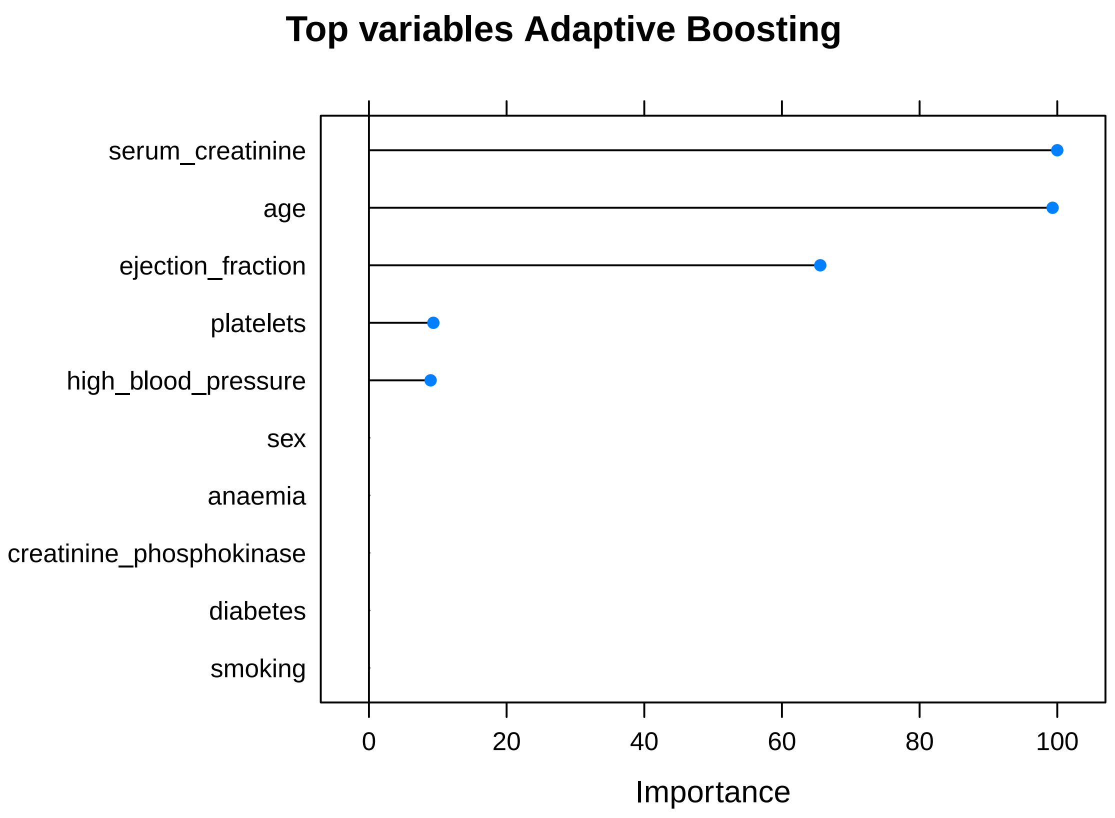
3.2 Adaptive Boosting

Tuning parameters:

1. mfinal (#Trees)
2. maxdepth (Max Tree Depth)
3. coeflearn (Coefficient Type)

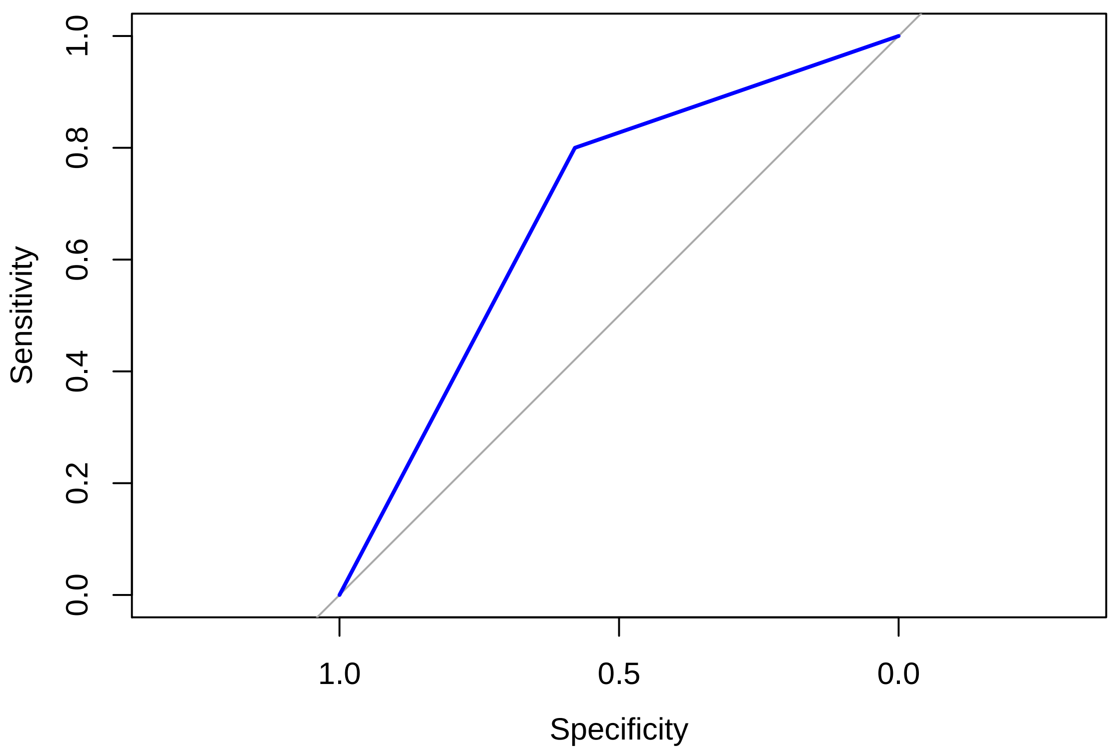


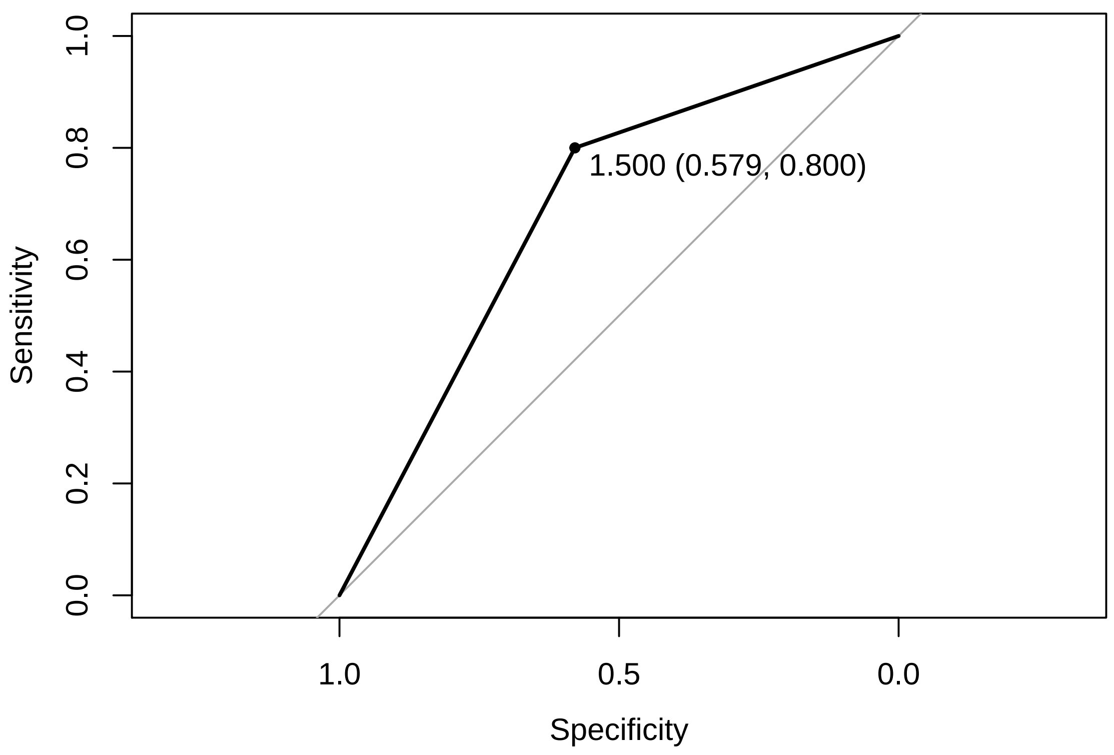
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  32   8
##        yes  8  11
##
##                Accuracy : 0.729
##                  95% CI : (0.597, 0.836)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.246
##
##                   Kappa : 0.379
##
##  Mcnemar's Test P-Value : 1.000
##
##             Sensitivity : 0.800
##             Specificity : 0.579
##          Pos Pred Value : 0.800
##          Neg Pred Value : 0.579
##              Prevalence : 0.678
##          Detection Rate : 0.542
##    Detection Prevalence : 0.678
##       Balanced Accuracy : 0.689
##
##        'Positive' Class : no
##
```



serum creatinine, age and ejection fraction are the most important

ROC curve for Adaptive Boosting

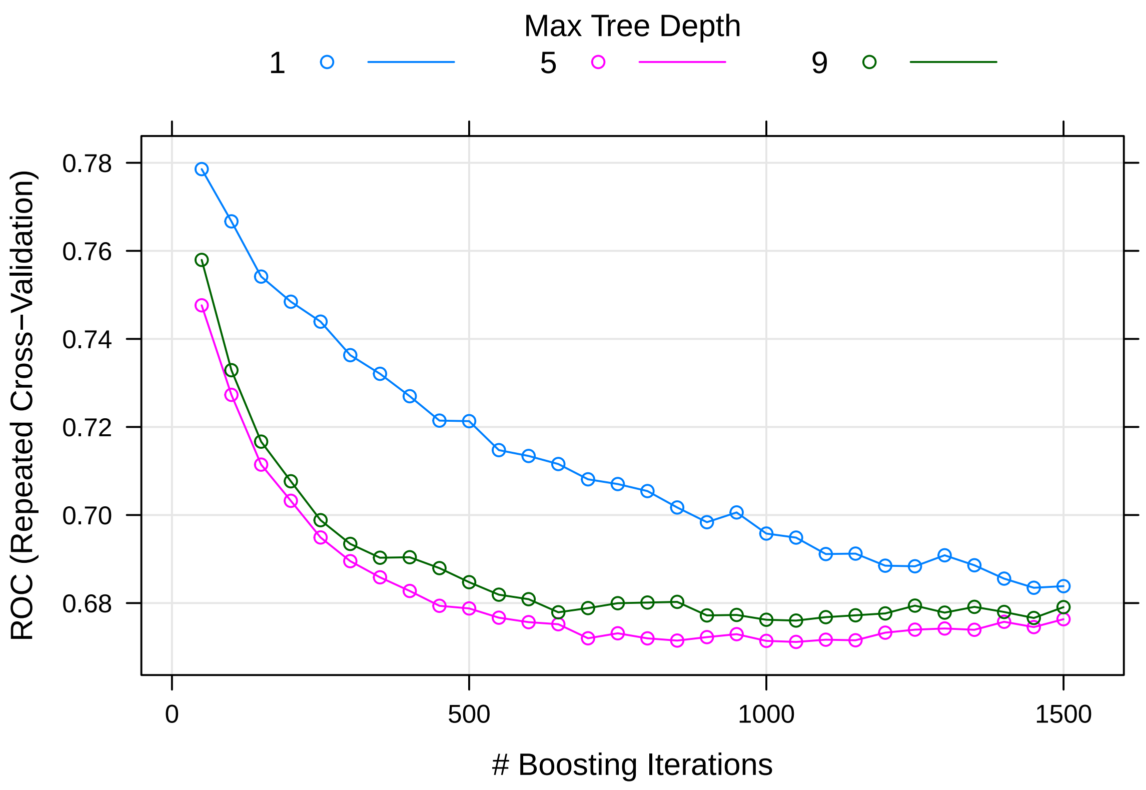




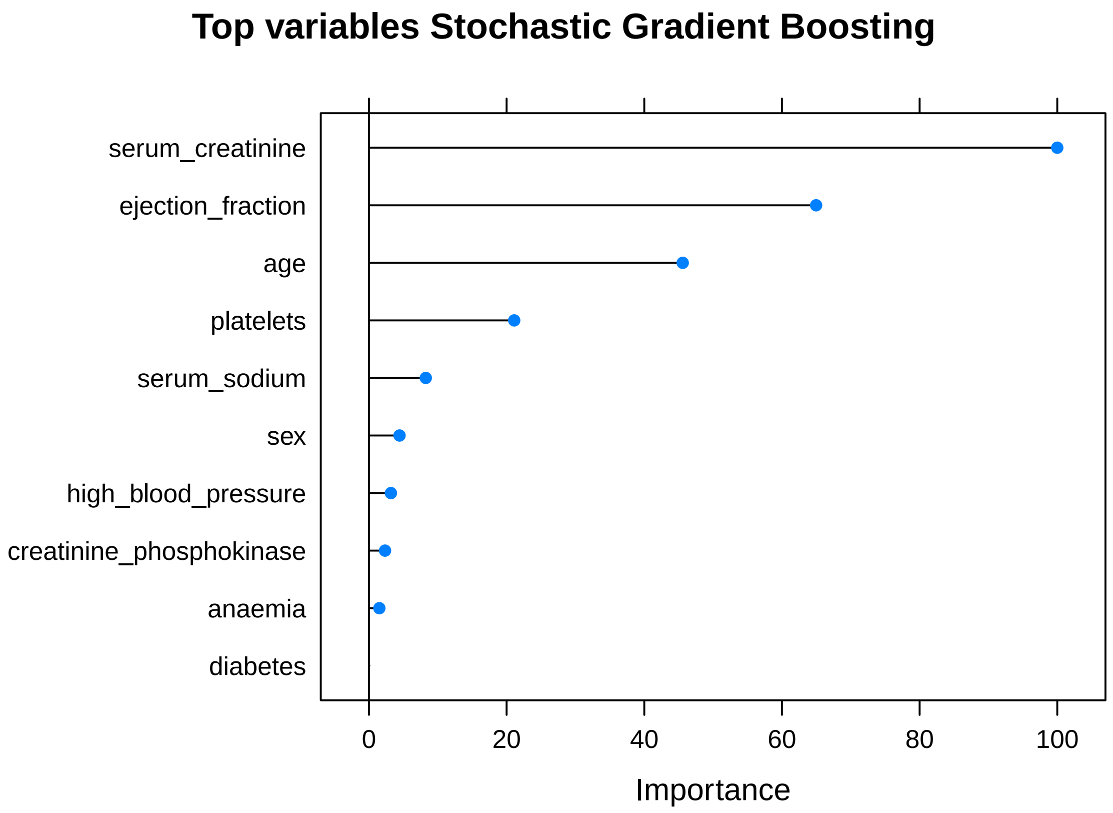
3.3 Stochastic Gradient Boosting

Tuning parameters:

1. interaction.depth (Max Tree Depth)
2. n.trees (Boosting Iterations)
3. shrinkage (Bandwidth Adjustment)
4. n.minobsinnode (Min. Terminal Node Size)

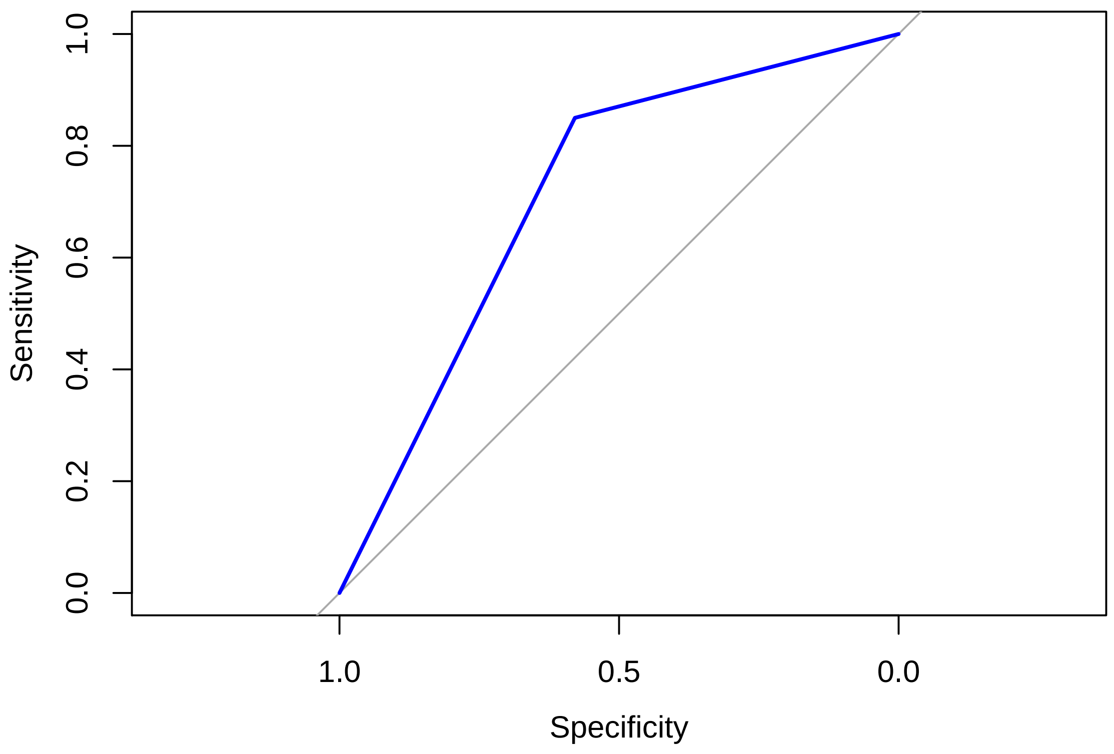


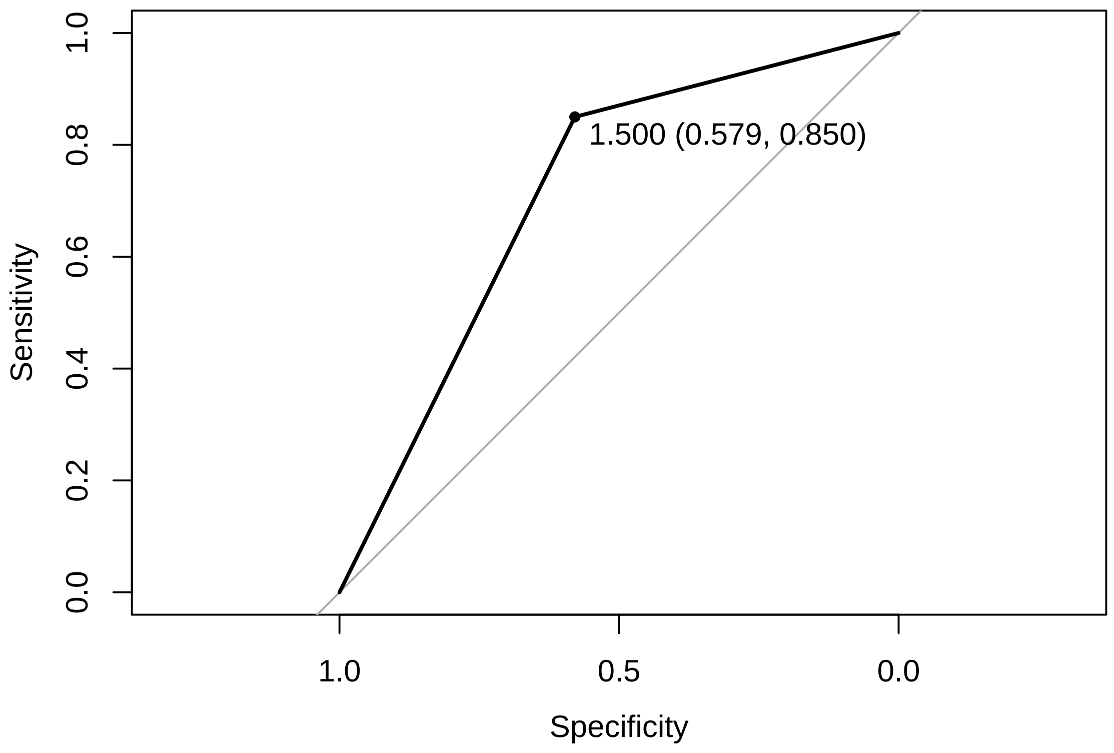
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  34   8
##        yes  6  11
##
##                Accuracy : 0.763
##                  95% CI : (0.634, 0.864)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.103
##
##                   Kappa : 0.441
##
##  Mcnemar's Test P-Value : 0.789
##
##             Sensitivity : 0.850
##             Specificity : 0.579
##          Pos Pred Value : 0.810
##          Neg Pred Value : 0.647
##              Prevalence : 0.678
##          Detection Rate : 0.576
##    Detection Prevalence : 0.712
##       Balanced Accuracy : 0.714
##
##        'Positive' Class : no
##
```



serum creatinine, ejection fraction and age are the most important

ROC curve for Stochastic Gradient Boosting

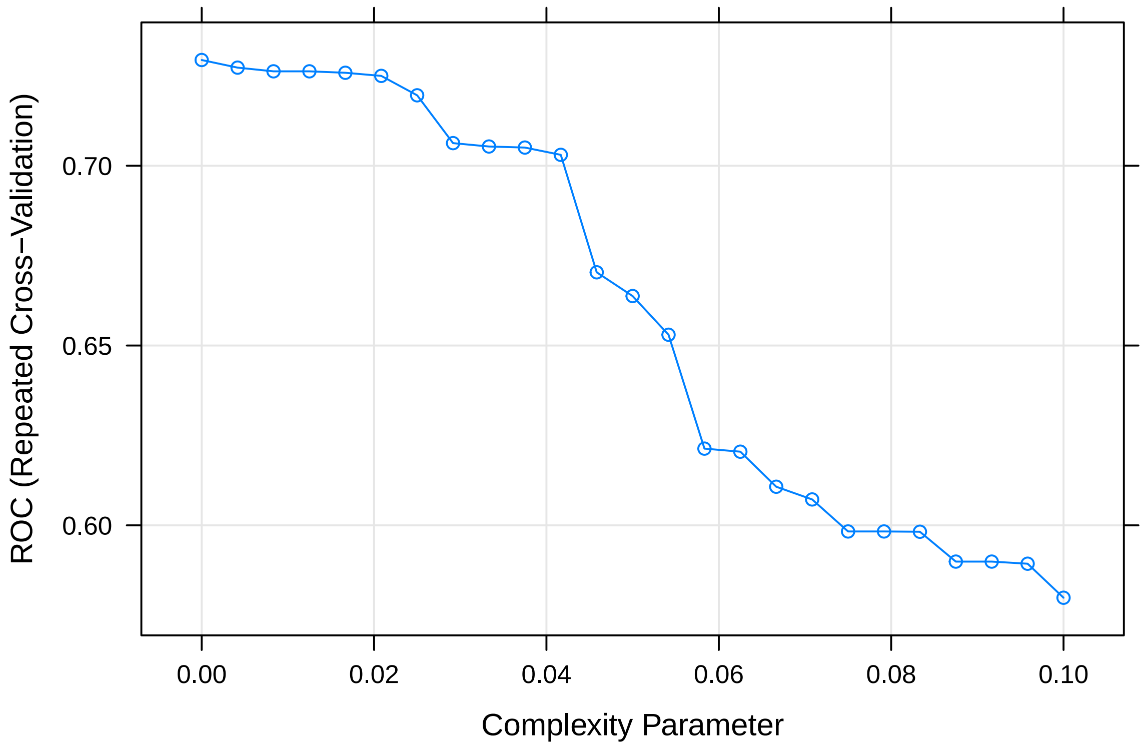




3.4 Classification Trees

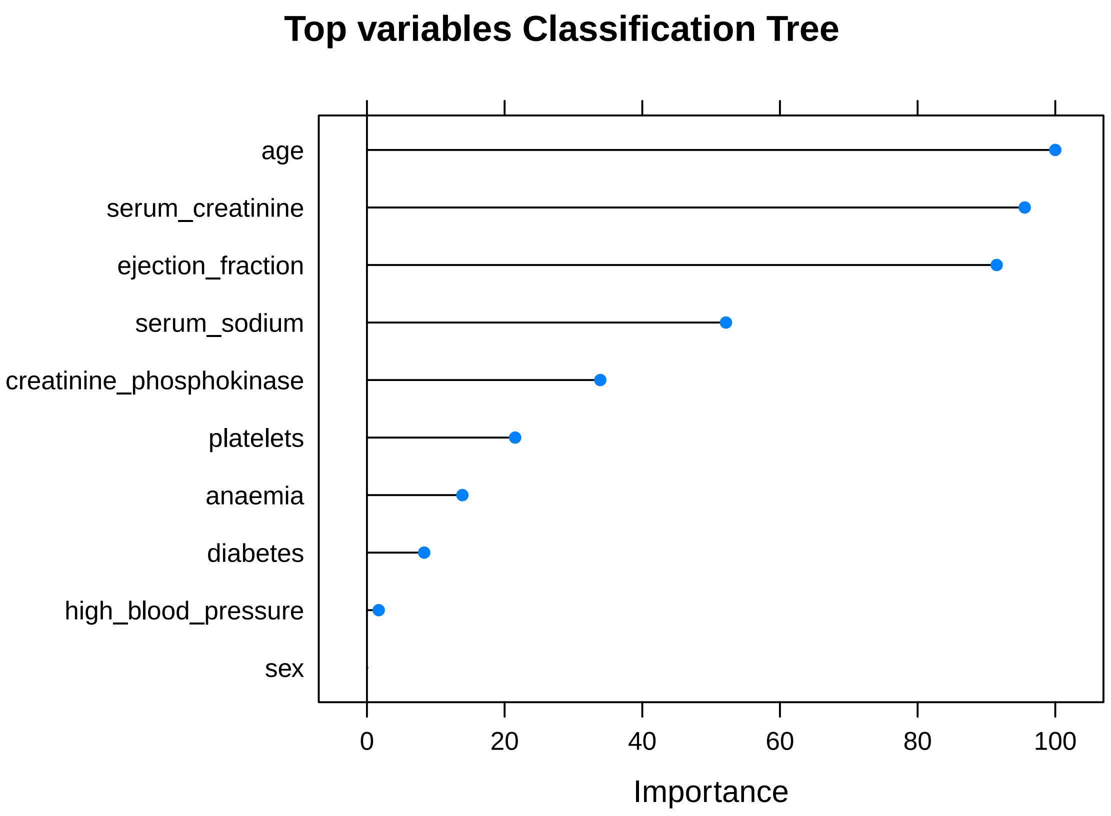
Tuning parameters:

1. cp (Complexity Parameter)



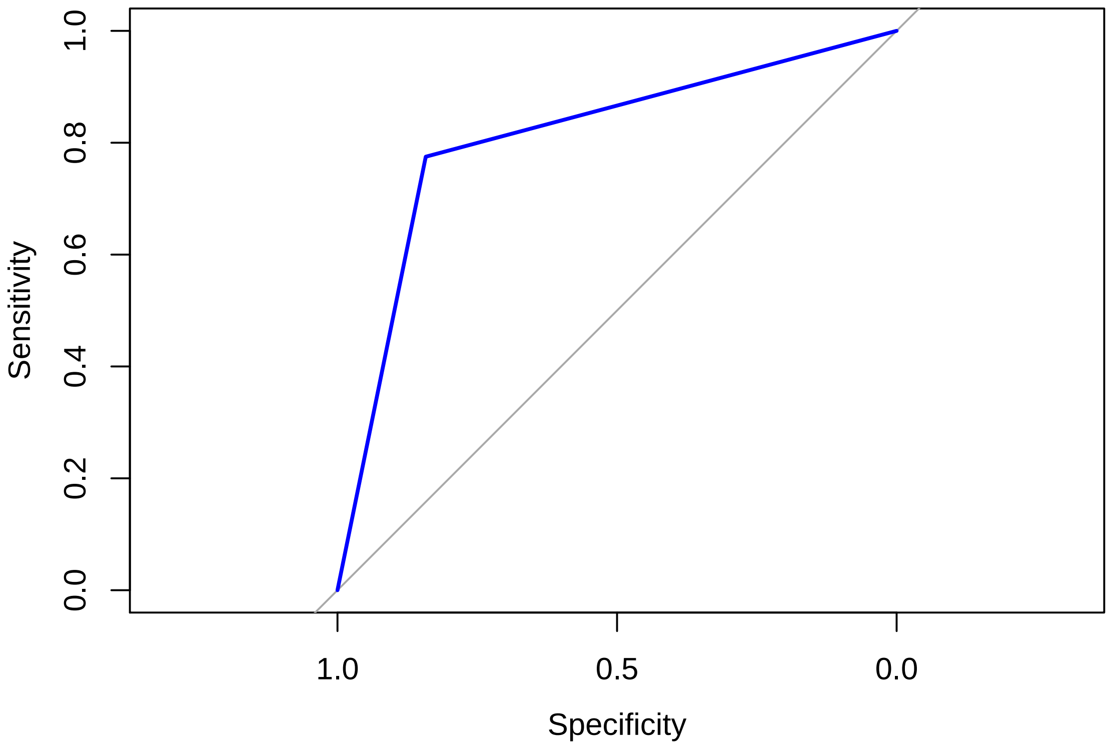
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  31   3
##        yes  9  16
##
##                Accuracy : 0.797
##                  95% CI : (0.672, 0.89)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.0314
##
##                   Kappa : 0.57
##
##  Mcnemar's Test P-Value : 0.1489
##
##             Sensitivity : 0.775
##             Specificity : 0.842
##          Pos Pred Value : 0.912
##          Neg Pred Value : 0.640
##              Prevalence : 0.678
##          Detection Rate : 0.525
##    Detection Prevalence : 0.576
##       Balanced Accuracy : 0.809
```

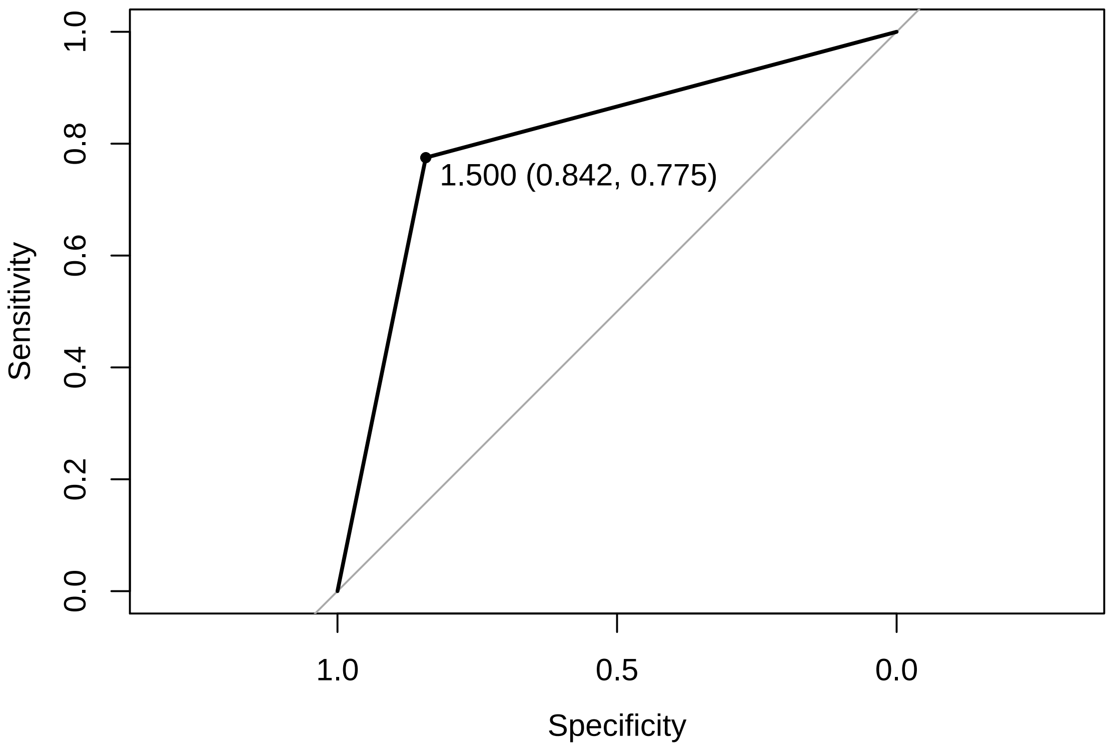
```
##
##        'Positive' Class : no
##
```



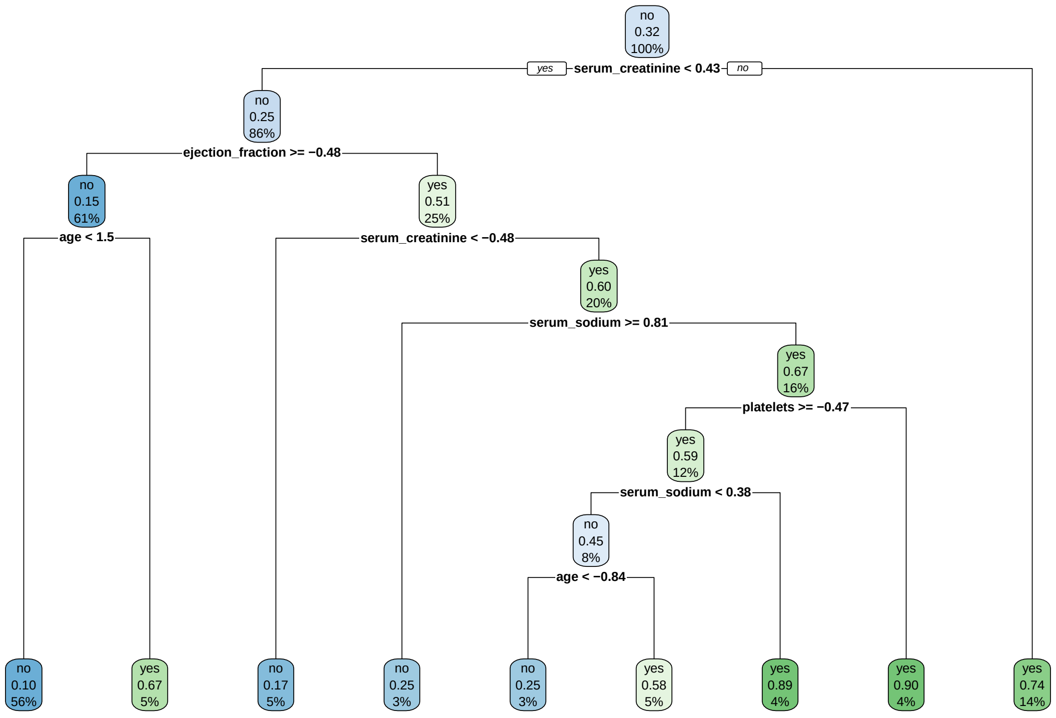
age, serum creatinine and ejection fraction are the most important

ROC curve for Classification Trees





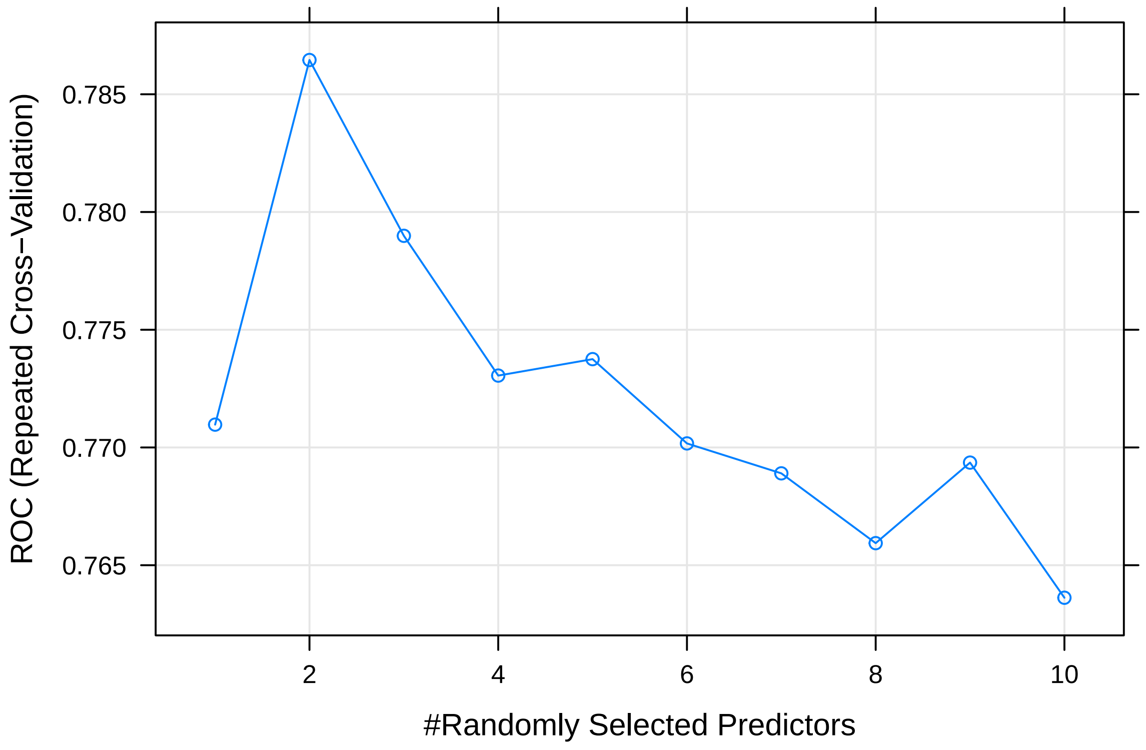
The graph below shows the decision tree


no
0.32
100%
yes
serum_creatinine < 0.43
no
no
0.25
86%
ejection_fraction >= −0.48
no
0.15
61%
age < 1.5
yes
0.51
25%
serum_creatinine < −0.48
yes
0.60
20%
serum_sodium >= 0.81
yes
0.67
16%
platelets >= −0.47
yes
0.59
12%
serum_sodium < 0.38
no
0.45
8%
age < −0.84
no
0.10
56%
yes
0.67
5%
no
0.17
5%
no
0.25
3%
no
0.25
3%
yes
0.58
5%
yes
0.89
4%
yes
0.90
4%
yes
0.74
14%

3.5 Random Forest

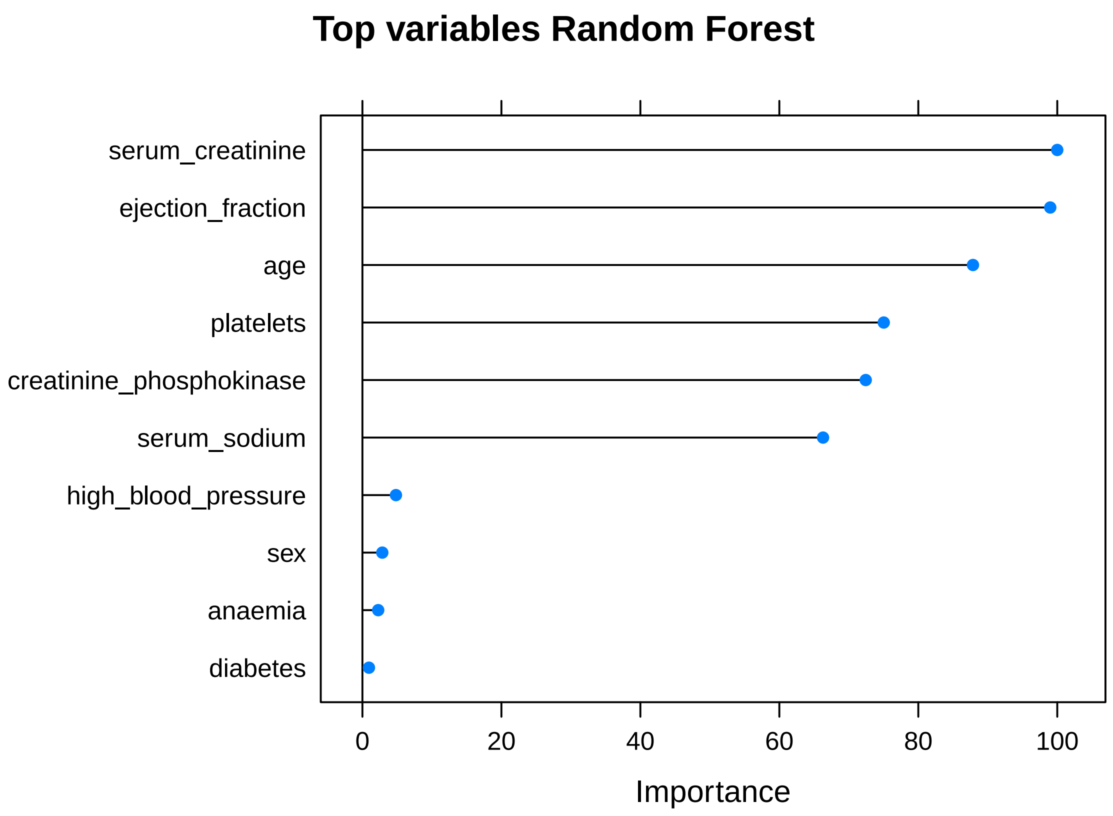
Tuning parameters:

1. mtry (#Randomly Selected Predictors)



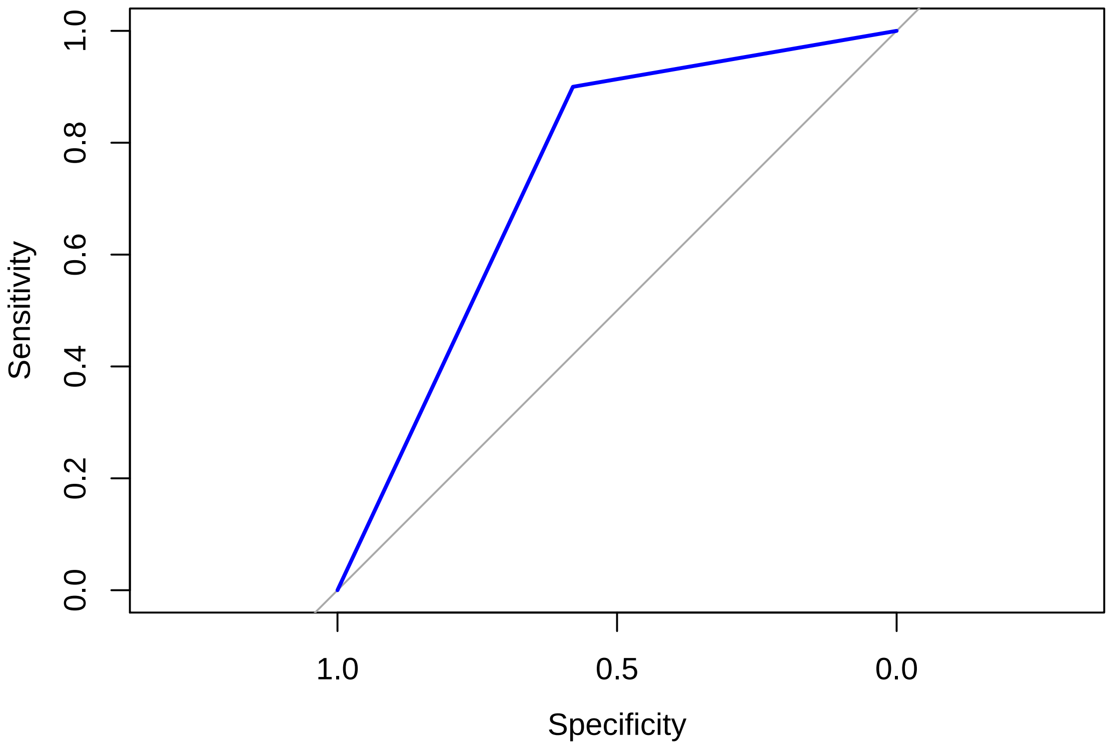
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  36   8
##        yes  4  11
##
##                Accuracy : 0.797
##                  95% CI : (0.672, 0.89)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.0314
##
##                   Kappa : 0.507
##
##  Mcnemar's Test P-Value : 0.3865
##
##             Sensitivity : 0.900
##             Specificity : 0.579
##          Pos Pred Value : 0.818
##          Neg Pred Value : 0.733
##              Prevalence : 0.678
##          Detection Rate : 0.610
##    Detection Prevalence : 0.746
##       Balanced Accuracy : 0.739
```

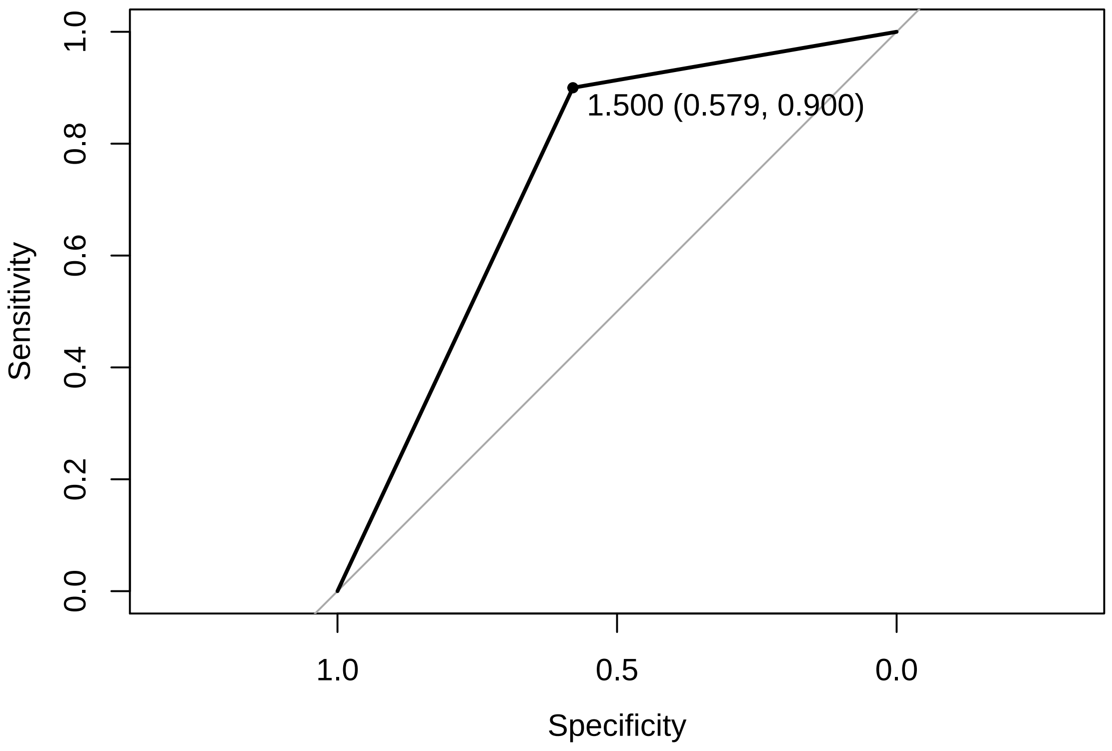
```
##
##       'Positive' Class : no
##
```



serum creatinine, ejection fraction and age are the most important

ROC curve for Random Forest

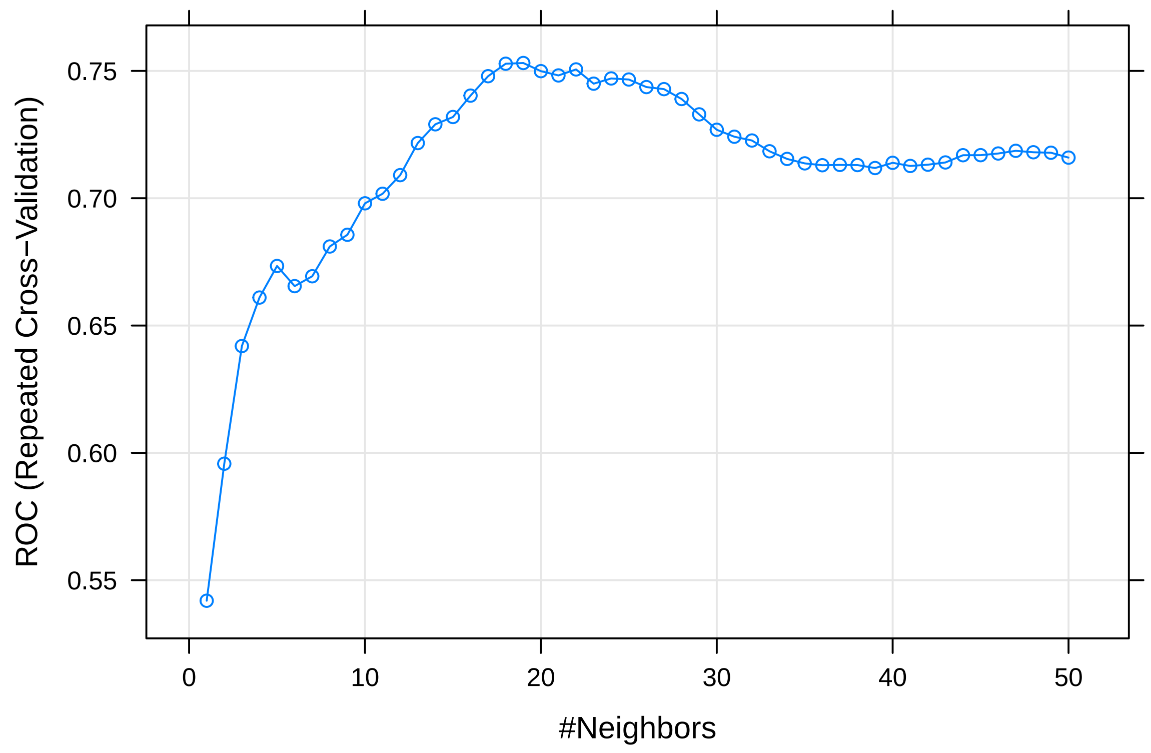




3.6 K Nearest Neighbor (KNN)

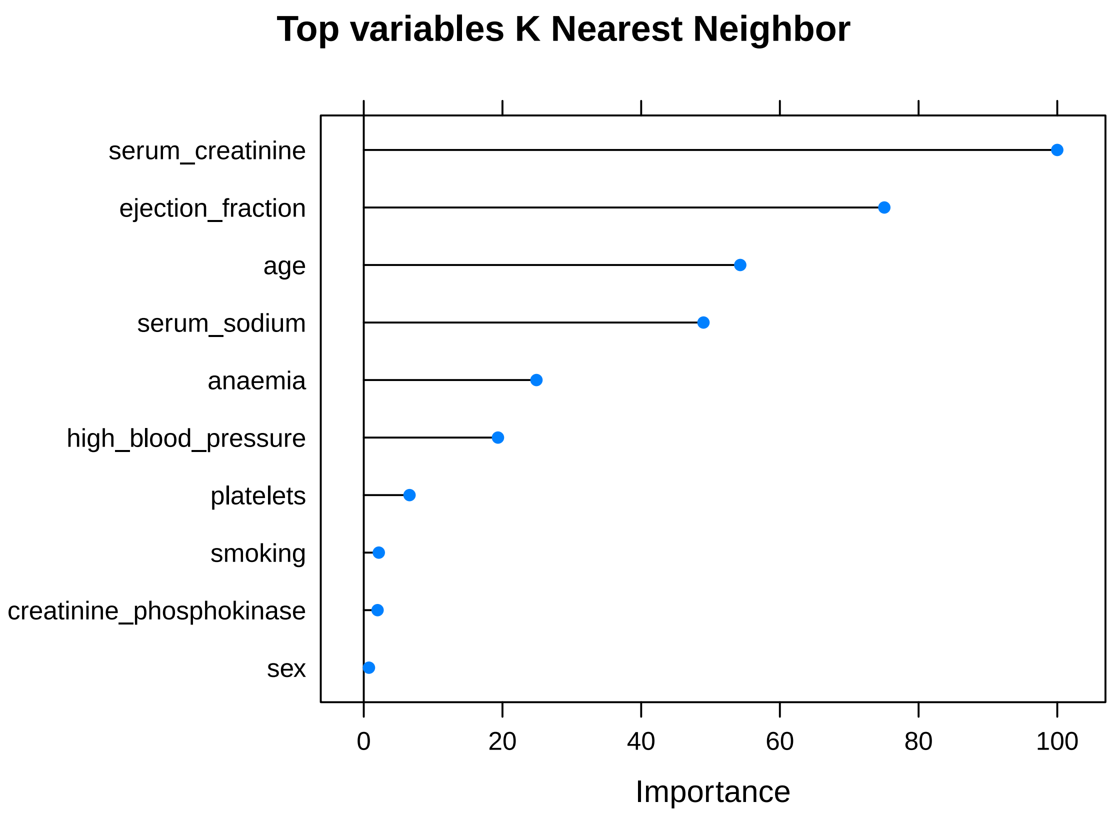
Tuning parameters:

1. k (#Neighbors)



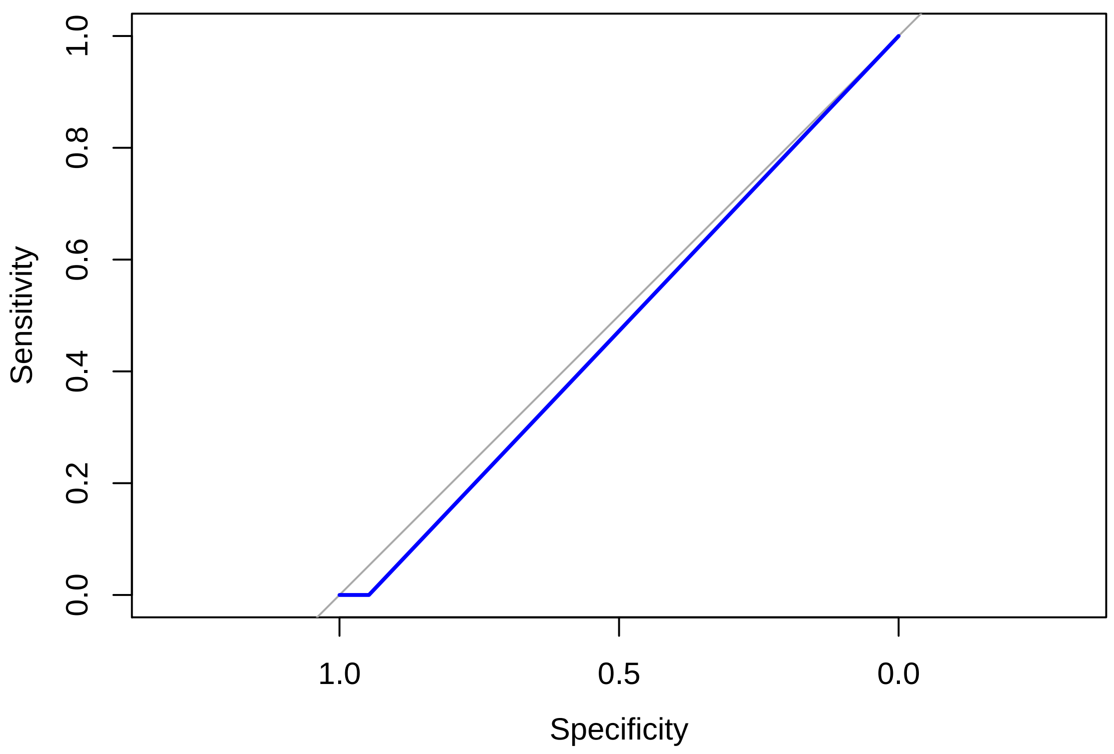
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  40  18
##        yes  0   1
##
##                Accuracy : 0.695
##                  95% CI : (0.561, 0.808)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.451
##
##                   Kappa : 0.07
##
##  Mcnemar's Test P-Value : 6.15e-05
##
##             Sensitivity : 1.0000
##             Specificity : 0.0526
##          Pos Pred Value : 0.6897
##          Neg Pred Value : 1.0000
##              Prevalence : 0.6780
##          Detection Rate : 0.6780
##    Detection Prevalence : 0.9831
##       Balanced Accuracy : 0.5263
```

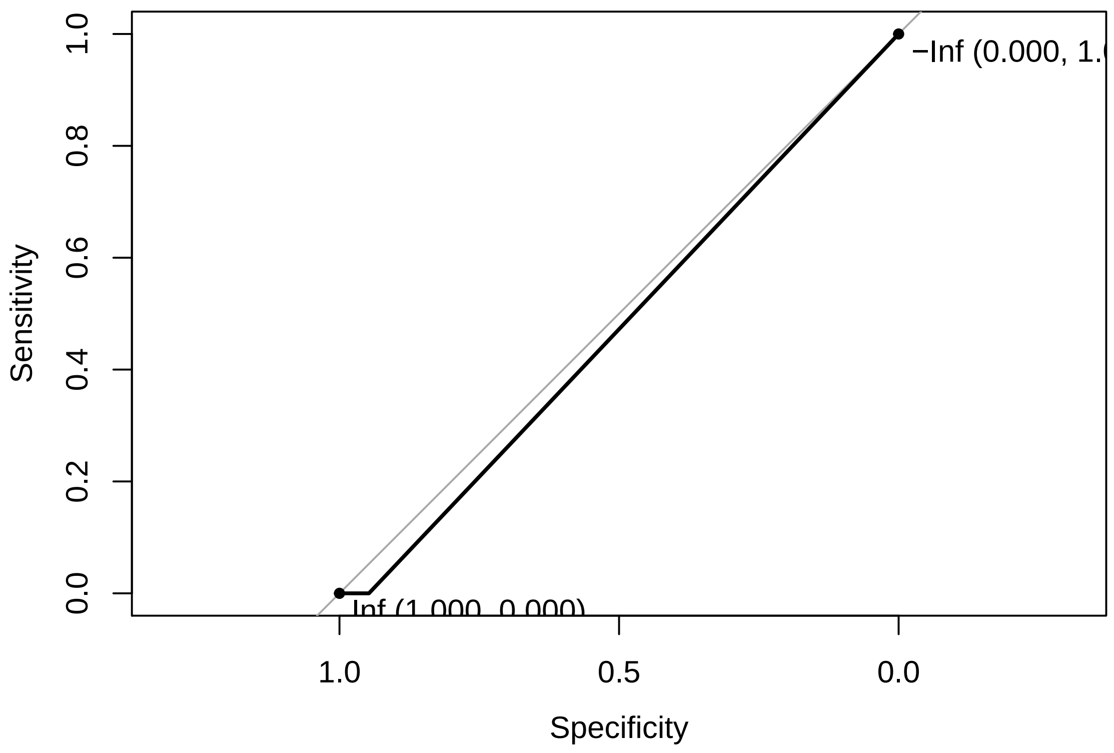
```
##
##        'Positive' Class : no
##
```



serum creatinine, ejection fraction and age are the most important

ROC curve for K Nearest Neighbor

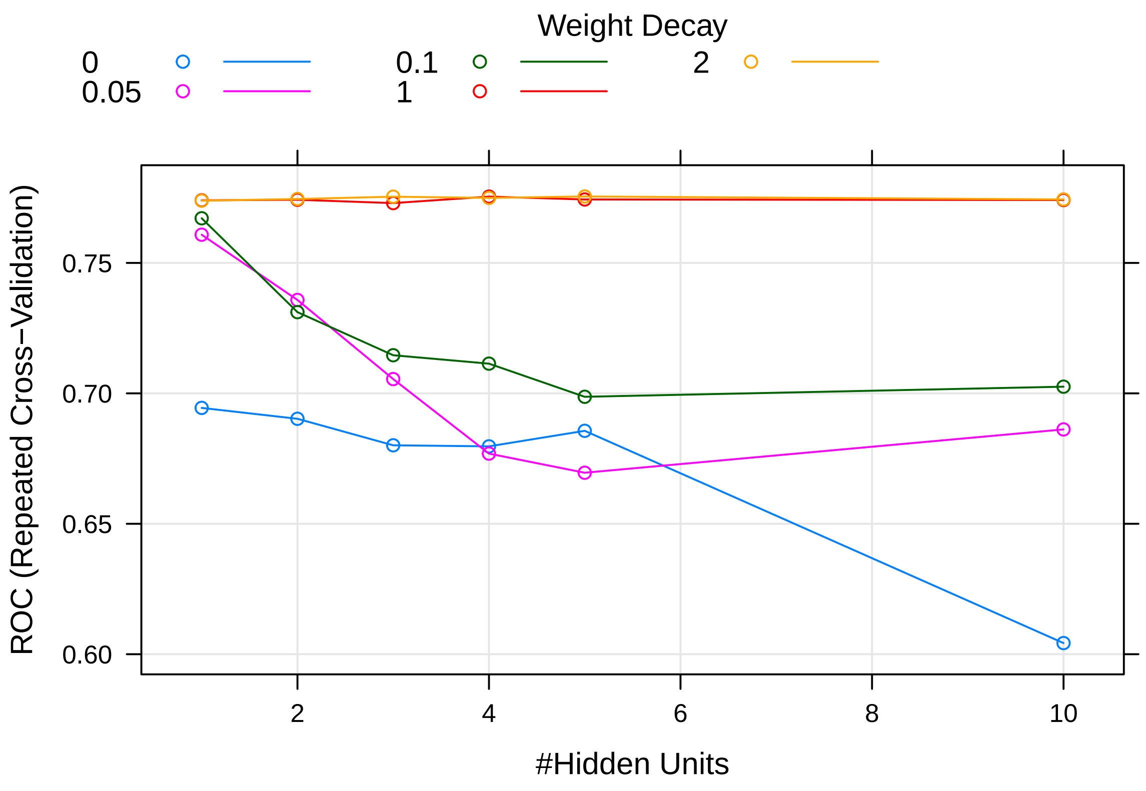




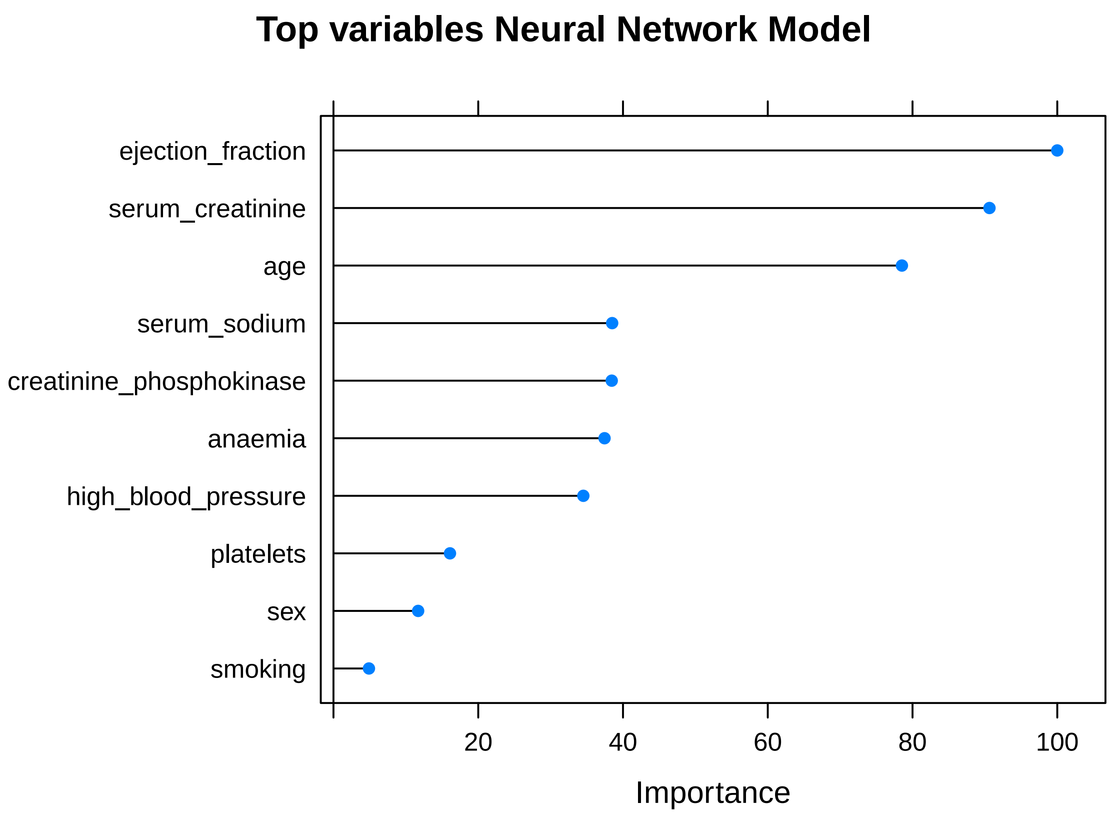
3.7 Neural Network

Tuning parameters:

1. size (#Hidden Units)
2. decay (Weight Decay)

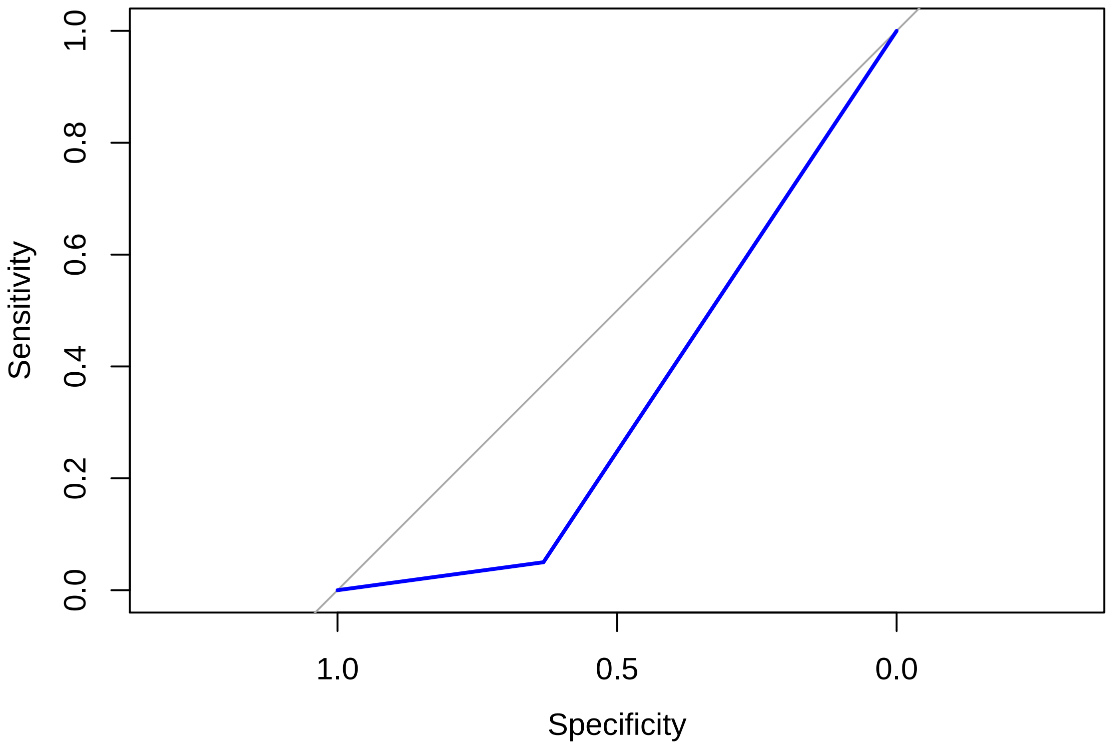


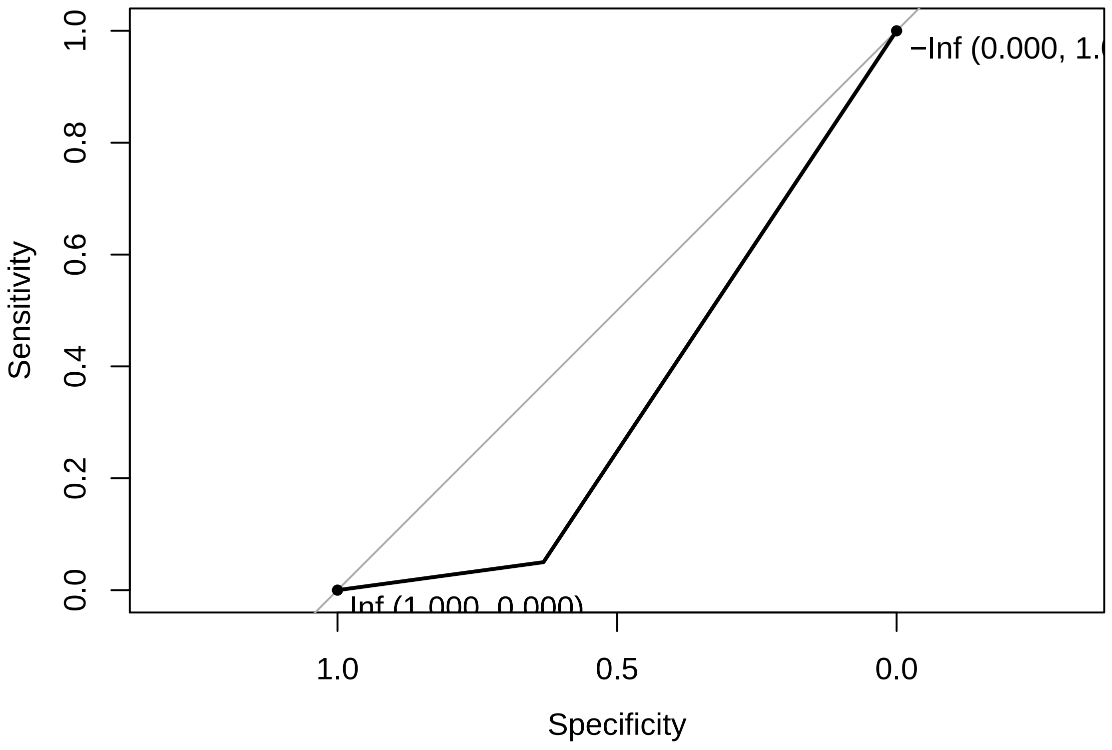
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  38  12
##        yes  2   7
##
##                Accuracy : 0.763
##                  95% CI : (0.634, 0.864)
##     No Information Rate : 0.678
##     P-Value [Acc > NIR] : 0.1029
##
##                   Kappa : 0.369
##
##  Mcnemar's Test P-Value : 0.0162
##
##             Sensitivity : 0.950
##             Specificity : 0.368
##          Pos Pred Value : 0.760
##          Neg Pred Value : 0.778
##              Prevalence : 0.678
##          Detection Rate : 0.644
##    Detection Prevalence : 0.847
##        Balanced Accuracy : 0.659
##
##         'Positive' Class : no
##
```



ejection fraction, serum creatinine and age are the most important

ROC curve for Neural Network





3.8 Compare algorithms

In this last phase, are compared the results produced by the various algorithms

```
##
## Call:
## summary.resamples(object = models_results)
##
## Models: SVM, ADP_Boost, Gradient_Bost, Class_Tree, RND_Forest, KNN, NN
## Number of resamples: 100
##
## ROC
##                Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
## SVM           0.398   0.588  0.646 0.659   0.750 0.955    0
## ADP_Boost     0.520   0.691  0.773 0.759   0.825 0.961    0
## Gradient_Bost 0.589   0.722  0.780 0.779   0.836 0.963    0
## Class_Tree    0.529   0.656  0.725 0.729   0.801 0.996    0
## RND_Forest    0.550   0.724  0.785 0.786   0.851 0.963    0
## KNN           0.500   0.699  0.773 0.753   0.824 0.934    0
## NN            0.492   0.725  0.775 0.775   0.844 0.971    0
##
## Sens
##                Min. 1st Qu. Median  Mean 3rd Qu. Max. NA's
## SVM           0.750   0.924  0.941 0.944   1.000    1    0
## ADP_Boost     0.647   0.824  0.875 0.876   0.938    1    0
## Gradient_Bost 0.647   0.824  0.875 0.883   0.938    1    0
## Class_Tree    0.562   0.750  0.824 0.814   0.875    1    0
## RND_Forest    0.647   0.824  0.882 0.883   0.938    1    0
## KNN           0.875   0.941  1.000 0.980   1.000    1    0
## NN            0.765   0.875  0.938 0.937   1.000    1    0
##
## Spec
##                Min. 1st Qu. Median   Mean 3rd Qu.  Max. NA's
## SVM           0.000   0.000  0.000 0.0716   0.125 0.500    0
## ADP_Boost     0.000   0.250  0.375 0.3968   0.500 0.875    0
## Gradient_Bost 0.000   0.286  0.375 0.4288   0.571 0.857    0
## Class_Tree    0.125   0.375  0.500 0.5139   0.625 1.000    0
## RND_Forest    0.125   0.250  0.375 0.4046   0.500 0.875    0
## KNN           0.000   0.000  0.125 0.0882   0.129 0.286    0
## NN            0.000   0.250  0.286 0.3048   0.429 0.714    0
```

	SVM	ADP_Boost	Gradient_Bost	Class_Tree	RND_Forest	KNN	NN
Sensitivity	0.925	0.800	0.850	0.775	0.900	1.000	0.950
Specificity	0.158	0.579	0.579	0.842	0.579	0.053	0.368
Pos Pred Value	0.698	0.800	0.810	0.912	0.818	0.690	0.760
Neg Pred Value	0.500	0.579	0.647	0.640	0.733	1.000	0.778
Precision	0.698	0.800	0.810	0.912	0.818	0.690	0.760
Recall	0.925	0.800	0.850	0.775	0.900	1.000	0.950
F1	0.796	0.800	0.829	0.838	0.857	0.816	0.844
Prevalence	0.678	0.678	0.678	0.678	0.678	0.678	0.678
Detection Rate	0.627	0.542	0.576	0.525	0.610	0.678	0.644
Detection Prevalence	0.898	0.678	0.712	0.576	0.746	0.983	0.847
Balanced Accuracy	0.541	0.689	0.714	0.809	0.739	0.526	0.659

K Nearest Neighbor algorithm has been identified as the best one for ***sensitivity***, instead ***Classification Tree*** is the best for ***specificity***

We use again the classification model, but considering only serum creatinine, ejection fraction and age as variables

```
##
## Call:
## summary.resamples(object = models_results)
##
## Models: SVM, ADP_Boost, Gradient_Bost, Class_Tree, RND_Forest, KNN, NN
## Number of resamples: 100
##
## ROC
##                Min. 1st Qu. Median  Mean 3rd Qu.  Max. NA's
## SVM           0.422   0.661  0.730 0.730   0.807 0.971    0
## ADP_Boost     0.441   0.654  0.754 0.731   0.807 0.941    0
## Gradient_Bost 0.402   0.655  0.748 0.737   0.820 0.956    0
## Class_Tree    0.368   0.629  0.706 0.701   0.774 0.974    0
## RND_Forest    0.522   0.663  0.737 0.729   0.802 0.963    0
## KNN           0.520   0.680  0.746 0.747   0.819 0.989    0
## NN            0.420   0.670  0.739 0.738   0.821 0.912    0
##
## Sens
##                Min. 1st Qu. Median  Mean 3rd Qu. Max. NA's
## SVM           0.647   0.812  0.875 0.877   0.938    1    0
## ADP_Boost     0.625   0.824  0.882 0.893   0.941    1    0
## Gradient_Bost 0.688   0.812  0.875 0.872   0.938    1    0
## Class_Tree    0.625   0.812  0.875 0.865   0.938    1    0
## RND_Forest    0.588   0.812  0.824 0.837   0.896    1    0
## KNN           0.812   0.938  0.941 0.955   1.000    1    0
## NN            0.824   1.000  1.000 0.987   1.000    1    0
##
## Spec
##                Min. 1st Qu. Median   Mean 3rd Qu.  Max. NA's
## SVM           0.125   0.286  0.375 0.4055   0.500 0.875    0
## ADP_Boost     0.000   0.250  0.375 0.3698   0.500 0.857    0
## Gradient_Bost 0.000   0.286  0.429 0.4214   0.571 0.875    0
## Class_Tree    0.000   0.286  0.429 0.4477   0.571 1.000    0
## RND_Forest    0.000   0.277  0.375 0.3934   0.500 0.875    0
## KNN           0.000   0.125  0.250 0.2300   0.286 0.625    0
## NN            0.000   0.000  0.000 0.0864   0.143 0.500    0
```

	SVM	ADP_Boost	Gradient_Bost	Class_Tree	RND_Forest	KNN	NN
Sensitivity	0.875	0.875	0.900	0.900	0.950	1.000	1.000
Specificity	0.474	0.632	0.316	0.474	0.368	0.316	0.158
Pos Pred Value	0.778	0.833	0.735	0.783	0.760	0.755	0.714
Neg Pred Value	0.643	0.706	0.600	0.692	0.778	1.000	1.000
Precision	0.778	0.833	0.735	0.783	0.760	0.755	0.714
Recall	0.875	0.875	0.900	0.900	0.950	1.000	1.000
F1	0.824	0.854	0.809	0.837	0.844	0.860	0.833
Prevalence	0.678	0.678	0.678	0.678	0.678	0.678	0.678
Detection Rate	0.593	0.593	0.610	0.610	0.644	0.678	0.678
Detection Prevalence	0.763	0.712	0.831	0.780	0.847	0.898	0.949
Balanced Accuracy	0.674	0.753	0.608	0.687	0.659	0.658	0.579

K Nearest Neighbor and ***Neural Network*** algorithms are the best one for ***sensitivity***, instead ***Adaptive Boosting*** is the best for ***specificity*** With only three variables the ***sensitivity*** doesn't change. Instead for ***specificity*** we have lower performance

4 Conclusion

The objective of this project was to analyze the Heart failure dataset to build a classifiers to predict the heart failure and what are the most important variables. In particular was compared the ***classification*** capacity of the algorithms, with particular regard to the topic of ***specificity*** and ***sensitivity***.

We started by analyzing the dataset to understand the structure of the data and in particular was analyzed the correlation between predictors and the possibility of reducing their number

We then selected ROC as the metric to compare algorithms

We identified the algorithms to be implemented to create the system and then evaluated their quality in terms of ***sensitivity*** and ***specificity***.

For each algorithm we evaluate the most important predictors.

K Nearest Neighbor algorithm has been identified as the best one for ***sensitivity***, instead ***Classification Tree*** is the best performance for ***specificity***.

The most important variables are: serum creatinine, ejection fraction and age.

We then repeated the construction of the classification model using a dataset reduced to its three most important variables.

K Nearest Neighbor and ***Neural Network*** algorithms are the best one for ***sensitivity***, instead ***Adaptive Boosting*** is the best performance for ***specificity***.

With only three variables the ***sensitivity*** doesn't change. Instead for ***specificity*** we have lower performance.

The present project does not represent an exhaustive analysis, to obtain this result should be considered the possible variants of the implemented algorithms, as well as the opportunity to implement other algorithms among those available for classification problems.

Last but not least, it would be possible to use a dataset with many more observations, as the dataset used contains only 299 observations.