Roméo LÉON
MSc in Big Data and Business Analytics
R & Business Analytics
2020/12/8

Homework III

Problem 1

Question 1

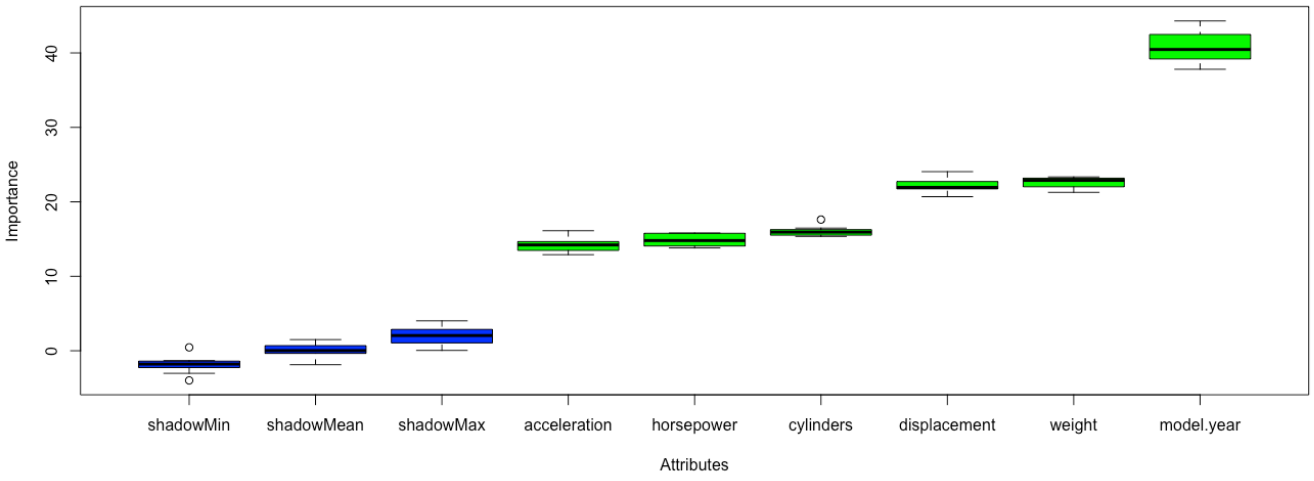
The first step to solve this question is to load the CSV file provided, then to identify how the missing values are displayed in the dataset. Here the former were in the form of question marks.

Then, I decided to reformat the missing values '?' into the 'NA' format to later clean the data with the tools provided by R.

Given the small number of missing values - Only 6 in the horsepower column – that I identified with the is.na() method of R, I decided to drop them as they represented only 1,5% of the entire dataset.

Question 2

First of all, to perform linear regression and local polynomial regression, I needed to identify the attributes that I would be using on those two models. I have chosen to perform feature selection using the Boruta library, according to the latter the importance is organized as following :

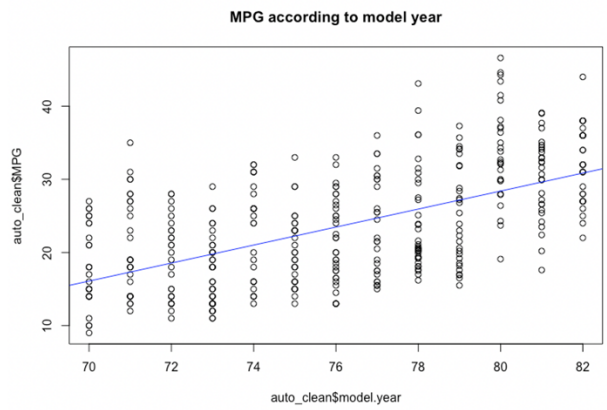


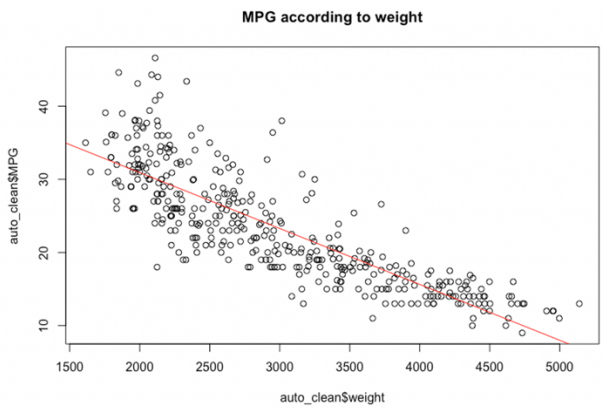
Hence, the two most important attributes are : 'model.year' and 'weight'

Then, I created the linear regression models in accordance with the previously selected attributes. I used the lm() functions and then assessed the performance of the models by looking at the summary statistics and more precisely the Adjusted R-squared of each linear regressions (by using summary()). For both models I got the following results :

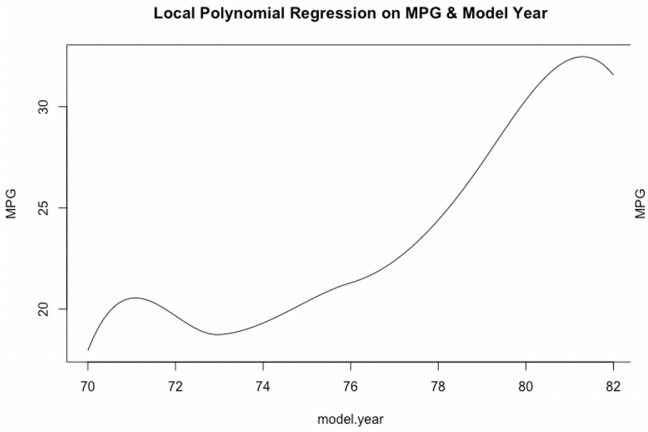
- linear_year : Adjusted R-squared = 0,3353
- linear_weight : Adjusted R-squared = 0,6918

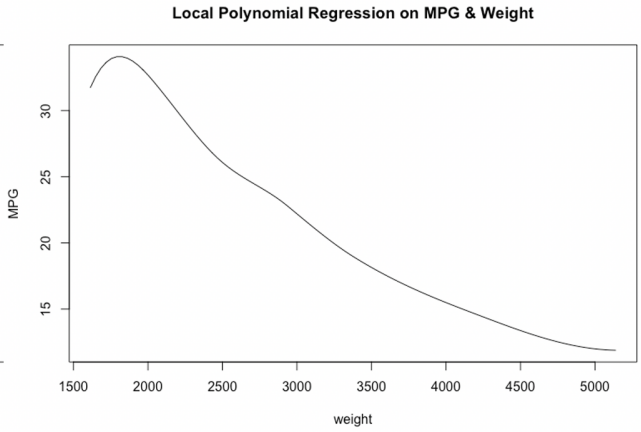
Afterwards, I plotted the data of the different attributes with the 'target' variable MPG and added the linear regression on the same plot.





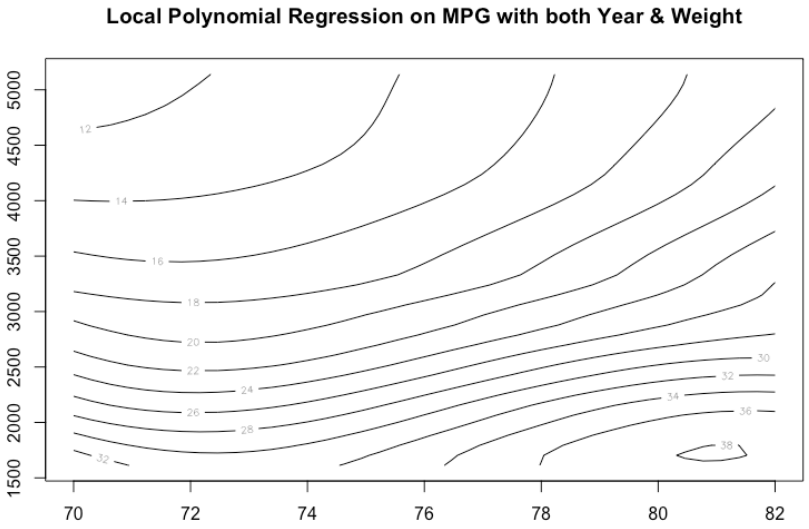
After performing the linear regression, I create models for Local Polynomial Regression with the locfit() and lp() functions. I have tuned the hyperparameter 'deg' to set a local polynomial regression of order 4 in order for the regression to fit the data more precisely while not overfitting it.





From those two kind of regression, we can clearly see that the older a vehicle is, the less MPG it can drive (supported by the adjusted r-square value mentioned above). Furthermore, the weight is also a determining factor for the MPG metric, as the heavier a vehicle is, the less MPG it can drive. In addition to that, it's clear that local polynomial regression is more suited to represent the data.

For the sake of comprehensiveness, I have decided to perform a local polynomial regression on both 'model.year' and 'weight' at the same time. The plot below corroborates the previous methods as the main outcome of this chart could be rephrased as : '**The lighter and newer a vehicle is, the better its MPG performances are.**'



Problem 2

The first step in solving the question is to load the CSV file provided. Then, I decided to map the values in the column 'income' (transform them into categorical values) to transform the expressions '<=50K' and '>50K' to 0 and 1 respectively.

In a second stage, I decided to explore the data visually in order to discover the distribution of the features, detect missing values and more generally get a first glance at how my data is structured.

Then, to perform the splitting of the data into a training and test set, I used the library 'caTools' which offered a very straightforward syntax and efficient code (few lines). I used a ratio of 6000/7125 values for the training set as required by the case study and assigned to two variables 'train_income' and 'test_income' the appropriate data.

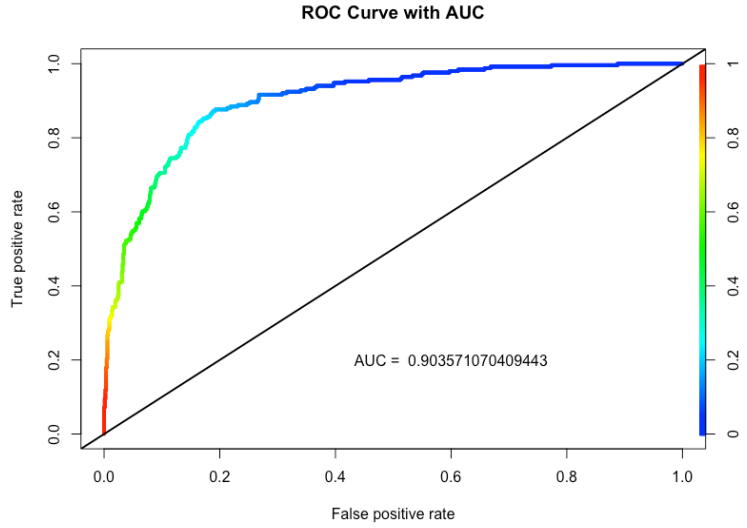
After splitting my data, I created the logistic regression instance with the glm() function and trained it on the training data. Then, I predicted the values for the test data and created a confusion matrix of the classification (see figure below).

To leverage this confusion matrix, I computed the accuracy of the model by using its value and developed the following calculation : $\frac{TP+TN}{TN+FP+FN+TP}$. I reached a satisfying accuracy of 0.857.

In addition to accuracy, I computed recall ($\frac{TP}{TP+FN}$) and precision ($\frac{TP}{TP+FP}$) which yield respectively values of 0.687 and 0.665, in other words, out of all the positive class (Value = 1 (>50K)) we predicted 68,7% of them correctly (The percentage of true positive that were correctly identified) and the precision score indicates us that out of all the predicted positive values, 66,5% of them are correctly classified.

Lastly, I performed the ROC and AUC analysis. I used the library ROCR that yields the TPR and FPR values by predicting the classification using the prediction model I used earlier and the test set. More precisely, I used the function performance() to fetch the former and lastly used the same function to extract the AUC score.

Finally, I plotted the ROC curve, the line of random classification and added the AUC as a legend, please find below the plot.



I obtained a relatively high AUC of 0.903 which indicates that my model has good performance in distinguishing the classes between positive and negative.