# Problem Set 3: Predicting Poverty

*"Wars of nations are fought to change maps. But wars of poverty are fought to map change"* M. Ali

## 1 Introduction

This problem set was inspired by a recent competition hosted by the world bank: Pover-T Tests: Predicting Poverty. The idea is to predict poverty in Colombia. As the competition states, *"measuring poverty is hard, time consuming, and expensive. By building better models, we can run surveys with fewer, more targeted questions that rapidly and cheaply measure the effectiveness of new policies and interventions. The more accurate our models, the more accurately we can target interventions and iterate on policies, maximizing the impact and cost-effectiveness of these strategies."*

The objective is to predict poverty at the household level. Data, however, are provided at the household and individual levels. You can use individual-level information to build extra variables to improve your prediction.

The data comes from DANE and the mission for the "Empalme de las Series de Empleo, Pobreza y Desigualdad - MESE". The data contains four sets divided into training and testing at the household and individual levels. You can use the variable `id` to merge households with individuals. You will note that some variables are missing in the testing data sets; this is designed to make things a bit more challenging. More information about the data is available at the competition website.

An essential dimension for policymakers is they can *rapidly and cheaply* measure poverty. When building your model, aim for a model that uses the minimum number of variables.

There are two expected outputs:

1. A `.pdf` document.
2. Submissions with your team's predictions in Kaggle at the following link.

## 1.1 General Instructions

The main objective is to construct a predictive model of household poverty. Note that a household is classified as

$$Poor = I(Inc < Pl) \tag{1}$$

where $I$ is an indicator function that takes one if the family income is below a certain poverty line.

This suggests two ways to go about predicting poverty. First, approach it as a classification problem: predict zeros (no poor), and ones (poor). Second, as an income prediction problem. With the predicted income, you can use the poverty line and get the classification. You will explore both routes in this problem set.

The document must contain the following sections:

- Introduction. The introduction briefly states the problem and if there are any antecedents. It briefly describes the data and its suitability to address the problem set question. It contains a preview of the results and main takeaways.
- *Data.*[1]. When writing this section up, you must:
    1. Describe the adequacy of the data to solve the predictive question, the sample construction process, including how the data was cleaned, combined, and how new variables were created.
    2. Include a descriptive analysis of the data. At a minimum, you should include a descriptive statistics table with its interpretation. However, I expect a deep analysis that helps the reader understand the data, its variation, and the justification for your data choices. Use your professional knowledge to add value to this section. Do not present it as a "dry" list of ingredients.
- Models and Results. This section presents the specifications and models used for the predictive tasks. Here you must include three subsections:
    1. Classification models. This subsection describes the classification approach, that is, your attempt to directly predict zeros (no poor), and ones (poor). You must submit at least five (5) predictions trained with at least three (3) of the following algorithms: Linear Regression, Elastic Net, CARTs, Random Forest, and Boosting.

[1]This section is located here so the reader can understand your work, but probably it should be the last section you write. Why? Because you are going to make data choices in the estimated models. And all variables included in these models should be described here.

2. Income regression models. This subsection describes the income prediction approach, that is, your attempt to predict income first and then indirectly predict zeros (no poor) and ones (poor). You must submit at least five (5) predictions trained with at least 3 of the following algorithms: Linear Regression, Elastic Net, CARTs, Random Forest, and Boosting.
3. Final models. Here, you describe the two models, classification and regression, that you selected as your final submission in the competition. This subsection must include:
   - A detailed explanation of the final chosen models for evaluation in Kaggle. The answer must detail the model training, hyper-parameters selection, and any other relevant information.
   - A comparison to at least two (2) other specifications/models, for each approach.
   - A description of the variables used in the model and discuss their relative importance in the prediction.
   - A description of any sub-sampling strategy used to address class imbalances.

- Conclusions and recommendations. In this section, you state the main takeaways of your work.

# 2 Additional Guidelines

- Predictions have to be submitted on Kaggle. Check the competition website for more information.

- Turn a `.pdf` document in Bloque Neón. The document should not be longer than 10 (ten) pages and include, at most, 8 (eight) exhibits (tables and/or figures). Bibliography and exhibits don't count towards the page limit. You are welcome to add an appendix, but the main document must be self-contained. Specifically, a reader should be able to follow the analysis in the paper and be convinced it is correct and coherent from the main text alone, without consulting the appendix.

- The document must include a link to your GitHub Repository.
  - The repository must follow the template.
  - The README should help the reader navigate your repository. A good README helps your project stand out from other projects and is the first file a person sees when they come across your repository. Therefore, this file should be detailed enough to focus on your project and how it does it, but not so long that it loses the reader's attention. For example, Project Awesome has a curated list of interesting READMEs.
  - Include brief instructions to fully replicate the work.

  - The main repository branch should show at least five (5) substantial contributions from each team member.
  - The code has to be:
    * Fully reproducible.
    * Readable and include comments. In coding, like in writing, a good coding style is critical. I encourage you to follow the tidyverse style guide.

- Tables, figures, and writing must be as neat as possible. Label all the variables included. If you have something in your figures or tables, I expect they are addressed in the text. Tables must follow the AER format.