MLSDM Coursework 1

Simone Zanetti

Introduction

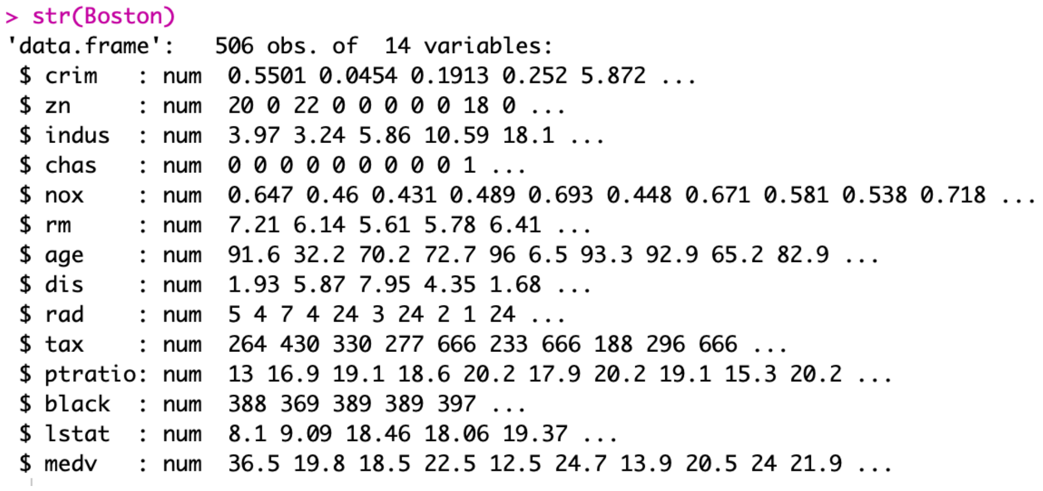
The following report deals with the attempt to summarise the process of obtainment of the tasks required by the coursework 1 of the Module '*Machine Learning and Statistical Data Mining*'. In particular, through the phases which are described in the following sections, I have attempted to obtain consistent models which could satisfy the need of predicting a quantitative variable (regression task), and the one of predicting a qualitative one with two categories (binary classification task).
For the following analysis a dataset related to the housing values in the suburbs of Boston is taken in consideration. This data have been collected by the U.S Census Service concerning housing in the area of Boston Mass, and they are used in this analysis with two purposes:

1. **Regression Task:** predict the median house value (variable *medv* of the dataset, measured in $1000s)

2. **Classification Task**: predict if a suburb has a crime rate below or above the median (variable *crim*[1] on the dataset)

Analysis of the dataset and Pre-processing

The first step in each project which involves the analysis of a dataset regards the necessity to verify the structure of it, and eventually provide the necessary modifications to make it more feasible for the investigation. In this specific case, the dataset which contains 14 variables and 506 rows, does not seem to suffer any lack which would lead to apply modifications. In fact, no *missing values* are identified, all the variables are numeric and do not imply the need to be encoded[2], and in general the data is tidy. To conclude, before to move on with the analysis, the data are split into two halves, respectively the *train set*, in which the models will be built, and the *test set* in which their performance will be verified.



```
> str(Boston)
'data.frame':	506 obs. of  14 variables:
 $ crim   : num  0.5501 0.0454 0.1913 0.252 5.872 ...
 $ zn     : num  20 0 22 0 0 0 0 0 18 0 ...
 $ indus  : num  3.97 3.24 5.86 10.59 18.1 ...
 $ chas   : num  0 0 0 0 0 0 0 0 0 1 ...
 $ nox    : num  0.647 0.46 0.431 0.489 0.693 0.448 0.671 0.581 0.538 0.718 ...
 $ rm     : num  7.21 6.14 5.61 5.78 6.41 ...
 $ age    : num  91.6 32.2 70.2 72.7 96 6.5 93.3 92.9 65.2 82.9 ...
 $ dis    : num  1.93 5.87 7.95 4.35 1.68 ...
 $ rad    : num  5 4 7 4 24 3 24 2 1 24 ...
 $ tax    : num  264 430 330 277 666 233 666 188 296 666 ...
 $ ptratio: num  13 16.9 19.1 18.6 20.2 17.9 20.2 19.1 15.3 20.2 ...
 $ black  : num  388 369 389 389 397 ...
 $ lstat  : num  8.1 9.09 18.46 18.06 19.37 ...
 $ medv   : num  36.5 19.8 18.5 22.5 12.5 24.7 13.9 20.5 24 21.9 ...
```

[1] For the purposes of the classification task, the variable will be turned into a dummy variable assuming values 0 and 1, called cr01

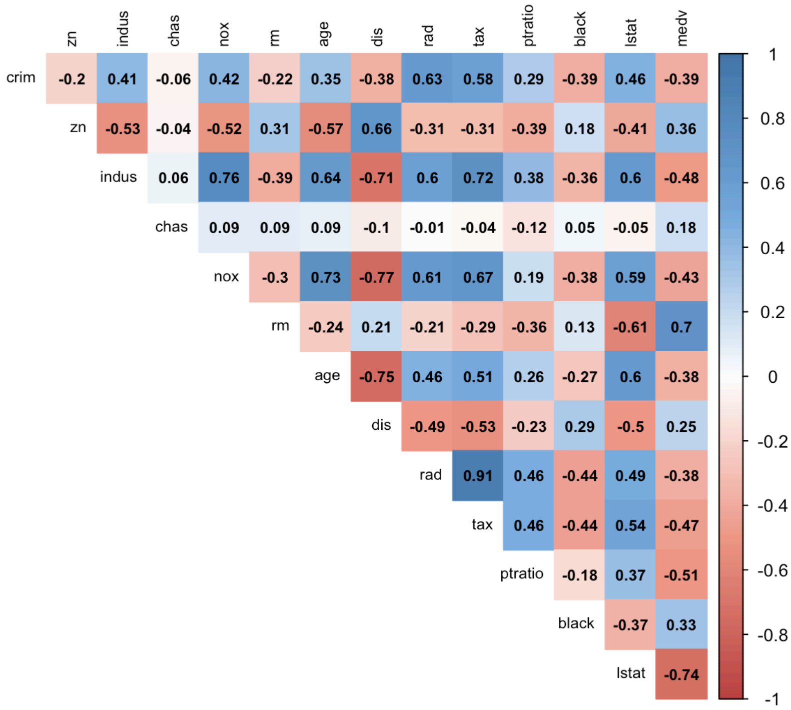
[2] *chas* has already been encoded, and the variable crim will be dummified when there will be the necessity.

REGRESSION TASK

Multiple Linear Regression

The first step in the process of definition of a Linear Regression model regarded the necessity to verify the correlation between the variables of the dataset. This process is important in order to be able to monitor and prevent the risk of *collinearity* between the predictors, which can weaken the model. In fact, the presence of collinearity can lead to the difficulty to separate the individual effects which the involved variables have for the response, with the risk of decreasing the quality of the model and/or misinterpret some of its data.

The heat map illustrated allows to observe the phenomenon of collinearity involving the different predictors, and the one related to the response variable *medv.*



	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
crim	-0.2	0.41	-0.06	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39
zn		-0.53	-0.04	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36
indus			0.06	0.76	-0.39	0.64	-0.71	0.6	0.72	0.38	-0.36	0.6	-0.48
chas				0.09	0.09	0.09	-0.1	-0.01	-0.04	-0.12	0.05	-0.05	0.18
nox					-0.3	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43
rm						-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.7
age							-0.75	0.46	0.51	0.26	-0.27	0.6	-0.38
dis								-0.49	-0.53	-0.23	0.29	-0.5	0.25
rad									0.91	0.46	-0.44	0.49	-0.38
tax										0.46	-0.44	0.54	-0.47
ptratio											-0.18	0.37	-0.51
black												-0.37	0.33
lstat													-0.74

✦ 2 Observations:

. The features RAD, TAX have a correlation of 0.91. DIS and AGE have a correlation of -0.75. These are just two examples of strongly correlated variables, which should not be selected together for training the model.

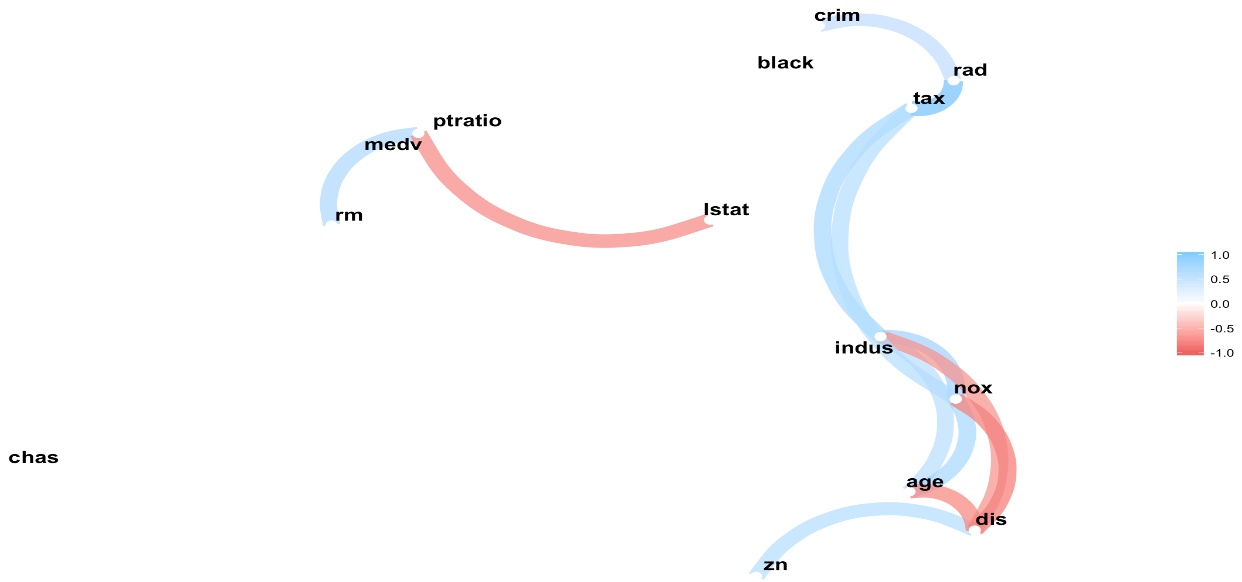
. RM has a strong positive correlation with MEDV (0.7) while as LSTAT has a high negative correlation with MEDV(-0.74)

STEP 1: Feature Selection

The first step in the process of building of the Linear regression regarded the necessity to define the quantity of predictors which would help the model to be assessed in its best way. To be fair, this phase was indeed only the second part of a process which started by verifying the presence of at least one variable related to response. In order to do so, a model containing all the predictors was build, and in the presence of a F-Statistics equal to 51.2 and a p-value: < 2.2e-16, the value expected to attest the Null Hypothesis could be considered largely overcome, confirming the confidence to assume that at least one variable was significant in the model.

In the context of the feature selection a series of attempts have been made. Specifically, the first attempt moved into the direction of excluding the predictors on the basis of the collinearity issues verified in the heat map above. For the occasion, a very intuitive graph[3] has been created into *R* and taken into consideration.

1.A. Manual feature selection: avoiding collinearity



Despite the importance of selecting predictors that can effectively have a predictive power and contribute to the model, which is translated in the selection of variables whose p-value allows to reject the *Null Hypothesis,* it was important to be aware about the fact that presence of collinearity could run the risk to make those p-value not reliable enough. In this context, on the basis of the graph illustrated above, a series of attempts in order to exclude the presence of collinearity between predictors in the model has been made and a series of models have been tested. In particular, during the process the attempt has been developed in two directions:

. Trying to delete the variables which proved no significance in the full model[4], otherwise

. For the variables above mentioned, trying to delete the predictors with high collinearity with them.[5]

The process did not show any significant improvement in its performance if compared to the full model. However, the necessity to exclude variables due to absence of significance and presence of collinearity lead to a choice of exclude the variable *age, Indus, chas.* The resultant `Model_1` slightly decreased the measure of its performance on the train set,

[3] The graph illustrates the connections between variables with presence of collinearity above 0.6

[4] The one with all the predictors set into it.

[5] As before mentioned, the true behaviour of a variable can be covered and misinterpreted in case of collinearity with another variable into the model.

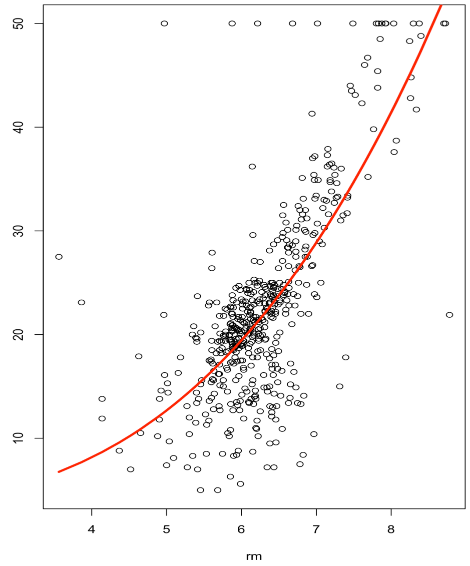
assessing it to R-square = 72.5%, but it allowed to build a model in where all the coefficients of the variables have a high level of confidence to never assume value 0[6].

1.B. Backward and Forward selection + Mixed methods

The following approach allowed me to set a more computational and strategic approach, where the software could automatically provide the best feature selection based on a sequential and defined approach.
The first of them is *Backward selection*, which allowed the software to start from a model containing all the predictors model and backward excluding from the model those parameters which were considered useless. The concept of utility in this case is measured by the *Akaike information criterion* (AIC), which when low suggests a parsimonious model. The second approach is the *Forward selection*, and the exact contrary of the aforementioned, since it starts from a model including no variables and include only those parameters which do provide the best result in terms of AIC. In this specific case, both the models returned the same result in terms of feature selection and consequently AIC. Specifically, the best chosen model involves the full amount of predictors with the exclusion of the variables *Indus* and *age,* and the AIC equal to 771.18. The same result is provided by the *mixed method*.(`Model_2`)

1.C. Extensions of the Linear regression: Interactions, Non-Linear transformations

The last approach regarded an attempt to deepen the process of feature selection by including interactions and/or non-linear transformations. In the first case, before considering to insert a combination between variables in the model, it is firstly opportune to verify if this interaction makes conceptually sense and secondly if it is statistically significant. In this context, by developing a model which only combines the variables *rm* and *lstat*[7] together, following the idea of the *Less is More,* the R-squared of the new model stated on the 72.67 %. Moreover, by adding the variables *ptratio*, *dis* and *nox* the new model has been able to reach an R-squared of 76,15 %. (`Model_3`)
A significative pattern seemed to be occurring in relationship with the variable rm. By applying a polynomial function to it with grade 3, it has been observed that the the residual sum of squares decreased in the presence of a cubic line. As a consequence, I tried to apply the new (cubic) parameter to the previous model, and the result is an increase in the R-squared of almost 4%, reaching the value of 80.4 %. (`Model_4`)



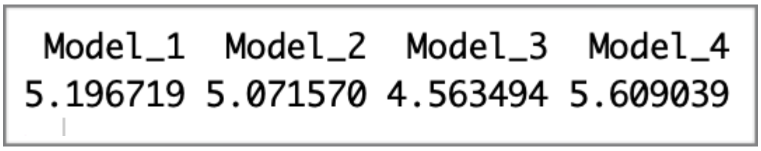
[6] i.e. *p value < 0.05*. A coefficient assuming value 0 would mean no influence in the response, plus the fact that changing their sign means change the slope of their linear regression.

[7] average number of rooms per dwelling * Lower status of the population (percent)

STEP 2: Model evaluation

After verified the performance of the four different models on the test data, it has been possible to observe that the lower *Root Mean Squared Error verifies* in the case of `Model_3` which despite having a slightly lower value of R-squared than `Model_4` produces a better predictive result on the test data. As a consequence, `Model_3` is selected to be compared with successive modules.

RMSE FOR THE FOUR MODELS



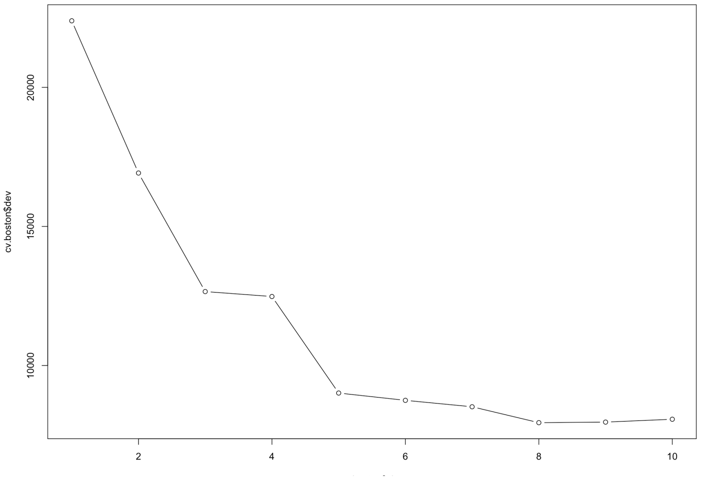
```
 Model_1  Model_2  Model_3  Model_4 
5.196719 5.071570 4.563494 5.609039 
```

Estimating the Accuracy with Cross-Validation

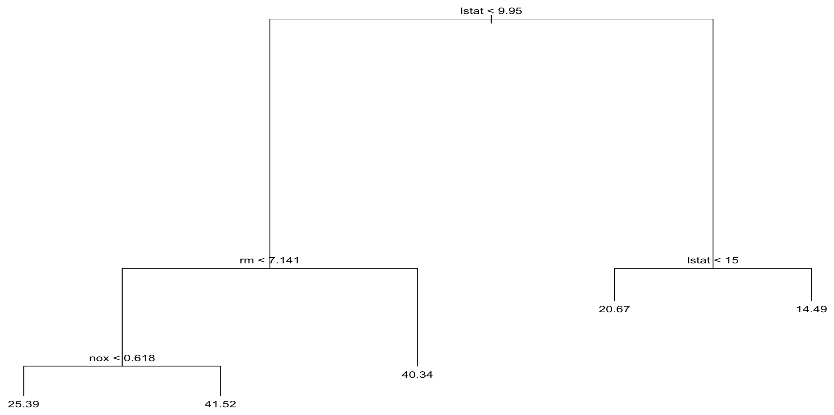
The dataset of interest represents a relatively small data. In fact, with 506 observations and the necessity to sacrifice a part of them for the test analysis (in this specific coursework, the middle of them), the investigation can be partially limited by this element. On the other side, the possibility to take advantage of the cross-validation approach in order to have a better estimate of the performance and be enough confident to build the model on the whole dataset, has led me to attempt this path in order to verify the possibility of better solutions. In this conditions, by performing 10-fold Cross-Validation with different parameters, the result which gave me the best estimation of the performance included the same parameters of the `Model_3`.This will lead me to the possibility to build the model on the entire dataset rather than the half of it, increasing its variance and being enough confident that its future performance on unseen data will have a RMSE around 4.43.

Regression Tree

The second direction in which I headed my approach was the one related to the concept of Decisions Tree (Regression tree, in this case), and its sequential optimisations.
The choice of attempting to build a model through this algorithm relied on the fact that Regression trees have high visual impact and interpretability, which does represent a factor to take into consideration when building a model. However, decision trees tend to be less competitive in terms of accuracy, and this is the reason why after the technique was taken in consideration and visualised, it became necessary to move towards more elaborated techniques, such as Random Forest, Bugging and Boosting.



After have performed the recursive splitting, which generated 10 terminal nodes, the Cost complexity pruning was applied in order to prevent the risk of having a model too prone to overfit. As shown in the graph the best size of the subtree is identified to be 5. In fact, despite the deviance still seems to decrease with the increasing of the size, the necessity of keeping the right trade-off between bias and variance determines the necessity of taking the value after which the trend of decrease stops(it 'decreases less').



The 5 nodes regression tree is then defined, and its Root Mean squared error is identified to be 4.07, the lowest identified so far.

Random Forest, Bagging

With the Random Forest process the necessity is to define the number of predictors *m* which are allowed to participate to constitution of the trees, out of the full range *p* of predictors. In this context, the generic rule to identify the number *m* has been followed, which means:

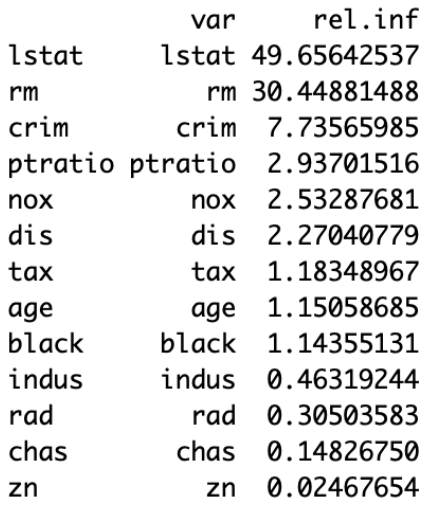
$$m = \frac{p}{3} \quad \text{in case of regression}$$

Consequently, the number has been chosen to be 5. This means that the amount of predictor participating to the building of the tree for each round is 5. The result of the Random Forest model has set the amount of *Root Mean Squared Error* at 3.10. A different amount of m has been attempted. However, no number have determined a diminution of it. With this purpose, a particular amount of variables to be considered was also tried in the following phases, that is the value $m = p$.

This particular case is called *Bagging* and, as earlier mentioned, it builds different decision trees with the possibility to take in consideration the whole range of predictors in occasion of every tree. Although this process reduces the independence between the different trees from each others, it still provides a very significative solution to reduce the variance without the necessary consequence of an increase of bias. At this regards, the result of the *Root Mean Squared Error* show closeness to the performance of the Random Forest model, but it do not reach them.

Boosting

The first step in the building of the Boosting model was to set the different parameters:
. The number of trees has been initially set very high. Despite Boosting could suffer of overfitting, this strategy allowed me to work on the other parameters before in order to verify the trend of them.
. The shrinkage parameters, which represents the learning rate of the boosting process.
. The number of split for each tree, indicated as interaction.depth.
One of the advantages of the Boosting process is that it allows to graphically visualise the role of each variable within the building of the model. Moving from this opportunity, I attempted to create a model which could exclude from the process the variable with less relative influence to the process. The result is stored into a model which do not include the variables *zn, chas, rad, indus.*
This strategy has allowed to reach the value of root mean squared error of 4,14.



```
            var     rel.inf
lstat     lstat 49.65642537
rm           rm 30.44881488
crim       crim  7.73565985
ptratio ptratio  2.93701516
nox         nox  2.53287681
dis         dis  2.27040779
tax         tax  1.18348967
age         age  1.15058685
black     black  1.14355131
indus     indus  0.46319244
rad         rad  0.30503583
chas       chas  0.14826750
zn           zn  0.02467654
```

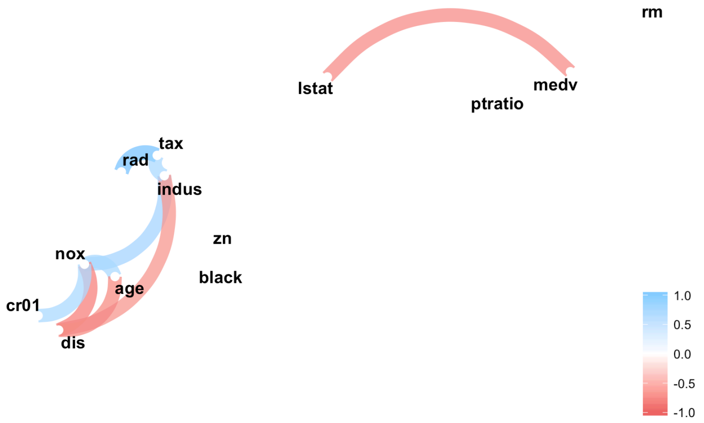
Regression task: models chosen

~~.Linear regression(Model_3).~~	~~RMSE ≅ 4.56~~	~~(4.43 after CV)~~
.Regression Tree.	RMSE ≅ 4.07	
.Random Forest.	RMSE ≅ 3.10	
~~.Bagging~~	~~RMSE ≅ 4.18~~	~~(very similar to RF)~~
.Boosting.	RMSE ≅ 4.14	

CLASSIFICATION TASK [8]

The necessity to predict a binary output which could identify whether an area is above or below the median in relationship with the crime rate per capita (*crim* variable) led me to the necessity of creating a new variable, which has been named *cr01*. In particular, this variable represents the dummification of the *crim* one, and it takes value 1 in case the crime rate is identified to be above the median, and value 0 in case it is below. Consequently, it was important to delete the variable *crim* from the dataset, avoiding to take advantage of it while building the model. In fact, this would create a case of high collinearity in the definition of the model, as well as it would logically not represent a big challenge the one to predict if a variable is above an average, knowing its value.

The second step approaching the classification task, just like it was the case for the regression task, regarded the necessity to verify the collinearity. In this case, the main focus on the analysis is focused on the correlation between the predictors and the response variable, since the other correlations have been already shown in the above section, and they do not differ from that. From the graph shown below the variables correlated for an amount superior of 0.7 are identified. In this case, the response *cr01* seems to be highly (positively) correlated with *nox.* However, by slightly reducing the parameters of the correlation in order to visualise a correlation above 0.6, it was was possible to observe the following correlations:



Negative correlation between cr01 and:
dis (-0.62)

Positive correlation between cr01 and:
indus (0.60)
age (0.61)
rad (0.62)
tax (0.61)

In order to conclude the introductive analysis, it was necessary to verify the proportion of 0 and 1 that the variable cr01 takes within the dataset. This is an important aspect, since all the results obtained by the process of classification need to be compared to this called *Baseline*. [9] In this particular case, due to the characteristics of the split [10], the data are perfectly divided into two halves.

- 50 % below the median. (value 0)
- 50 % above the median. (value 1)

[8] Since a big portion of the contents that will be considered in this section have been already treated in detail in the previous section, with the goal of avoiding repetitiveness this section will be presented with a focus on the results.

[9] Supposing to have 90% of observation identified with 1 (Yes), if the model has a True Positive Rate of 70%, then the model is not performing well, since by simply considering all the variables Yes without any model the True Positive Rate would increase.

[10] The 0/1 values are split based on their value above or below the median, which is the value that splits the data in two parts.

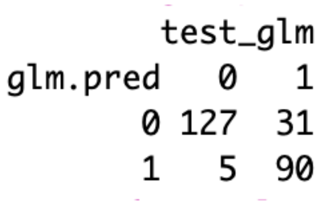
The necessity to perform a balanced analysis needs to be taken into account when dividing the dataset in order to obtain the Train and the Test set. In this sense, it is important that the values of response variable keep the same proportion than the original dataset.

Logistic Regression

The approach of feature selection followed in this case regarded the attempt to increase the performance of the model, measured with the AIC index, by defining the variables through the *Backward selection*, *Forward selection*, and the mix between them (*Mixed selection*). Specifically, the three cases have resulted in an identical result in terms of selection of the feature. This is a logistic regression including *nox + rad + age + medv* as variables .

With regards to the defined model, it is important to observe that all the values are confirmed to be significant by having a p-value far below 0.05. The AIC is 138.88 and the *Area Under the Curve* (AUC) is equal to 0.853.

The performance of the model was then analysed in relationship with the test set, setting the threshold of the probability value to 0.5. The confusion matrix showed a significative accuracy. In fact, 85.77 % of the time the model predicted the right value of *y,* which is far above the percentage of accuracy deriving by the *prior probability* (0.5 per class). However, it is important to observe how the False Positive Rate is higher than the False Negative Rate (3% vs. 25 %). The meaning is that the model tends to be right 'more often' by predicting values of crime above the median than the time it has to predict values below the median.[11] In this context, when defining a model it is important to be aware of which mistake in the prediction would be more 'annoying' for the people interested on it (can be us, a client, a boss, or the community). In my opinion, in this situation can be better to make a mistake by predicting a value below the mean, rather than define an area safer than the reality. As a consequence, despite the confusion matrix is showing an unbalanced situation between *sensitivity* and *specificity*, the model can be 'forgiven' due to the fact that it does the 'less worse' mistake[12].



```
         test_glm
glm.pred   0   1
       0 127  31
       1   5  90
```

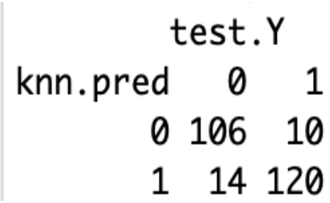
A slightly better result seems to be obtain by removing the variable *age* from the model. The attempt was made due to the high collinearity between this variable and the variable *nox.* This allowed to reach an accuracy of 86.95 %, and an AUC of 0.8636. Moreover, the slight tendency of the model of a higher sensitivity than specificity is coherent with the above mentioned attempt to minimise False Positive Rate. However, both the results of a 10-fold Cross Validation as well as Leave-One-Out cross validation have confirmed a higher potential for the model including *age* as a variable.

[11] 25 % of the times, it considers a crime rate to be below the median while it actually is above.

[12] Pardon my poetic licence !

K-Nearest Neighbours

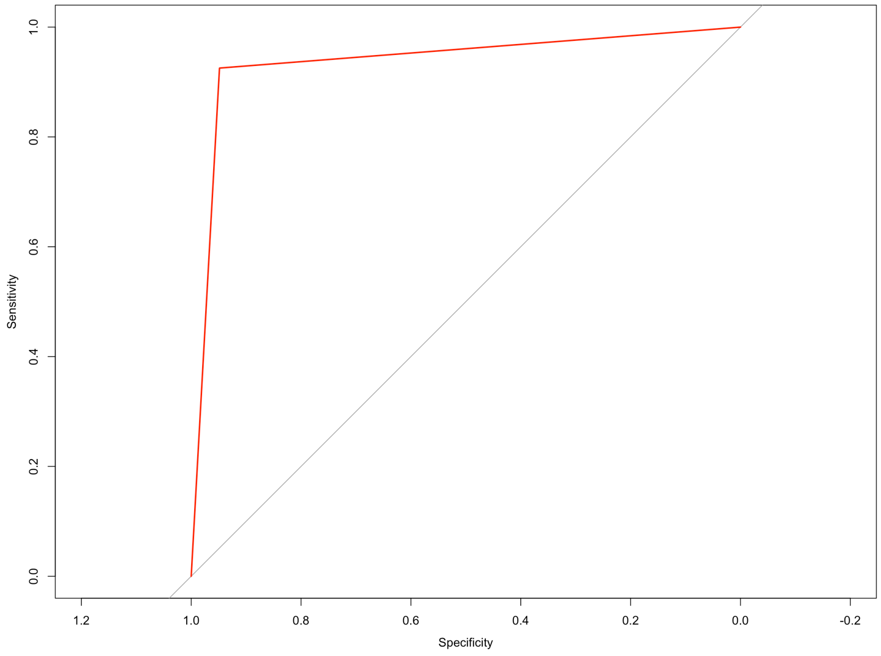
Due to the fact that the K-nearest Neighbours algorithm relies on the concept of distance to be able to work, the first necessary step in order to introduce this method in my analysis regarded the Standardisation of the data. In particular, each data has been rescaled in order to have a *mean = 0* and *standard deviation = 1*. Before that, the dataset has been randomised. By setting the number of close neighbours to identify to 1 (k = 1), the results showed an accuracy of 90.4%., with AUC = 0.9047. Several experiments have been performed with the goal of finding the best value of k, which is the number of neighbours which concur to the definition of the class of the response. One of those number was the square root of the number of observations contained in the training set (in this case, k = 15). This represents a rule of thumb to start performing an analysis with KNN. However, the best parameter k has revealed to be the first tried, which was 1.



```
         test.Y
knn.pred   0   1
       0 106  10
       1  14 120
```

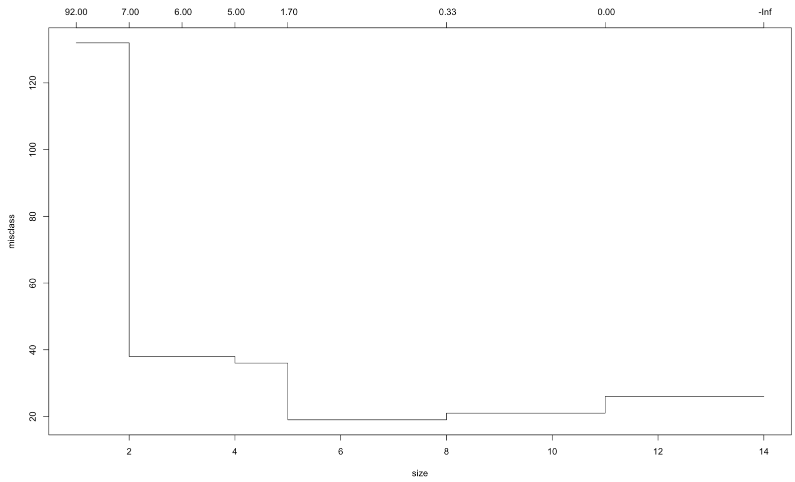
Curse of Dimensionality

Despite the high and surprising results of the model above defined, it was important to take in consideration the issues which raise when dealing with KNN in high-dimensional spaces. As a consequence, it was important to reduce the number of dimensions (*I.d the number of variables*). The new model, containing only 3 predictors, showed more accuracy than the previous ones. In particular AUC = 0.9368 with KNN (k = 1). In the graph below, the ROC curve is shown.



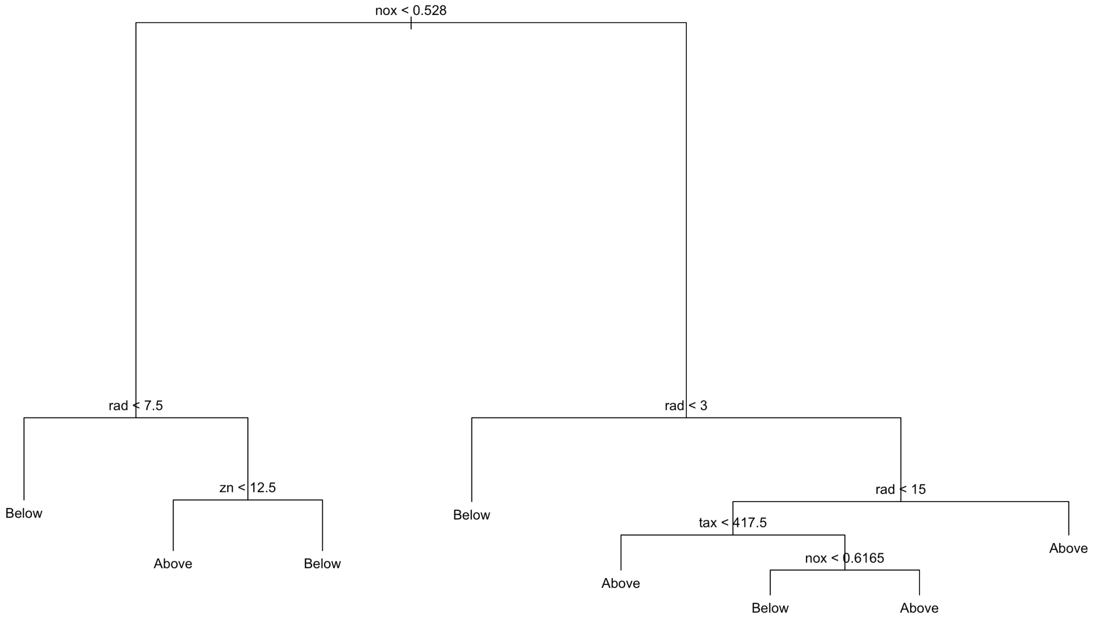
Classification Tree

The classification tree phase started with the necessity of convert the numerical dummy variable into a categorical one. In order to ease the interpretability of the tree, I created a new variable called *crime* with two categories *Above* and *Below.* At this purpose, it was important to not forget to delete the old variable *cr01*, since keeping it would have lead to a perfect collinearity between this variable and the new response variable, making the model perfectly accurate, and useless! The initial recursive tree contained 14 nodes, and despite the high accuracy shown (90%), it was necessary to work in the direction of reducing the number of nodes in order to avoid the risk of overfitting.



The graph illustrates the number of optimal nodes between 5 and 7, as resulted from the cross-validation applied during the pruning process. In fact, for this values the classification error rate is at its lowest level.

The new model obtained through the classification tree algorithm proved an accuracy of 94.47 % over the test set, as well as a very intuitive solution which helps to understand the concept to those who can be unfamiliar with the subject.

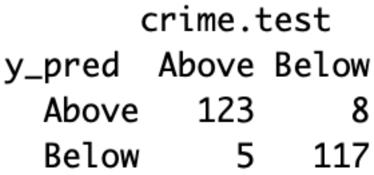


Random Forest, Bagging

According to the generic rule the number of variables allowed for each turn should be equal to the square root of the total number of predictors:

$$m = \sqrt{p}$$

In the specific case, three attempts have been made, and the best one has concurred into the creation of a significative model . In particular, those attempts regarded m = 4, m = 3, and m = 2. The model has been chosen with the last one, which proved an accuracy of 94.86 %.



```
        crime.test
y_pred  Above Below
  Above   123     8
  Below     5   117
```

The same results in terms of accuracy has been reached by the Bagging process, which is nothing different than a Random Forest which allows all the predictors to concur in the choice for each turn. (m = p).

Classification task: models chosen

.~~Logistic regression.   Accuracy ≈ 85.77%~~
.K-Nearest neighbour.   Accuracy ≈ 93.68%
.Classification Tree.   Accuracy ≈ 94.47%
.Random Forest   Accuracy ≈ 94.86%