LAB1 - DataQuality

Carles Garriga Estrade, Balbina Virgili Rocosa

3/1/2018

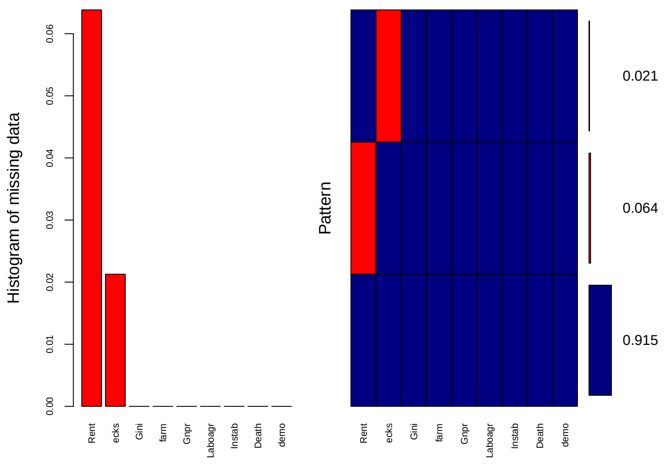
1. Impute missing values and save the result obtained.

First of all, the missing data has been identified and treated in order to be manipulated afterwards. The developed code can be found on LAB01.R file, attached to this report.

After loading the given dataset, we realized that missing values were represented with the missing code NA by default. So, its evaluation was performed from this NA values. With the results retrieved, we can confirm that there is a total number of 4 missing values on the given data, which correspond just a 0.009% of the total amount of data given. Furthermore, it is realized that three of this missing values belong to the 'Rent' variable and the remaining one to the 'ecks' variable. We have developed two different ways to find the missing values for each variable, so we have been able to check that the just mentioned missing values are the correct ones. Also, the missing values for each individual has been found, where Australie, Nicaragua, Peru miss 'Rent' value and Norvege lacks 'ecks'.

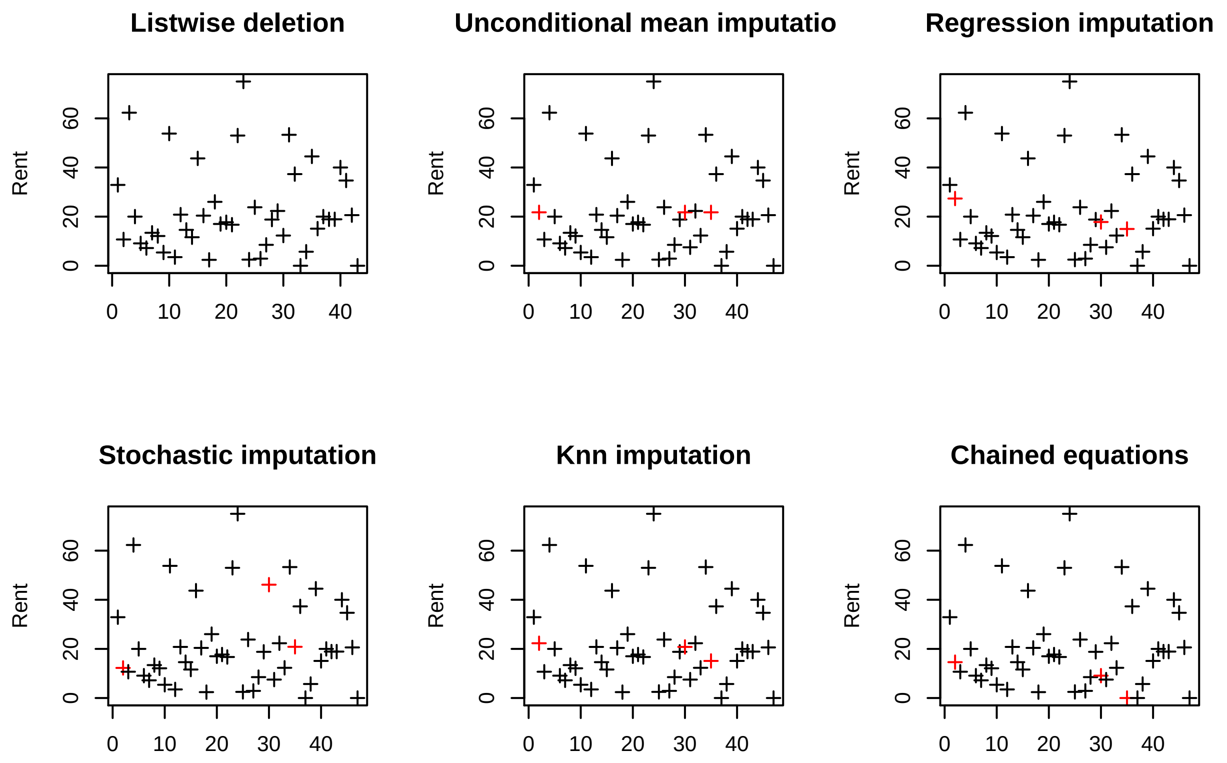
It is considered that the more missing values, the less reliable is the information provided by the variable. That is why the percentage of missing data for each variable and individual has been calculated too. The results obtained are 6.38% for Rent, 2.12% for ecks and 0% for the other variables. And 11.11% for each individual that has a missing value and 0% for all the other ones. Although there's no "reasonable" minimum missing threshold to be considered, missing values tend to increase the percentage considerably as the given dataset is not considered as a large one.

Finally, the patterns on missing values are checked.



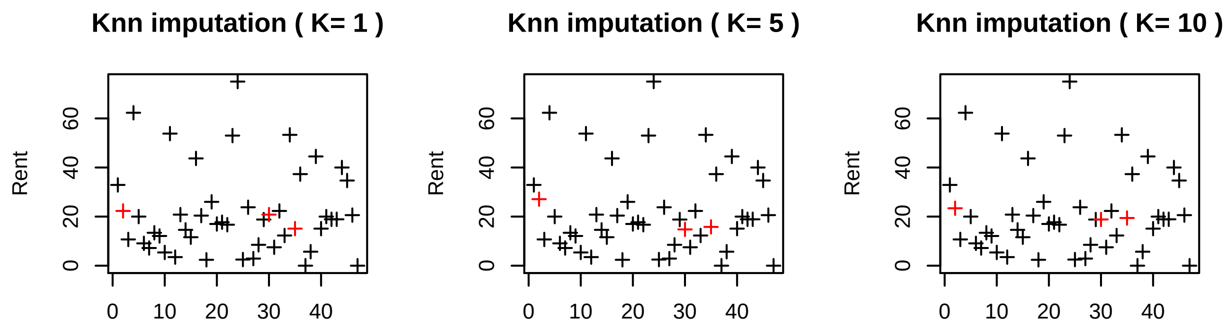
As we can see with the results retrieved, the dataset has a total of 4 missing values (3 for Rent and 1 for ecks). The percentage of each missing value pattern is low, so the data can be considered as MCAR, which is the most favorable situation because missing values aren't related to any pattern.

Once all missing values have been identified and revised, they need to be treated to be able to compute and extract from them the desired information. There are several ways to do it and six of them have been developed (the whole code can be found on LAB01.R file). The different obtained results for variable Rent are shown below.



- Listwise deletion: every individual with a missing value is deleted. This method looses information and bias the results.
- Unconditional mean imputation: every missing value is replaced by the global mean of the variable.
- Regression imputation: every missing value is replaced by the predicted value from a multiple regression.
- Stochastic imputation: every missing value is replaced by the predicted value from stochastic regression imputation.
- Knn imputation: every missing value is replaced by the values of the nearest neighbors.
- Chained equations: apply the nearest neighbours (with k = 1) to impute every missing value from the closest individual, to obtain the final realistic imputed values.

Finally, we have studied the K nearest neighbor imputation more deeply. In other words, we have executed this method for filling the missing values for Rent variable with different values of the parameter K, to be able to see its implication. The K variable corresponds to the number of neighbors that need to be explored to find the desired value of the missing value. The different results obtained are showed below.



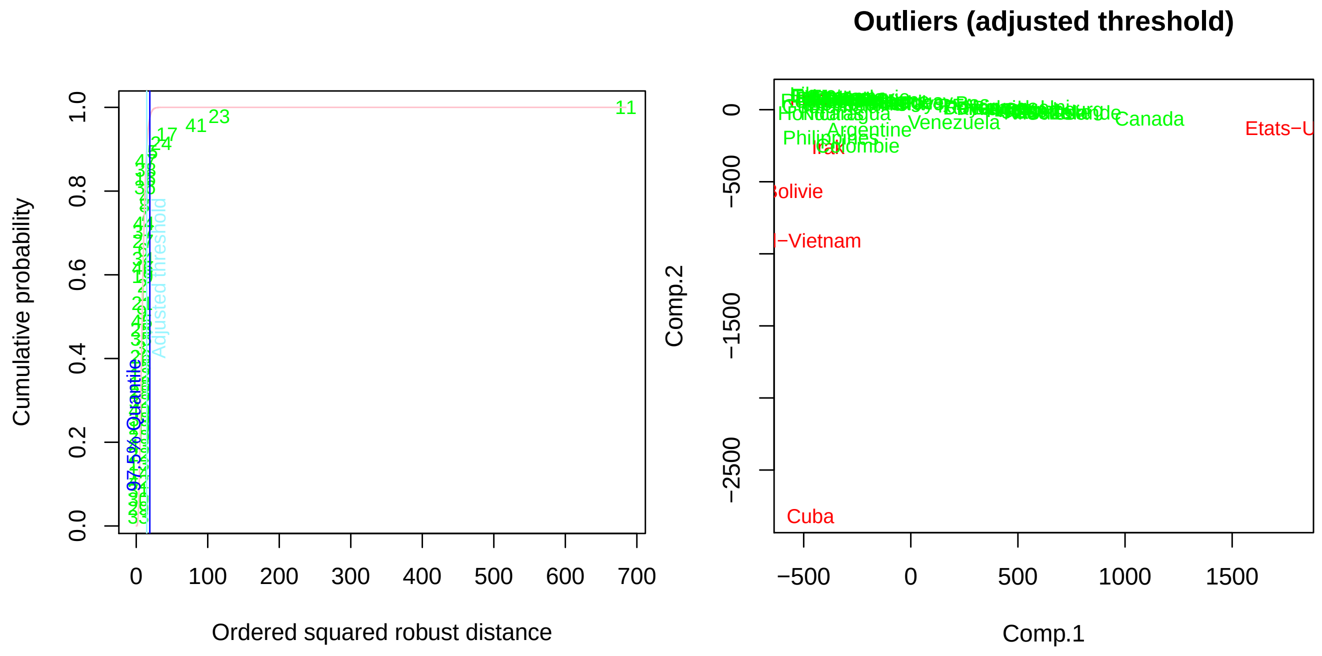
In KNN, finding the value of k is not trivial. A small value of k means that the noise will have a higher influence on the result whereas a large value will make it computationally expensive. In this case we would preferably choose K with a reasonable value of 5, being a good tradeof between noise (only 47 individuals and picking 5 neighbours at a time) and performance.

2. Write the R script to find multivariate outliers and detect them.

After we've imputed the missing values by using its nearest neighbours, another factor that we need to take into account are the outliers. In order to detect and remove them, two methods will be used.

The first implemented method uses the Mahalanobis distance and the principal components (providing the covariance matrix) in order to find the outliers present in the dataset. Two approaches are taken: ??? The first approach computes the mahalanobis distance and compares it with both the 97.5% quantile in the chi-square distribution as well as the adjusted quantile (using the adjusted Adaptive Reweighted Estimator arw()). ??? The second approach, using principal components, measures the scores of the first and second components (most significant) in order to determine which individual can be designated as an outlier (by comparing the scores of the first two components with the adjusted thresholds).

```
##     Bolivie        Cuba  Etats-Unis        Inde        Irak Sud-Vietnam
##        TRUE        TRUE        TRUE        TRUE        TRUE        TRUE
```

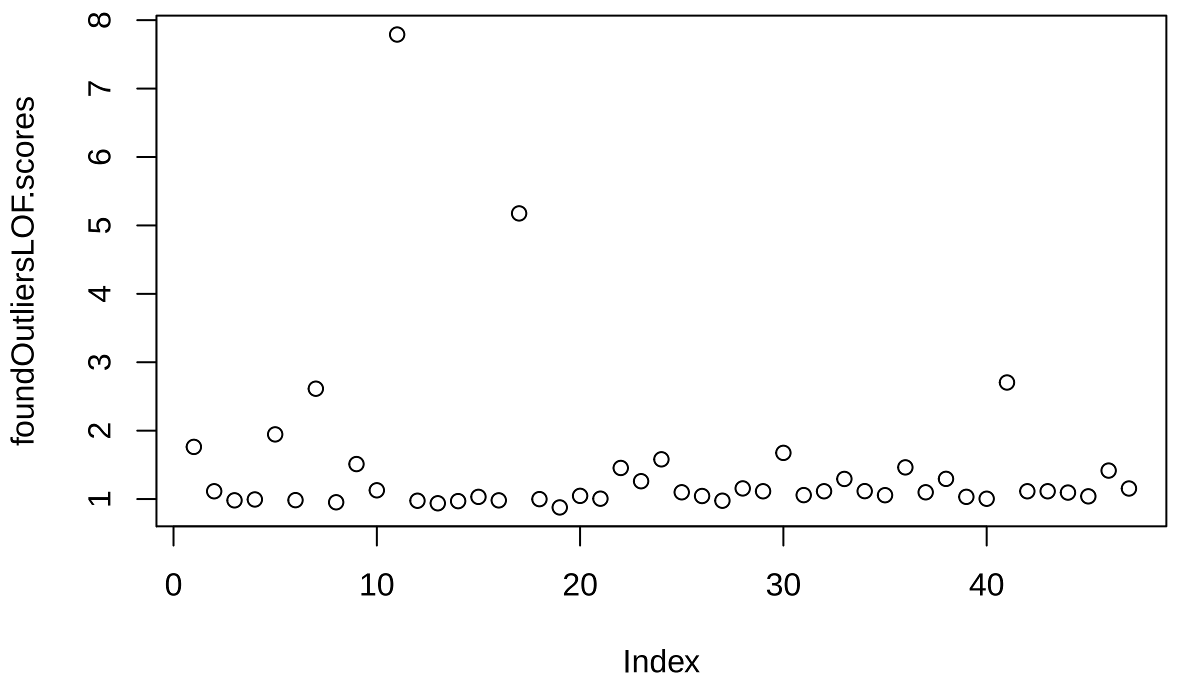


The resulting plot of the first approach, having set a quantile of 97.5% as threshold, shows a certain number of possible outliers (shown as the indexes of the observations) that need to be taken into account. However, a clearly outlier corresponds to the observation found in the 11th index, Cuba.

Observing the plot for the seccond approach, we can clearly confirm the outliers found in the first approach. Identifing the red-colored individuals as outliers, the gigatinc distance of Cuba regarding the rest of individuals is enough to ensure that Cuba is an outlier that needs to be taken into account.

A second method, based on the local outlier factor, provides a score based on the distance of a point to its k-nearest neighbours (density of the neighbourhood) compared with the density of the neighbours of the first point (distances of the k-nearest neighbours of the neighbour). The resulting score value (for each point) will determine if the point (along with his k nearest neighbours) is in a dense region (score < 1), or its comparable in distance to his neighbours (score ~ 1). Scores greater than one could indicate that the point is far (distance wise) from a denser neighbour and therefore, would be considered an outlier.

```
# LOF
foundOutliersLOF.scores <- lofactor(noNAvaluesDataset, k=5)
plot(foundOutliersLOF.scores)
```



```
foundOutliersLOF = order(foundOutliersLOF.scores, decreasing=T)[1]
```

Observing the previous plot, we can distinguish several points (identified in the x-axis as the indexes of observations in the dataset) that have an score much greater than even 4. From those points, we are going to consider the first one (with the highest score, index 11 -> Cuba) as outlier and should be treated before any further data analysis.