Costa Rican Household Poverty prediction

Programming in R
Group Project

Luca Martorano (u306264)

Riccardo Rampoldi (u129700)

Letizia Minarini (u736034)

Shashank Srinivasan (u336695)

Tolgahan Koçyigit (u826393)

Introduction

The problem of poverty is something faced in every part of the world. Looking worldwide, estimates sustains that in 2019 just under 600 million people across the world will live in poverty (reference). For this reason, what is nowadays important is trying to develop a method that helps national governments to identify which households live under the level of poverty and that could need support. In this case, the use of efficacious machine learning models could help governments not only to identify these households, but also to understand which are the main factors that are associated to the poverty status and act on them. So, these models can be used as important tools to understand and try to find a solution to the phenomenon. Our purpose in this work is to create a model able to classify between poor and non-poor households with the highest accuracy possible and to investigate the main factors that contribute to this classification.

Dataset

The dataset selected comes from the open data depository website *Caggle*. In the repository the data were already divided into two different csv files: a training and a test set. Despite that, he huge number of observations of the files and considerations about some variables led us to use [illegible] the file containing the train data; a file with 9557 observations [illegible] 143 variables. The link to the dataset can be found in the explanatory file.

EDA

In order to use the dataset, a preprocessing of the data was done. As first step, we checked and cleaned for the presence of missing data (NAs). Some columns were removed because of the lack of too many observations while the rest of missing values, due to the small number, were replaced with the variable's mean. The classification column has been modified to let us perform a binary (poor/not-poor) classification instead of a multilevel one. In order to reduce the high number of variables present in the dataset, a correlation matrix was done considering all the numerical variables. Some clearly correlated groups popped out and for this reason it has been decided to remove all variables that correlated for more than 90% in order to avoid the phenomenon of overfitting. Looking at the dataset it was possible to see how the class to discriminate were unbalanced, but in this case, because of the binary classification we decided to maintain the dataset non-balanced. We then decided to create some informative graphs of the most important variable (in our opinion) present in the dataset (appendix 1.).

Method

For the feature's selection, we applied some basic filtering methods to remove redundant and not significant variables:

- Removing anything that has only a single value
- Remove anything that has a correlation with another feature of > 0.90

In order to reduce the number of the variables we also used the "*varImp*" function of the caret package. Since varImp works only on models, every model was made twice: comprehending all variables and with the 20 most important ones.

Partitioning

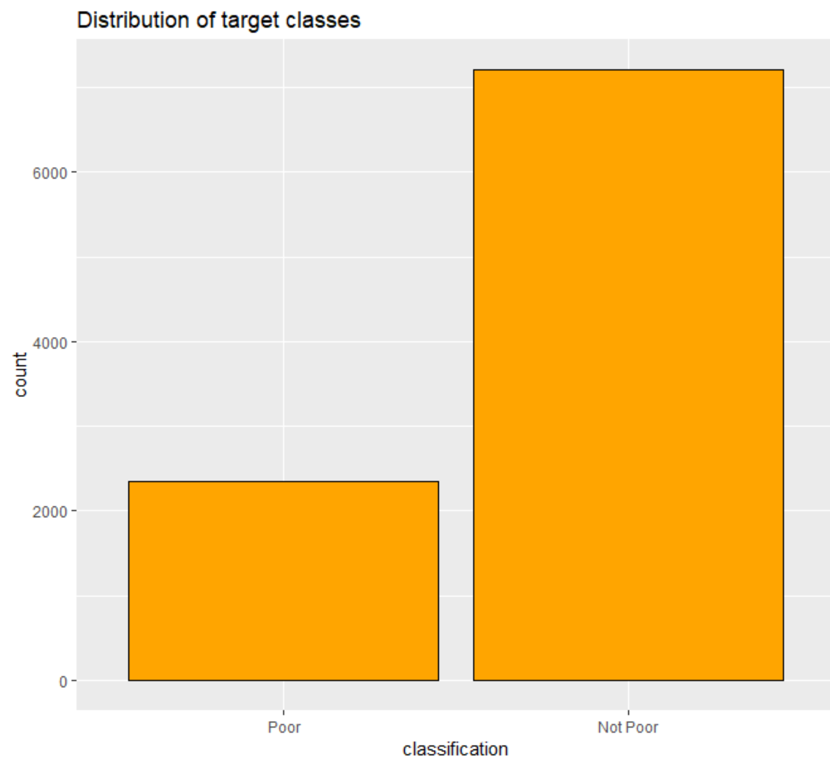
The final dataset obtained after pre-processing was partitioned, creating a training set (comprehending 30% of the data) and a test set (comprehending the remaining 70% of the data).

Conclusion

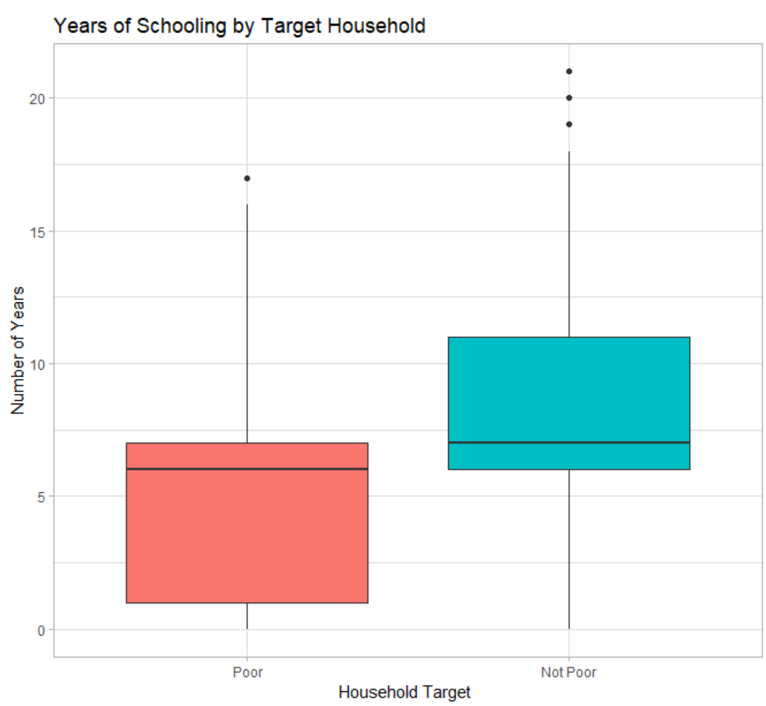
In order to achieve the aim of the project, six different models were considered. The six models were based on three different algorithms: k-nearest neighbors (**knn**), logistic regression (**glm**) and random forest (**rf**), this last one selected for its flexibility in both classification and regression tasks. Considering the different logic behind these algorithms we expected different outputs from the different models, and the results (appendix 2.) show that this is true. Nevertheless, despite the good performances of all the models, the random forest with the 20 most important variables seems to be, without doubts, the most suitable one for our binary classification task. This model is able to classify the poverty level of the households with an accuracy (metric used to evaluate the model) around 95%. This is a really good result, in particular if considering that the model is strongly equilibrated, having good performances also in terms of sensitivity, specificity, precision and negative predictive value. Furthermore, this model allows us to see which are the variables that affect it for the most:

“*meaneduc”* (average years of education for adults), “*SQBdependency”* (Dependency rate, calculated = (number of members of the household younger than 19 or older than 64)/(number of member of household between 19 and 64)) and “*overcrowiding*” (people per room). As said previously, looking at the most weight variables is really important for these kinds of tasks because it allows governments to act precisely, finding solutions to try to solve the problem. Anyway, it is important to remember that our model performs a binary classification, while a more accurate way to investigate and deal with the problem could come from a multilevel classification of the poverty level that could further identify that households that lies in conditions of extreme poverty and that need immediate help. This is just an initial step toward and a good example of how a machine learning algorithm could be used to solve or try to solve real world problems.

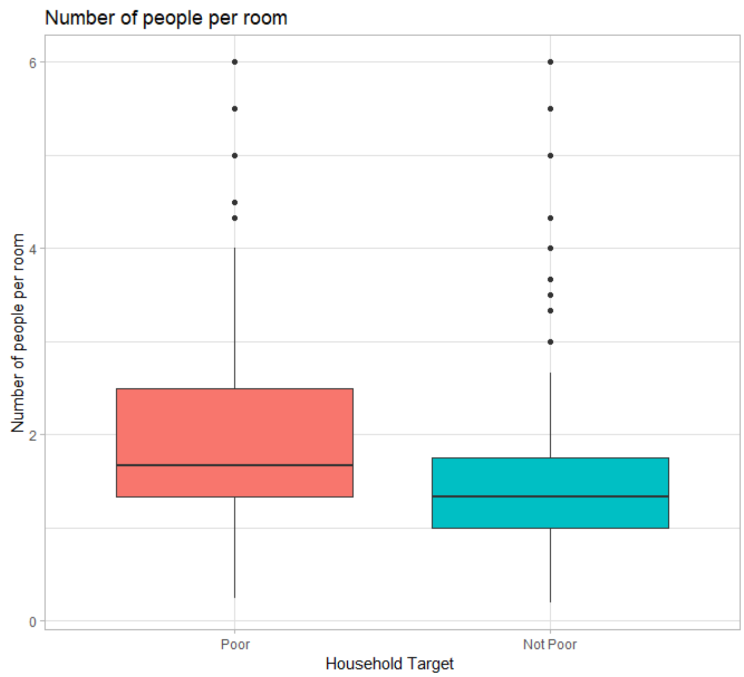
Appendix 1.



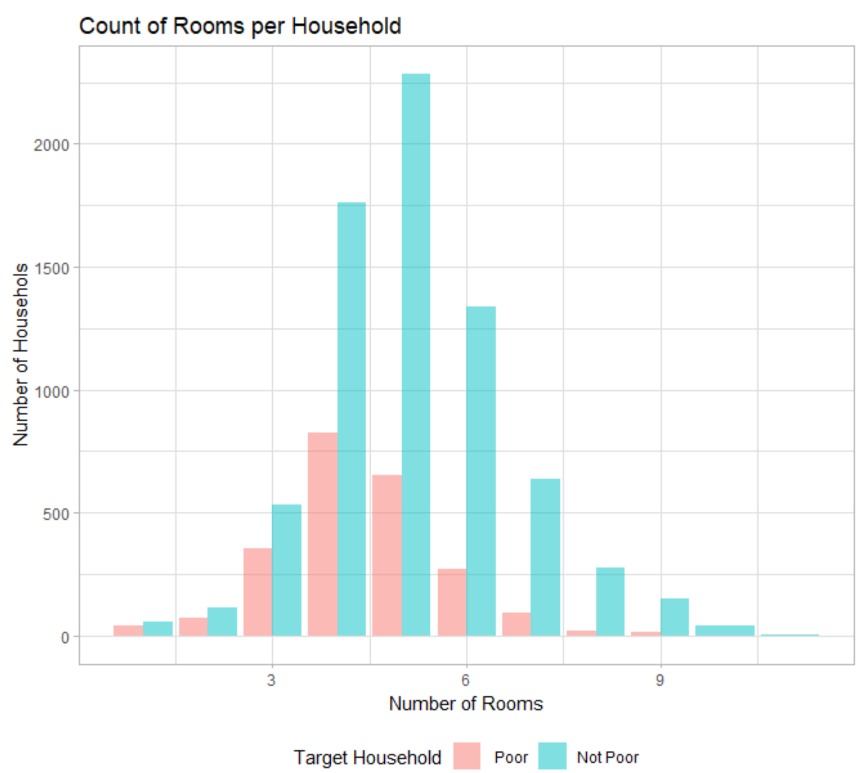
Plot 1. Representation of the distribution of the classification variable into its two levels.



Plot 2. Boxplot representation of number of years of education per classification level.



Plot 3. Boxplot representation of number of people per room per classification level.



Plot 4. Representation of number of rooms in households per classification level.

Appendix 2.

```
Confusion Matrix and Statistics

          Reference
Prediction Poor Not Poor
  Poor      370      101
  Not Poor  335     2060

               Accuracy : 0.8479
                 95% CI : (0.8342, 0.8608)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5383
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.5248
            Specificity : 0.9533
         Pos Pred Value : 0.7856
         Neg Pred Value : 0.8601
             Prevalence : 0.2460
         Detection Rate : 0.1291
   Detection Prevalence : 0.1643
      Balanced Accuracy : 0.7390

       'Positive' Class :  Poor
```

```
Confusion Matrix and Statistics

          Reference
Prediction Poor Not Poor
  Poor      470      130
  Not Poor  235     2031

               Accuracy : 0.8726
                 95% CI : (0.8599, 0.8846)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6385
 Mcnemar's Test P-Value : 5.221e-08

            Sensitivity : 0.6667
            Specificity : 0.9398
         Pos Pred Value : 0.7833
         Neg Pred Value : 0.8963
             Prevalence : 0.2460
         Detection Rate : 0.1640
   Detection Prevalence : 0.2094
      Balanced Accuracy : 0.8033

       'Positive' Class :  Poor
```

Figure 1. Confusion matrix of the two *knn* models. On the left the one considering the whole variables and, on the right, the one that considered the 20 most important variables.

```
Confusion Matrix and Statistics

          Reference
Prediction Poor Not Poor
  Poor      338      167
  Not Poor  367     1994

               Accuracy : 0.8137
                 95% CI : (0.7989, 0.8278)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : 1.258e-14

                  Kappa : 0.4446
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.4794
            Specificity : 0.9227
         Pos Pred Value : 0.6693
         Neg Pred Value : 0.8446
             Prevalence : 0.2460
         Detection Rate : 0.1179
   Detection Prevalence : 0.1762
      Balanced Accuracy : 0.7011

       'Positive' Class :  Poor
```

```
Confusion Matrix and Statistics

          Reference
Prediction Poor Not Poor
  Poor      280      150
  Not Poor  425     2011

               Accuracy : 0.7994
                 95% CI : (0.7842, 0.8139)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : 4.705e-09

                  Kappa : 0.3773
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.3972
            Specificity : 0.9306
         Pos Pred Value : 0.6512
         Neg Pred Value : 0.8255
             Prevalence : 0.2460
         Detection Rate : 0.0977
   Detection Prevalence : 0.1500
      Balanced Accuracy : 0.6639

       'Positive' Class :  Poor
```

Figure 2. Confusion matrix of the two *glm* models. On the left the one considering the whole variables and, on the right, the one that considered the 20 most important variables.

```
Confusion Matrix and Statistics

          Reference
Prediction  Poor Not Poor
  Poor       610       26
  Not Poor    95     2135

               Accuracy : 0.9578
                 95% CI : (0.9498, 0.9648)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8823
 Mcnemar's Test P-Value : 6.337e-10

            Sensitivity : 0.8652
            Specificity : 0.9880
         Pos Pred Value : 0.9591
         Neg Pred Value : 0.9574
             Prevalence : 0.2460
         Detection Rate : 0.2128
   Detection Prevalence : 0.2219
      Balanced Accuracy : 0.9266

       'Positive' Class :  Poor
```

```
Confusion Matrix and Statistics

          Reference
Prediction  Poor Not Poor
  Poor       626       41
  Not Poor    79     2120

               Accuracy : 0.9581
                 95% CI : (0.9501, 0.9652)
    No Information Rate : 0.754
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.885
 Mcnemar's Test P-Value : 0.0007312

            Sensitivity : 0.8879
            Specificity : 0.9810
         Pos Pred Value : 0.9385
         Neg Pred Value : 0.9641
             Prevalence : 0.2460
         Detection Rate : 0.2184
   Detection Prevalence : 0.2327
      Balanced Accuracy : 0.9345

       'Positive' Class :  Poor
```

Figure 3. Confusion matrix of the two *rf* models. On the left the one considering the whole variables and, on the right, the one that considered the 20 most important variables.