PREDICTING POVERTY

Edmundo Arias De Abreu
Economics Department
Universidad de los Andes

Lucía Maldonado
Economics Department
Universidad de los Andes

Juan Diego Heredia
Economics Department
Universidad de los Andes

October 21, 2024

ABSTRACT

We develop and test various machine learning models to predict poverty at the household level in Colombia using the 2018 GEIH survey. Implementing Elastic Net, Random Forest, XGBoost, and Neural Networks, we address challenges of missing data, class imbalance, and feature engineering. Through cross-validation and hyperparameter optimization using Weights and Biases, we enhance model performance. XGBoost emerges as the best predictor, achieving an F1 score of 0.666 on the Kaggle test set and 0.678 on our internal validation set. This model effectively handles mixed data types and missing values, with feature importance analysis revealing insurance and subsidy coverage and household dependency ratio as key predictors of poverty status. Our findings demonstrate the potential of machine learning, particularly XGBoost, in predicting poverty using survey and tabular data, offering insights for targeted poverty reduction strategies. However, discrepancies between internal validation and test set performance highlight the need for more robust cross-validation strategies in future research. This study contributes to data-driven approaches for addressing socioeconomic challenges and provides a promising tool for policymakers in poverty alleviation efforts.

1 Introduction

The prediction of economic phenomena has never been an exact science. In the macroeconomic area, for example, Prakash Loungani found in 2001 that predictions by economic development centers such as the World Bank and the International Monetary Fund failed to foresee 148 out of the 150 recessions that occurred in the 1990s (An et al., 2018).[1] In another research exercise, Andrew Brigdem found that, of the 469 recessions that took place over 30 years (1988-2018) in 194 countries, the International Monetary Fund predicted only 4 more than 6 months in advance (Brigden, 2019).[2] Likewise, Paul Samuelson's famous 1996 quote, "*the [US] stock market has forecast nine of the last five recessions*" is just one example of the skepticism that predictive methods have generated among researchers and policymakers worldwide.

One of the fundamental challenges of prediction is that it deals with non-linear relationships in which different variables can contribute to the occurrence of a phenomenon across units of time, individuals, and geographic units, among many others. Prediction methods, taking advantage of the increase in computational power in recent years, have made significant progress in determining the realization of different states across a broad range of areas of interest. In the United States, for example, *The National Oceanic and Atmosphere Administration* with its prediction tool *Atlantic Hurricane Outlook* has accurately predicted 52% of the hurricanes that occurred in the country over the past 24 years (Vaughan, 2024). In Afghanistan, the *World Food Program* has developed a prediction model using socio-economic and climatological data to improve the targeting of food aid packages to the communities most at risk of malnutrition in the country (WFP Asia & Pacific, 2024).

[1] Prakash Loungani was Budget Manager at the International Monetary Fund at the time of his study's publication.
[2] Andrew Brigdem was the Chief Economist of Fathom Consulting at the time of the study's publication.

Poverty prediction has been approached from multiple fronts, due to its importance even to this day in many parts around the world. Authors such as Mathiassen and Wold (2021) and Christiaensen et al. (2012) have developed income prediction methods based on consumption surveys and linear regression methods for Malawi and Vietnam, respectively. On the other hand, authors like Wong and Shuaibi (2024) have developed selection models for predicting binary poverty outcomes in Cambodia. All of these models aim to improve poverty predictions, as well as methods to enhance the targeting of programs and the monitoring of indicators, while reducing the 'measurement burden' that comes with data collection through surveys for generating these indicators. In this paper, Colombia is taken as a case study to contribute to the poverty prediction literature through *machine learning* and *Big Data* techniques that favor models capable of reducing the required measurement burden.

Our project is a competition on Kaggle (Predict Poverty). We are provided with a cross-section dataset that surveys thousands of Colombian households regarding their location, salary, composition, education attainment, subsidies, etc. Furthermore, we have at our disposal three supplementary datasets: test set at the household level[3]. To predict the poverty class we deploy a variety of machine learning models and algorithms, ranging from tree-based methods to neural networks. We display preliminary scores for all tested models but we focus and explain in greater detail and run experiments on the best-scoring models. All scripts, figures and experiment tracking can be found in this GitHub Repository.

2 Data

This document is based on data from the Monetary Poverty and Inequality Measurement mission by the National Administrative Department of Statistics (DANE) and the National Planning Department. This mission was launched in 2006 with the aim of facilitating the comparability and reporting of employment, poverty, and inequality series in the country. The reports from this mission are available at both the household and individual levels. The predictive model in this document leverages primarily the individual-level reports, although some household-level variables are also incorporated. The cleaning process and key data features are described in sections 2.1 and 2.2.

2.1 Data Cleaning

The data were divided into two sections provided by the *Kaggle* competition: train and test. Each dataset was available in both individual and household-level versions, which could be joined using the household ID as the key. The cleaning process was performed on the raw individual-level datasets, rather than the household-level datasets, due to their higher granularity. All cleaning steps applied to the raw train dataset were also applied to the raw test dataset to ensure that all variables used to train the model were available for prediction in both datasets. This section describes the cleaning process applied to these datasets to prepare them for model training and prediction.

The raw training dataset consisted of a total of 543,109 observations and 135 descriptive variables, while the raw test dataset contained 219,644 observations and 63 variables. The data correspond to the 2018 cross-section. Since the training data included more variables than the test data, the training dataset was transformed to retain only the variables present in both datasets. Then, 19 variables were created by taking individual-level data and transforming it into household-level data. These variables capture various characteristics of households and individuals, including the number of members, employment status, pension coverage, and housing conditions.[4] For instance, *num_individuos_menores_18* was constructed by counting the number of individuals within a household with a reported age under 18 and dividing it by the total number of individuals in the household. Each of these variables played a crucial role in providing a comprehensive view of household dynamics, which was absent in the dataset at the household level.

The variables created can be grouped into seven categories: household education level, which captures the educational attainment of household members; work-based features, which include employment status, unemployment rates, and activity diversity within the household; subsidies, which reflect whether the household receives government support, such as food or transport subsidies; pension contribution, which measures the presence of contributors or pensioners within the household; household size, indicating the number of people in the household; dependency ratio, which measures the proportion of dependents (children and elderly) to working-age members; and health insurance coverage, which captures the presence, type, and extent of health insurance within the household.

[3]Henceforth refereed to as 'Test Set' will be used to predict poverty and enter submissions on Kaggle

[4]The variables created include: *educ_attainment*, *P6240_unemployment_rate*, *P6240_main_household_activity*, *P6240_activity_diversity*, *total_subsidies*, *has_pension_contributor*, *prop_pension_contributors*, *prop_pensioners*, *pension_status*, *dependency_ratio*, *has_health_insurance*, *main_insurance_type*, *insurance_coverage_rate*, *num_individuos*, *num_individuos_menores_18*, *num_individuos_mayores_65*, *jefe_mujer*, *share_empleado_privado*, *alimentos*.

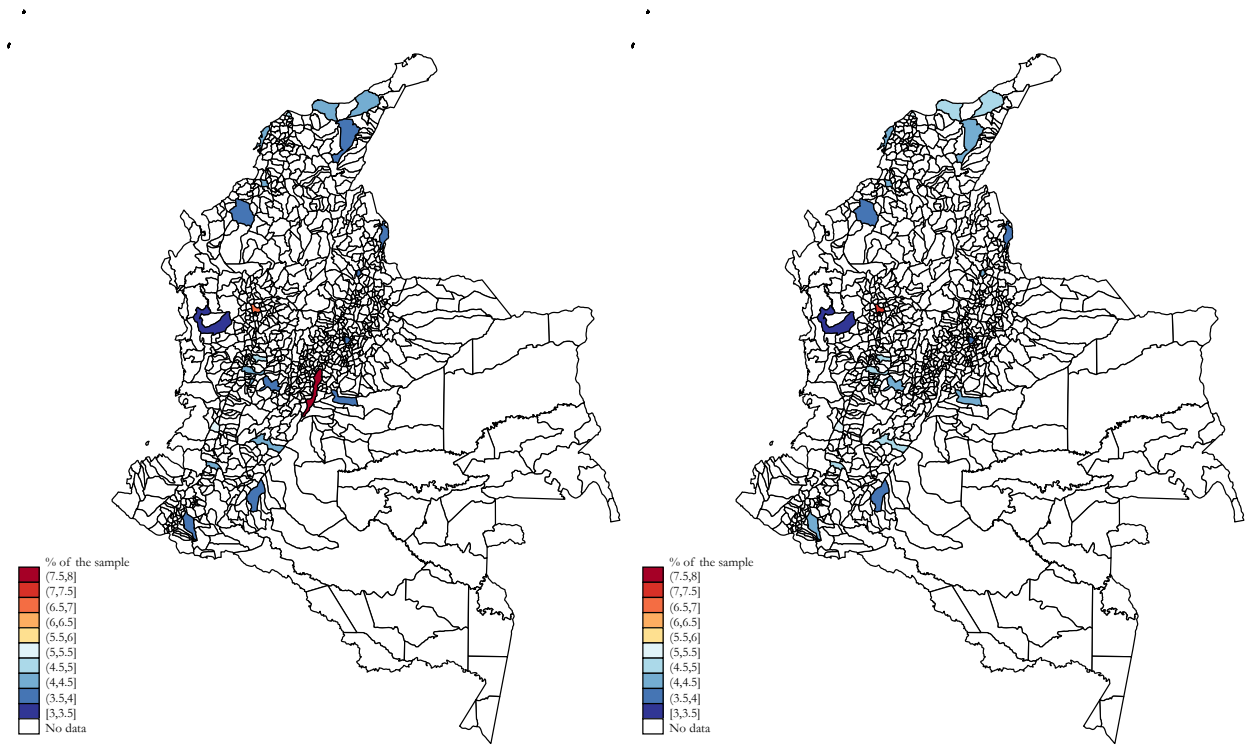
Once these variables were created, the raw train and test datasets were collapsed to the household level and merged with the original dataset, which includes the poverty indicator (the target variable). The number of individuals per room was calculated using the available data on household members and rooms. With this, the cleaning process was completed, and the data were ready for model training and testing.

2.2 Descriptives

Data from the Monetary Poverty and Inequality Measurement (MPIM) include reports for 24 departments and 13 metropolitan areas in the training set, and 23 departments and 12 metropolitan areas in the test set (see figure 1). The only city of train data missing in test is Bogotá. Figure 1 presents the share of households in each municipality relative to the total number of households in each dataset. The share of household per municipality varies from 4% to 10% in the train set, while the concentration of households in test set is slightly minor spanning from 1.7% to 3.8%. Although the range of the test set is lower, municipalities in the test set have have a higher concentration of in share above 4%, than the train set.

Figure 1: Share of Households per Municipality (2018)

(a) Train Set (b) Test Set



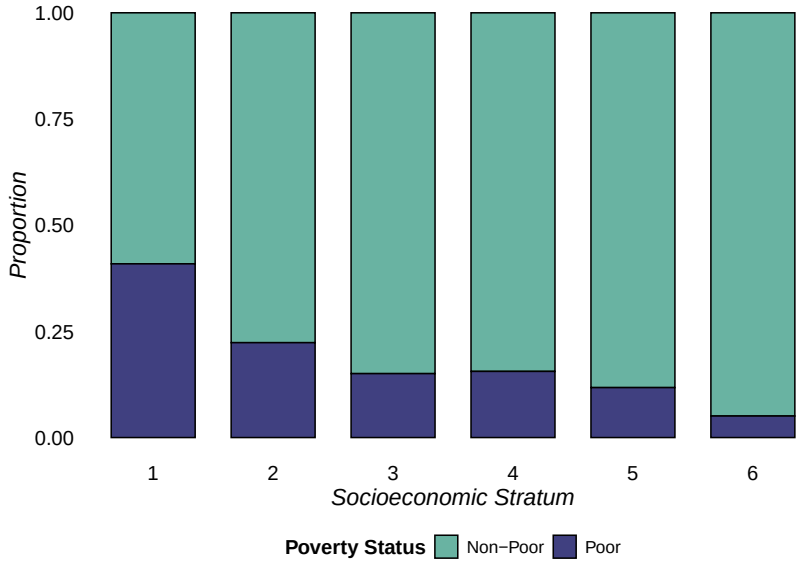
Note: The figure shows the share of households per municipality. Data were retrieved from the Monetary Poverty and Inequality Measurement, 2018.

Figure 2 presents the share of poor and non-poor individuals across the six socioeconomic strata in Colombia. The results show that there is no deterministic relationship between poverty and this socioeconomic classification. In particular, less than half of the people in stratum 1 are poor (40.9%), while 5.07% of people in stratum 6 are poor. These numbers closely follow the report offered by the Ministerio de Hacienda in 2014, which stated that 48% of stratum 1 households in Colombia are non-poor (Ruiz Granados, 2014). Given the decreasing correlation between these two variables, including socioeconomic stratum seems important in predicting income; however, since this variable is not available in the test set, this is not possible.

The share of households in the training and test sets with a female head, and the share of households with a household head who has a university education, depending on the gender of the household head, is presented in Figure 3. Less than half of the households in both datasets have a female head (40.3% in training and 40.5% in test); however, households with a female head are more likely to have a household head with a university degree. In particular, the share of households with a female household head who has a university education relative to the total number of households

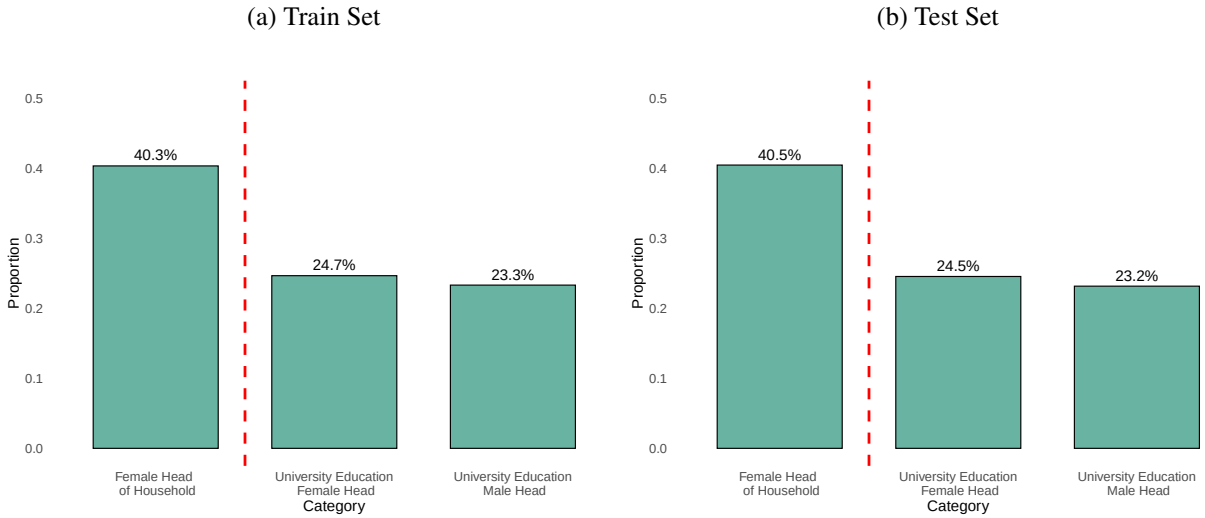
with female heads is higher (24.7% in training, and 24.5% in test) than the same share for men (23.3% in training, 23.2% in test). These results are consistent with the most recent DANE report from the Encuesta de Calidad de Vida, which stated that 4 out of 10 households in Colombia have a female head (DANE, 2024). Given that female household heads often face experiences such as forced displacement and bear a higher burden of non-remunerated care labor, with more unstable and low-income jobs, the gender of the household head is a relevant predictor of poverty.

Figure 2: Share of Poor and Non-Poor by Socioeconomic Strata



Note: The figure displays the share of poor and non-poor across socioeconomic strata in the training data. Data were retrieved from the Monetary Poverty and Inequality Measurement, 2018.

Figure 3: Households by Female Head and University Education



Note: The figure shows the share of households with a female head, as well as the share of households with at least one member with university education, depending on whether the household head is female. The red line separates the female head indicator from the university education bars. Data were retrieved from the Monetary Poverty and Inequality Measurement, 2018.

Table 1 compares the training and test datasets for the 12 main variables constructed from the dataset at the individual level. The variables have a low share of missing values since the cleaning process replaced missing values with zeros. The underlying idea was that non-reporting a variable was a signal of not falling within the category. For instance, it was assumed that if a household did not have any individual reporting being employed in private companies, then nobody in the household actually had that type of employment. The report shows that, on average, households in the

training data are composed of 4.25 individuals, with 1.37 members under 18 and 0.31 over 65 years. Additionally, the average share of household members employed is 47.25%, with 10.54% employed in private companies, 20.74% self-employed, and 2.56% employed in public companies. Only 11% of families in the training set receive food as part of their payment. Finally, the average total number of hours worked weekly by the household is 81.41, which is roughly the legal weekly number of hours worked by two individuals in Colombia. Differences in variables like *Number of individuals, Female head of household*, and *Dummy: Household has someone over 65* across training and test datasets are minor, as indicated by their p-values, suggesting that these differences are not statistically significant. Although several variables show statistically significant differences across groups, they can be considered economically unimportant since the magnitude of the differences is small. The largest statistically significant difference corresponds to the percentage of self-employed per household, with a 5-percentage-point difference across groups.

Table 1: Summary Statistics for Train and Test Datasets

Variable	Train			Test			P-value
	Non-missing (%)	Mean	Std Dev	Non-missing (%)	Mean	Std Dev	
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Number of individuals	100.00	4.25	2.09	100.00	4.29	2.10	0.000
Number of individuals under 18	100.00	1.37	1.35	100.00	1.40	1.36	0.000
Number of individuals over 65	100.00	0.31	0.60	100.00	0.31	0.61	0.197
Dummy: Household has someone over 65	100.00	0.24	0.42	100.00	0.24	0.43	0.012
Female head of household	100.00	0.40	0.49	100.00	0.40	0.49	0.323
Someone in the household has university education	90.03	0.57	0.50	89.88	0.57	0.50	0.020
Percentage of the household employed	99.99	47.25	29.48	99.99	47.04	29.23	0.005
Percentage employed in private companies	99.99	19.54	26.28	99.99	18.59	25.64	0.000
Percentage employed in public companies	99.99	2.56	11.00	99.99	2.56	10.96	0.862
Percentage self-employed	99.99	20.74	25.93	99.99	21.36	25.92	0.000
Received food as part of payment	100.00	0.11	0.31	100.00	0.11	0.31	0.101
Total hours worked by the household (weekly)	100.00	81.41	55.64	100.00	81.59	55.71	0.224

Note: The table presents summary statistics for the main 14 variables constructed in the train and test datasets at the individual level. The table is at the household level and includes the share of non-missing entries, the mean, and the standard deviation for each of these variables. Additionally, column (8) presents the p-value of a t-test comparing the mean of the same variable across the train and test datasets.

3 Models

3.1 Elastic Net

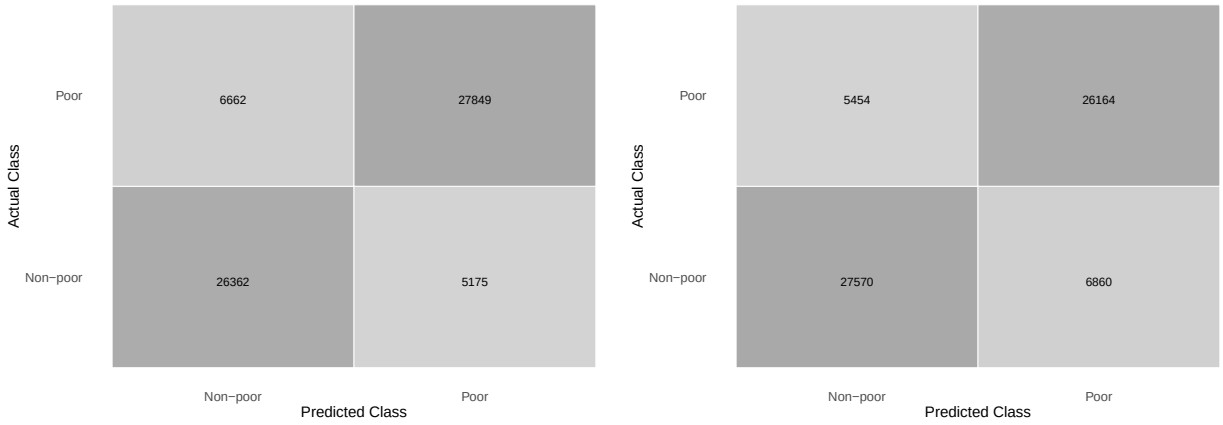
The first model implemented consisted of a standard Elastic Net with a **predetermined** mixing parameter (α) of 0.5, intended to weight equally the LASSO and RIDGE regularization in the model, and a penalization parameter (λ) selected by minimizing the cross-validation error. This model included as predictors the 19 variables constructed from the raw datasets at the individual level, described in section 2.1. Importantly, the Elastic Net model implemented aimed to address the **class imbalance in the dependent variable**, where 80% of households were non-poor and only 20% were classified as poor in the training data. Specifically, **undersampling** was conducted to randomly remove households in the dominant class until an equal number of households in each class was ensured. This sought to resolve the issue of models becoming biased toward the majority class, which can lead to poor performance in identifying the minority class and result in misleading accuracy and poor predictive power for rare but important cases. Additionally, the data used in the model underwent **median imputation**, where missing values were replaced with the median of each respective variable, ensuring that the central tendency of the data was maintained without being influenced by outliers.

This first Elastic Net model produced an F1 score of 0.82 on the training dataset and an F1 score of 0.63 on the test set available in *Kaggle*. Panel (a) of Figure 4 presents a confusion matrix of the results over the training data. The first

model, using a **Bayesian classifier** (e.g., a threshold of 0.5), correctly classified as poor 84.3% of all those predicted as poor (precision), and 80.7% of all actual poor (recall). As expected, the F1 score decreased on the test data. However, the predictions were **13 percentage points above random guessing**.

Figure 4: Elastic Net Confussion Matrices

(a) First Model (b) Final Model



Note: Data were retrieved from the Monetary Poverty and Inequality Measurement, 2018.

The final Elastic Net (EN) model aimed to increase the F1 score through **fine-tuning** parameters without changing the predictor variables.[5] In particular, this Elastic Net model calculated the mixing parameter (α) endogenously, as well as the optimal threshold. To select the best mixing parameter, the final EN model implemented an iterative approach using the *caret* package in R, modifying the mixing parameter from 0.05 to 0.95 and selecting the model with the lowest cross-validation error. In this case, the optimal mixing parameter was 0.08, indicating that the optimal Elastic Net was **closer to a Ridge regression**. This could be due to the presence of several moderately strong predictor variables instead of a few predominant ones. Once the model was trained, a new threshold was established by minimizing the F1 score in the validation set. The obtained value was slightly higher than the Bayes classifier with an optimal threshold of 0.53. The final model achieved a precision of 79.2%, a recall of 82.7, and an F1 score in the training dataset of 80.9%. Although the F1 score in the training-validation set was lower, the F1 score in the test set **increased by one percentage point** to 0.64. This is an example of a case where having a higher F1 score in the training set does not necessarily mean a decrease in the F1 score in the test set.

3.2 Random Forest

The first tree-based method we implemented was random forest in order to predict poverty status at the household level. Random Forest is a learning method that works simply by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) of the individual trees. To briefly describe how this model works, we first begin by what's known as *bagging* (bootstrap aggregating)– that is, creating multiple subsets of the original dataset through random sampling with replacement. For each subset a decision tree is constructed. In our context, we can think of a tree in our forest as a "vote" for a class– the final prediction being the majority vote across all trees.

A nice trait of this type of model is that it provides a measure of feature importance, helping identify which variables are most crucial in predicting poverty status at the Colombian household level. We will dive deeper on feature importance when we get to XGBoost models because they performed better. Using random forest we set up the following training and evaluation:

1. Handle missing values using median imputation for numeric features and mode imputation for categorical features. The target variable 'Pobre' takes the value of 1 if a household in our train data is poor, 0 otherwise.
2. Apply undersampling to address class imbalance, ensuring an equal representation of poor and non-poor households in the training data.

[5]The imputation and undersampling remained unchanged across the first and last models.

3. We use 5-fold cross validation to asses model performance while keeping an eye on over-fitting.

4. For each fold, we train a Random Forest model on the undersampled data.

5. We evaluate the model using AUC and F1 score metrics on the validation (or hold-out) set.

Table 2: Hyperparameters of the Random Forest Model

Hyperparameter	**Value**	**Description**
Number of Trees	500	Number of decision trees in the Random Forest ensemble
Impotrance	'impurity'	Method for calculating feature importance
Probability	TRUE	Model outputs class probabilities instead of hard classifications

Table 2 displays the hyperparameters used when training and then predicting using our Random Forest models. As can be seen in 5, the random forest model was not our best-scoring model. Hence, we abstained from further experimentation and fine-tuning. For tree-based models such as this one, we believed (and were later proved correct) that XGBoost would outperform Random Forest, therefore we prioritized it.

3.3 XGBoost

As we mentioned, one of the key models we implemented was XGBoost (eXtreme Gradient Boosting), a decision tree-based algorithm that uses the boosting approach. XGBoost is a supervised learning method that combines multiple weak models, in this case, decision trees, to create a more robust final model. The core idea behind this approach is to improve predictions sequentially, building each tree with the goal of correcting the prediction errors made by the previous trees. This process ensures that the final model is more accurate, generalizes better to unseen data and is less prone to overfitting.

Unlike other approaches such as *bagging*, where weak models are trained independently and their predictions are averaged, boosting is a sequential technique where each new tree is built to correct the errors of the previous ones. XGBoost, in particular, focuses on minimizing a loss function by using the negative gradient of the loss as a measure of the errors made, thus adjusting predictions with iterative precision. Each tree aims to focus on those examples that were poorly predicted by the previous trees, allowing the model to learn from its mistakes and improve its performance with each iteration.

For the XGBoost training process, we followed these steps: First, missing values were either dropped or imputed; numerical features were imputed using the median, while categorical features were imputed using the mode. Additionally, we applied the 'SMOTE' oversampling technique to balance the representation of poor and non-poor households, as the dataset exhibited a strong class imbalance. This allowed the model to access a balanced dataset, thereby improving its ability to correctly identify poor households. Next, we scaled the numerical variables using *StandardScaler*, while categorical variables were transformed using *one-hot encoding* to ensure that all features were properly handled.[6]

We used 5-fold stratified cross-validation that maintained the target variable distribution, which allowed us to split the dataset into subsets and train the model on different combinations of training and validation data, ensuring consistent results and minimizing overfitting. The XGBoost model was trained with initial parameters optimized through weights and biases (W&B), using the F1 score as the main evaluation criterion. The F1 score is especially useful in this context due to the imbalanced nature of the dataset, as it balances precision and recall to provide a more representative measure of the model's overall performance.

Below are the best parameters found in this iteration:

[6]Although XGBoost can handle raw features, scaling numerical variables helps prevent features with larger magnitudes from dominating the model's decisions, ensuring fair feature importance. One-hot encoding categorical variables allows XGBoost to effectively leverage categorical information, as it transforms discrete categories into a format that the model can process.

Figure 5: XGBoost Feature Importance

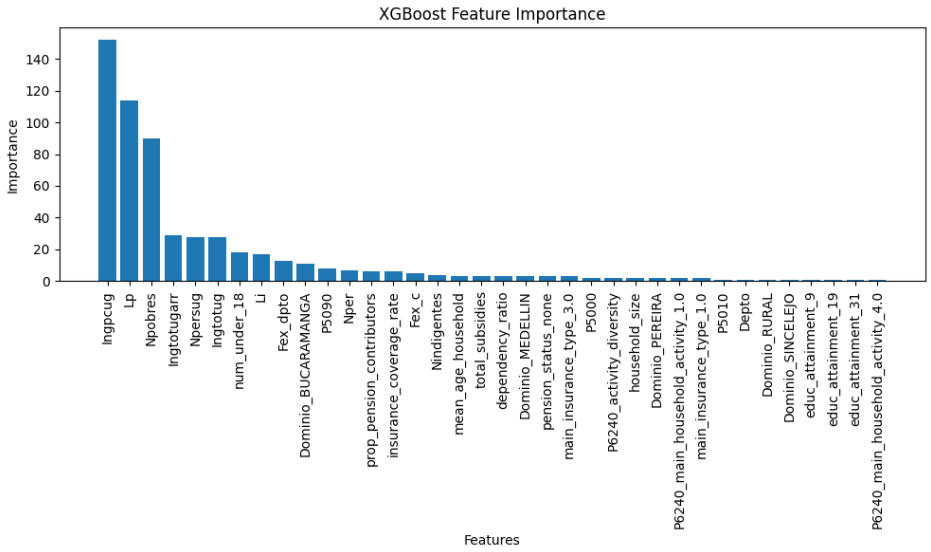


Table 3: XGBoost Model Hyperparameters

Hyperparameter	**Value**	**Description**
max_depth	15	Maximum depth of each tree
eta	0.0803	Learning rate for incremental tree adjustment
subsample	0.643	Percentage of data used in each iteration
colsample_bytree	0.5509	Proportion of features used by each tree
n_estimators	210	Number of trees in the boosting ensemble
scale_pos_weight	adjusted	Adjusted factor to balance classes

At the end of the cross-validation process, the average F1 score was 0.621, positioning XGBoost as one of our best-performing models for poverty prediction. The feature importance analysis highlighted that numerical variables related to insurance coverage and household dependency ratio were the most relevant for predicting poverty status.

Table 5 displays the most features that the XGBoost model found to be most predictive of poverty. Some of these come as no surprise: total income of the household, number of people, poverty and homelessness thresholds, and more. Other features which were engineered using the person-level datasets resulted important as well such as mean age of household, pension status, dependency ratio and level of education attainment. This highlights the importance of enriching our initial dataset.

3.4 Hyperparameter Optimization with Weights and Biases (W&B)

To further improve the performance of our XGBoost model, we used the *Weights and Biases* (W&B) platform, a tool for monitoring and optimizing machine learning models. W&B facilitates the automatic search of hyperparameters through a process known as *Bayesian Optimization*, which explores the hyperparameter space more efficiently compared to manual or random search.

The optimization process focused on maximizing the F1 score, which is particularly useful in imbalanced datasets like the one we were using. For this, the following parameter ranges were explored: the learning rate was adjusted between 0.01 and 0.2, the number of estimators varied between 100 and 300, and the maximum tree depth was optimized in a range between 3 and 15. Additionally, different values of *subsample* between 0.5 and 1.0 were tested, and the proportion of columns used by each tree (*colsample_bytree*) was also optimized within the same range. The *min_child_weight* parameter, which controls the minimum number of observations required for a leaf node, was adjusted between 1 and 6. Finally, the classification threshold for converting prediction probabilities into classes was optimized between 0.3 and 0.6.

To implement this optimization, we used W&B's *sweeps* functionality, which automates hyperparameter exploration and allows different configurations to be evaluated efficiently. Each hyperparameter combination was trained and evaluated using a validation set, and the results were monitored in real time through the W&B interface.

Throughout this process, W&B allowed us to visually compare the performance of different hyperparameter configurations and select the best one based on the F1 score. By the end of the optimization process, the optimal parameter combination was identified, resulting in a significant improvement in the model's performance. We tracked and reported models in the following W&B Report.

3.5 Neural Networks

The fourth and final model we implemented was a deep neural network model. We thought this type of model would increase our test performance because it is particularly adept at learning complex patterns and relationships within high-dimensional data, making it well-suited for the dataset at hand. In this section we now describe its architecture, training and evaluation loops, hyperparameters and tuning procedure.

3.5.1 Architecture

Our neural network, named *PovertyPredictor*, consists of a feed-forward architecture consisting of three hidden layers. The model takes in a combination of numeric and categorical features (all features from our dataset except 'id'), processed through a custom preprocessing pipeline that scales and integrates both numeric and categorical features. This pipeline handles missing data imputation, standardization of numeric features, and one-hot encoding of categorical variables, ensuring that all input data is appropriately formatted for the neural network and no data is lost or improperly handled and could hinder our predictions.

Specifically, the architecture of our neural network has an input layer whose size is determined by the number of processed features, three hidden layers with ReLU activation functions, batch normalization, and dropout for regularization.[7] Finally, an output layer with two neurons, corresponding to our binary classification task. The model processes the input data through these layers, learning increasingly abstract representations of the features at each level. The final output is passed through a softmax function to produce probabilities for each class (poor or not poor).

3.5.2 Training

For training, we employed the Adam optimizer, which adaptively adjusts learning rates for each parameter, facilitating efficient convergence. We used cross-entropy loss as our objective function, which is standard for binary classification tasks. To address class imbalance in our dataset, we applied the Synthetic Minority Over-sampling Technique (SMOTE) to augment the minority class.

Our training loop incorporated early stopping to prevent overfitting. We monitored the validation F1 score and saved the best-performing model weights. With this approach we hoped that we would select a model that generalizes well to unseen data.

Table 4: Key Hyperparameters of Neural Network Model

Hyperparameter	**Value**
Batch size	32
Hidden layer 1 size	256
Hidden layer 2 size	128
Hidden layer 3 size	32
Dropout rate	0.159
Learning rate	0.000654
Weight decay	1.27e-5

Table 4 displays the initial hyperparameters used for base training of the model. We would proceed to choose the best-performing hyperparameters (fine-tuning) using Weights & Biases (W&B) for it helped us track experiments, manage and version datasets, and collaborate on different models tested.

We leveraged the power of this platform to find what combination of hyperparameters– given a fixed hyperparamet space– led to a higher F1 score on the validation set. The hyperparameter tuning process begins with defining a 'sweep' configuration, that is, a possible set of values that each hyperparameter of Table 4 could take. Next, for each combination (called run) we ran the model through our training pipeline and logged the validation F1 score. This hyperparameter tuning process using W&B allows for a systematic and efficient exploration of the hyperparameter space. It automates the process of trying different configurations and helps identify the optimal set of hyperparameters for our task of poverty prediction. For a detailed report and showcase refer to this link Neural Network Tuning (W&B).

[7]The respective number of neurons per hidden layer are 256, 128 and 32 before fine-tuning

4 Results

Table 5: Results: F1 Scores for Kaggle Submission and Internal Validation

Model	F1 Score	
	Kaggle Submission	**Internal Validation**
(Base Models)		
Elastic Net (base)	0.636	0.825
Random Forest (base)	0.535	0.603
XGBoost (base)	0.607	0.656
Neural Network (base)	0.535	0.655
(Fine-tuned Models)		
Elastic Net (fine-tuned)	0.643	0.809
Random Forest (fine-tuned)	0.535	Not Fine-Tuned
XGBoost (fine-tuned)	0.666*	0.678
Neural Network (fine-tuned)	0.627	0.636

Table 5 highlights the main results of our models. We report the scores on the Kaggle submission, which is calculated based on the 20% of the test data, and in our internal validation loops, where we split our training data into folds (for tree-based methods) or 70-30 splits. As we see, base models evidence high variability in their scores, but their F1 scores start converging once fine-tuning is done. Our best score is attributable to the XGBoost model (both in Kaggle submission and internal validation). This is unsurprising when we know these types of model tend to outperform others when it comes to tabular data. Our dataset contained a mix of continuous and categorical features. XGBoost handles mixed data types well without extensive preprocessing (as in the case of neural networks). Furthermore, XGBoost is generally more robust to outliers and can handle missing data natively. Our data contained a vast number of missing values for key variables. Finally, our training dataset consisted of approximately 160,000 observations which is not large enough for the neural networks to make a serious difference in terms of prediction accuracy. XGBoost often performs well on small to medium-sized datasets, which are common in socioeconomic studies. Neural networks, on the other hand, typically require larger datasets to fully leverage their capacity to learn complex patterns.

Another interesting insight from Table 5 is the disparity between Kaggle scores and our internal validation set. Because we didn't have access to the test set's targets, our best way to get a sense of how well the trained model would generalize to this unseen data was to construct a validation set from the training data. The disparity is proof that the validation sets constructed were not very similar to the test data. For further competition it is key to build a robust validation set so that it more accurately reflects the characteristics of the unseen test data.

5 Discussion on Best Model & Conclusion

Our paper reveals that the XGBoost model consistently outperformed other approaches in predicting poverty status at the household level in Colombia. After fine-tuning, the XGBoost model achieved an F1 score of 0.666 on the Kaggle submission and 0.678 on our internal validation set, marking it as our most robust and accurate predictor. The superior results can be attributed to a number of different factors, including but not limited to its handling of mixed data types, feature importance determination, dataset size and structure.

This study demonstrates the efficacy of machine learning techniques, particularly XGBoost, in predicting poverty status at the household level using survey data from Colombia. Our best-performing model achieved an F1 score of 0.666 on unseen test data, outperforming random guessing and providing a valuable tool for poverty assessment. With this appraoch we are also able to inform policy-makers on which variables are the most relevant when it comes to predicting if a household is poor or not. The success of XGBoost in this context highlights its potential for similar socioeconomic prediction tasks, especially when dealing with mixed data types, missing values, and moderate-sized datasets. Furthermore, while XGBoost performed best in our comparison, the relatively close performance of other models (particularly the fine-tuned Elastic Net and Neural Network) suggests that ensemble methods combining multiple model types could potentially yield even better results.

However, the discrepancy between our internal validation scores and Kaggle submission scores across all models underscores the importance of robust cross-validation strategies that accurately reflect the characteristics of unseen data. Future work (or in our case, problem sets) could focus on improving the representativeness of validation sets to enhance model generalization *prior* to any model engineering and deployment. In conclusion, this study not only provides a practical tool for poverty prediction but also demonstrates the value of machine learning in informing evidence-based

policy-making in the realm of socioeconomic development. Going forward it is key to balance ‘data-based’ solutions, where the emphasis is on enriching the dataset, and ‘model-based- solutions where the aim is to build a more robust and accurate model.

References

An, Z., Jalles, J. T., & Loungani, P. (2018, March). *How well do economists forecast recessions?* (Tech. rep. No. WP/18/39). International Monetary Fund. https://www.imf.org/en/Publications/WP/Issues/2018/03/08/How-Well-Do-Economists-Forecast-Recessions-45882

Brigden, A. (2019). The economist who cried wolf? [Accessed: [insert date of access]]. *Fathom Consulting*. https://www.fathom-consulting.com/the-economist-who-cried-wolf/#_ftn3

Christiaensen, L., Lanjouw, P., Luoto, J., & Stifel, D. (2012). Small area estimation-based prediction methods to track poverty: Validation and application. *The Journal of Economic Inequality*, *10*, 267–297.

DANE. (2024). Cuatro de cada 10 hogares en el país tienen a una mujer como cabeza de la familia [Accessed: 2024-10-20]. *La República*. https://www.larepublica.co

Mathiassen, A., & Wold, B. K. G. (2021). Predicting poverty trends by survey-to-survey imputation: The challenge of comparability [Advance Access Publication Date: 20 June 2021]. *Oxford Economic Papers*, *73*(3), 1153–1174. https://doi.org/10.1093/oep/gpab014

Ruiz Granados, C. (2014). Entre los estratos 1, 2 y 3 hay escondidos 1,3 millones de hogares ricos [Accessed: 2024-10-20]. *La República*. https://www.larepublica.co

Vaughan, L. (2024). How accurate are noaa hurricane forecasts? [Accessed: [insert date of access]]. *Predict*. https://medium.com/predict/how-accurate-are-noaa-hurricane-forecasts-5bf6907358d1#:~:text=These%20summaries%20provide%20the%20predicted,range%2052%25%20of%20the%20time.

WFP Asia & Pacific. (2024). Predicting hunger to prevent famine: How early warnings help wfp respond in afghanistan [Accessed: [insert date of access]]. *Medium*. https://medium.com/@WFP_Asia_Pacific/predicting-hunger-to-prevent-famine-how-early-warnings-help-wfp-respond-in-afghanistan-8e9da6c71fb9

Wong, G., & Shuaibi, A. (2024). Model selection and optimization for poverty prediction on household data from cambodia. *Journal of Emerging Investigators*, *10*, 1–10. https://www.emerginginvestigators.org/articles/model-selection-and-optimization-for-poverty-prediction-on-household-data-from-cambodia