Universidad de los Andes

Big Data y Machine Learning

Ignacio Sarmiento-Barbieri

# Problem set 2 - Predicción de la pobreza

Repositorio: github.com/GeorgeWton1986/T2_BDML

*Elaborado por:*
Laura Sarif Rivera Sanabria
Jorge Eliecer Viafara Morales
Nicolas Jacome Velasco
Zaira Alejandra Garcia
Bernal

13 de abril de 2025

# 1. Introducción

La pobreza es un desafío político y social crucial en muchos países vulnerables, donde suele medirse mediante el gasto de consumo (Galvis Caballero, 2021). Factores como la residencia regional y el nivel educativo son esenciales al diseñar intervenciones para mejorar los medios de vida (Okiabera, J. O., 2020). Para enfrentar la pobreza, se han implementado subsidios y transferencias monetarias. Sin embargo, los programas sociales universales también benefician a personas que no necesitan ayuda, impulsando el uso de pruebas de medios indirectas (PMT) basadas en variables socioeconómicas como el género y las condiciones de vivienda (Brown, Ravallion & van de Walle, 2018).

Este estudio implementó siete algoritmos de clasificación: Red Elástica, Logit, Random Forest, Regresión lineal, Boosting, Naive Bayes y CART para predecir la pobreza en Colombia, a través de variables socio económicas recolectadas por medio de la encuesta denominada Medicion de Pobreza Montenaria y Desigualdad del año 2018 donde los mejores resultados se obtuvieron mediante Red Elástica, la información empelada se encuentra en dos bases de datos, una de entrenamiento (Train) y la otra de prueba (Test). La estructura del documento incluye limpieza de datos, análisis de variables, predicción de pobreza y conclusiones, proporcionando una visión integral de los factores que influyen contextos de vulnerabilidad.

# 2. Datos

## 2.1 Construcción de la muestra

La metodología se presentará conforme su desarollo en R studio.

1. Incluir los paquetes necesarios en R, utilizando la función `require("pacman")` y `p_load()` para facilitar la instalación y gestión de herramientas esenciales para la manipulación y análisis de datos. Se emplean paquetes como `tidyverse` para la transformación de datos, `glmnet` para la implementación de algoritmos de regularización, `caret` para la creación de modelos predictivos, y `MLmetrics` para la evaluación del desempeño de los modelos.

2. Permitir que las bases de datos `train_hogares.csv`, `train_personas.csv`, `test_hogares.csv` y `test_personas.csv` se almacenen en un repositorio que permita autonomía en el cargue y descargue de la información.

3. Iniciar la inspección y el análisis exploratorio de las columnas y la selección de variables clave en `train_hogares`, como el identificador del hogar (`id`). Se creó la variable `pobre_hand`, que asigna a los hogares en pobreza el indicador (1) y a los no pobres el indicador (0) en función del ingreso per cápita y la línea de pobreza (`Lp`). En el caso de `train_personas`, se procesaron los datos mediante la creación de variables como `mujer` (indicador de género), `mayor_edad` (mayores de 18 años), `afi_salud` (afiliación a salud) y `EducLevel` (nivel educativo). Posteriormente, se filtran los registros del jefe de hogar para capturar características específicas de este, como su nivel educativo y estado laboral, y se unen con la base agregada por hogar.

4. Limpiar `train_hogares` generando nuevas variables indicadoras sobre la tenencia de la vivienda (pagada, hipoteca, arriendo, usufructo, sin titulo). Luego, `train_hogares` y `train_personas` se combinan mediante el identificador `id`, consolidando la información en una única base de entrenamiento (`train`) y una de prueba (`test`).

5. Convertir las variables clave en factores categóricos, como el dominio geográfico, el tipo de tenencia de la vivienda y los niveles educativos. Este procesamiento asegura que los datos sean adecuados para el análisis y modelado posterior, facilitando la interpretación de los resultados y la construcción de modelos predictivos.

## 2.2 Análisis descriptivo

La variable objetivo "Pobre" indica que aproximadamente el 20 % de la población ecuestada se encuentra en condición de pobreza. Este hallazgo no solo es coherente con las cifras nacionales de pobreza, sino también refuerza la importancia de tratar este desbalance en la muestra `train` en los modelos predictivos utilizados en este documento. Ahora bien, se presenta el siguiente análisis descriptivo de las variables empeladas.

Cuadro 1: Estadísticas descriptivas de las variables

| # | Variable | Media | Mediana | Min | Max | Des.st |
|---|---|---|---|---|---|---|
| 1 | Pobre | 0.20 | 0 | 0 | 1 | 0.40 |
| 2 | nmujeres | 1.74 | 2 | 0 | 14 | 1.18 |
| 3 | nmayor_edad | 2.37 | 2 | 0 | 19 | 1.20 |
| 4 | nafi_salud | 2.71 | 2 | 0 | 19 | 1.40 |
| 5 | H_Head_Educ_level | 4.37 | 5 | 0 | 6 | 1.40 |
| 6 | maxEducLevel | 5.12 | 5 | 0 | 6 | 1.11 |
| 7 | H_cotizando | 1.12 | 1 | 0 | 2 | 0.83 |
| 8 | pagada | 0.38 | 0 | 0 | 1 | 0.48 |
| 9 | hipoteca | 0.03 | 0 | 0 | 1 | 0.18 |
| 10 | arriendo | 0.39 | 0 | 0 | 1 | 0.49 |
| 11 | usufructo | 0.15 | 0 | 0 | 1 | 0.36 |
| 12 | sin_titulo | 0.05 | 0 | 0 | 1 | 0.21 |
| 13 | Clase | 1.09 | 1 | 1 | 2 | 0.29 |
| 14 | Nper | 3.29 | 3 | 1 | 28 | 1.77 |
| 15 | Lp | 271522.31 | 279944.53 | 167222.48 | 303816.69 | 33656.89 |
| 16 | H_Head_mujer | 0.42 | 0 | 0 | 1 | 0.49 |
| 17 | H_afi_salud | 1.00 | 1 | 0 | 1 | 0.00 |
| 18 | H_Head_ocupado | 0.71 | 1 | 0 | 1 | 0.45 |
| 19 | nt_trab_semanal | 1.50 | 1 | 0 | 14 | 1.03 |
| 20 | ncotizando | 2.39 | 2 | 0 | 21 | 1.83 |
| 21 | nocupados | 1.50 | 1 | 0 | 14 | 1.03 |

La estructura del hogar de acuerdo con la composición demográfica y la dependencia, demuestra que en promedio el tamaño del hogar `nper` es de 3.3 personas junto con una alta proporción de adultos y mujeres. Por otra parte, la relación del vínculo con el sistema de salud y el mercado laboral permite un acercamiento al sistema de protección social, donde el número promedio de afiliados por hogar es relativamente alto y el número de cotizantes es significativamente menor debido al régimen de seguridad social de salud.

Adicionalmente, la tenencia de vivienda y patrimonio del hogar es un indicador de la estabilidad económica, de este modo, se observa que las modalidades de arriendo y vivienda pagada son más frecuentes, no obstante, la presencia de la modalidad sin título y usufructo demuestra debilidades en el acceso formal a un inmueble por parte de la familias. Por último, la educación del jefe de hogar y el nivel máximo educativo alcanzado por el hogar aportan en la comprensión del capital

humano, además de permitir un acercamiento a posibles análisis estructurales de pobreza. A nivel agregado, los hogares presentan niveles medio de educación, sin embargo, hay casos sin ningún nivel educativo, limitando oportunidades laborales en el sector formal, posible aumento del gasto público ante otorgamientos de subsidios, así como un posible aumento de la base de pobreza.

# 3. Modelos y resultados

Esta sección describe los modelos utilizados para la predicción con modelos entrenados en siete algoritmos de clasificación: Red Elástica, Logit, Random Forest, Regresión lineal, Boosting, Naive Bayes y CART.

## 3.1 Red elástica

Elastic Net es una método de regularización que combina las ventajas de Lasso (L1) y Ridge (L2), ayudando a mejorar la predicción y la selección de variables en contextos con muchas variables correlacionadas. El modelo se ajusta mediante los parámetros `alpha` y `lambda`, donde `alpha` controla la mezcla entre Lasso y Ridge, y `lambda` ajusta el grado de penalización. Esta combinación reduce el sobreajuste y mejora la interpretabilidad del modelo.

Para entrenar el modelo, se utilizó la función `trainControl` con validación cruzada de 5 pliegues, probando valores de `alpha` entre 0.1 y 0.9 en incrementos de 0.2, y valores de `lambda` en escala logarítmica desde 0.0001 hasta 0.1. Esto permitió encontrar los mejores hiperparámetros con eficiencia computacional razonable. Se entrenaron dos modelos: el primero (`mod_1_EN_Gral`) utilizó la base `train_factor` y optimizó la métrica F1-Score y la curva ROC; el segundo (`mod_1_EN`) se entrenó con las 20 variables más relevantes (`train_importantes`) y empleó un rebalanceo de muestra para mejorar la clasificación de pobreza, con el F1-Score como métrica principal.

Las predicciones se generaron utilizando la función `predict` con el segundo modelo, ajustando el umbral para obtener los mejores resultados. Los resultados fueron almacenados en un archivo CSV para su análisis posterior. Las métricas de desempeño, como la matriz de confusión y el F1-Score, fueron evaluadas y comparadas entre los modelos, mostrando el impacto de la selección de variables y el rebalanceo en la precisión del modelo.

## 3.2 Modelo Logit

La regresión logística binaria (Logit) es una técnica estadística utilizada para modelar una variable dependiente binaria (0 o 1), estimando la probabilidad de que un hogar sea pobre (1) en función de varias variables explicativas. A diferencia de la regresión lineal, el modelo Logit usa una función logística (sigmoidea) que asegura que las predicciones se encuentren entre 0 y 1, permitiendo interpretarlas como probabilidades. Este modelo es útil para clasificar observaciones en dos categorías y comprender cómo las variables influyen en la probabilidad de pertenecer a una categoría.

En el ejercicio realizado, se entrenó un modelo Logit utilizando la función `glm` sobre la base `train_factors`, con la variable `Pobre` como objetivo. Las probabilidades estimadas (`prob_hat`) fueron clasificadas usando un umbral optimizado basado en la regla de Bayes, donde si la probabilidad era mayor que el umbral, se clasificaba como pobre. Se construyó una matriz de confusión y se calcularon métricas como precisión, exactitud, sensibilidad, especificidad, F1 Score y tasa de error para evaluar el desempeño del modelo.

Posteriormente, se entrenó un segundo modelo Logit utilizando validación cruzada de 5 pliegues con la función `train` del paquete `caret`, lo que permitió estimar el desempeño fuera de muestra.

Se repitieron las evaluaciones de desempeño y se analizó la importancia de las variables predictoras. Finalmente, se utilizaron las predicciones del modelo validado para clasificar los hogares en la base de prueba (`test_factors`), generando las probabilidades y variables de clasificación. Los resultados finales se almacenaron en un archivo CSV para su posterior uso.

## 3.3 Random forest

Random Forest Regression (RFR), una técnica basada en la creación de múltiples árboles de regresión que mejora la precisión al reducir el sobreajuste. Para entrenar el primer modelo, se utilizó unicamente Random Forest empleando `mtry = 4` y `ntree = 500`, valores identificados previamente como óptimos, sin validación cruzada. Este modelo permitió realizar predicciones y analizar la importancia de las variables, lo que ayudó a identificar los factores más relevantes para la clasificación de los hogares pobre o no pobres. Para evaluar su desempeño, se utiliza el error Out-Of-Bag (OOB), calculado con datos no seleccionados en el entrenamiento, así como valorar la importancia de Gini que mide la relevancia de cada variable, donde valores más altos indican mayor impacto (Zhao et al., 2019).

Posteriormente, se implementó un segundo modelo con validación cruzada utilizando la librería `ranger` y la función `train` de `caret`, con 5 particiones para evaluar métricas como `AUC`, `sensibilidad`, `especificidad` y `accuracy`. Esta validación cruzada mejoró la robustez del modelo y ayudó a evitar el sobreajuste, además de simplificarlo al eliminar variables no relevantes, como Dominio y el identificador del hogar (id).

El tercer modelo abordar el desbalance de clases en los datos, adoptando una estrategia de ponderación de pesos para la función de pérdida, asignandolos la clase minoritaria, mejorando los resultados de F1 Score de 0.61 con respecto al segundo modelo de 0.52. Por último se incorporó una transformación cuadrática de la variable `Nper` dado que se considera como una variable relevante. Este ajuste mantuvo la predicción en comparación al tercer modelo.

En cuanto al ajuste de los hiperparámetros, se realizaron pruebas iterativas para determinar los mejores valores, para `mtry` $= 4$ según la regla empírica de usar raíz cuadrática del número de variables, influyendo en la diversidad entre árboles. Se uso el coeficiente de Gini por su eficiencia y capacidad para manejar clases desbalanceadas (Louppe, 2014), y `min.node.siza` $= 1$ para permitir árboles más profundos que capturen patrones complejos, mitigando el sobreajuste gracias al promedio de múltiples árboles (Segal,2004), adicionalmente, el número de árboles definidos busca garantizar la estabilidad de las predicciones. La validación cruzada permitió evaluar el desempeño de los modelos y guiar la selección de los hiperparámetros, lo que resultó en la mejora de las métricas de desempeño como `AUC` y `accuracy`. La estrategia de ponderación de clases y la transformación de `Nper` fueron claves para mejorar la precisión y evitar el sesgo hacia las clases mayoritarias, logrando un modelo más equilibrado y efectivo para la predicción.

## 3.4 Regresión lineal

Para el entrenamiento del modelo bajo la metodología de regresión lineal, se implemento un modelo base como punto de partida por su facilidad de entrenamiento y bajo costo computacional. Por lo tanto, el modelo se adpató para el contexto de clasificación binaria mediante la transformación en clases. Se empleó la función `train` del paquete `caret`, aplicando validación cruzada de 5 pligues para garantizar una evaluación robusta y reducir el riesgo de sobreajuste.

Se desarrolló un segundo modelo utilizando únicamente las variables más importantes identificadas en el modelo base, lo cual redujo la complejidad del modelo sin pérdida significativa en el entrenamiento. Por último, se estimó un tercer modelo que incorporó interacciones entre las

mejores variables, obteniendo una mejor significativa para la predicción de los hogares pobres con respecto al primer modelo desarrollado al comparar las métricas de $R^2$, $RMSE$ y $MAE$.

## 3.5 Boosting

Boosting es una técnica de ensamblado que mejora el rendimiento de modelos débiles combinándolos de forma secuencial. Su objetivo es que cada nuevo modelo se entrene sobre los errores cometidos por los anteriores, reduciendo progresivamente el sesgo y aumentando la precisión del modelo final. A diferencia de otros métodos como el bagging, el boosting construye los modelos de manera aditiva, donde cada iteración corrige los errores de las predicciones previas.

El modelo XGBoost, una técnica de ensamblaje que combina modelos débiles secuencialmente. XGBoost se entrenó utilizando el paquete `xgboost` a través del paquete `caret` en R, con una grilla de hiperparámetros que incluyó, el número de rondas, la profundidad máxima del árbol, la tasa de aprendizaje, la regularización L1 (Lasso) y L2 (Ridge), y el peso mínimo de las hojas. Además, para enfrentar el desbalance de clases, se utilizaron pesos de clase. Este enfoque permitió que el modelo diera más énfasis a la correcta clasificación de la clase minoritaria, mejorando tanto la sensibilidad como la especificidad sin la necesidad de ajustar manualmente el umbral de decisión.

Los rangos de los hiperparámetros fueron seleccionados basados en la literatura y en pruebas preliminares para asegurar que el modelo tuviera la flexibilidad necesaria para adaptarse a los datos sin sobreajustarse. La estrategia de búsqueda empleada fue la *grid search*, la cual permitió evaluar exhaustivamente todas las combinaciones posibles de los hiperparámetros, se probaron dos valores para `nrounds` (250 y 500) y para `max_depth` (1 y 2) priorizando modelos simples que reduzcan el riesgo de sobreajuste. La tasa de aprendizaje *eta* se fijó en 0.1 y 0.01, valores recomendados para estabilizar el entrenamiento; además, se incluyó `gamma` en 0 y 1 para regular la ganancia mínima de división, y `min_child_weight` en 10 y 25 para controlar la complejidad del modelo. Los parámetros de muestreo `colsample_bytree` y `subsample`) se fijaron en 0.4 y 0.7 para introducir aleatoriedad y mejorar la generalización . Por último, la validación cruzada estratificada con AUC como métrica principal guió la selección de los hiperparámetros, asegurando que el modelo tuviera un rendimiento robusto y generalizable para `test`.

## 3.6 Naive Bayes

El modelo Naive Bayes es un clasificador probabilístico basado en el teorema de Bayes, que asume independencia condicional entre las variables predictoras dado el valor de la clase. A pesar de que esta suposición rara vez se cumple en la práctica, el algoritmo suele tener un buen rendimiento en tareas de clasificación, especialmente cuando se tienen muchas variables y los datos están razonablemente bien distribuidos.

El modelo Naive Bayes se implementó para clasificar los hogares como pobres o no pobres, utilizando el paquete `caret` con validación cruzada de 5 pliegues. Para manejar el desbalance de clases, se prestó especial atención a la sensibilidad y especificidad. Se realizó preprocesamiento a las variables mediante centrado y escalado, y se probó el uso de funciones de densidad `kernel` frente a la versión discreta del modelo. El mejor desempeño se logró con la versión `kernel`, con un AUC alto. Sin embargo, el modelo tendió a clasificar la mayoría de las observaciones en la clase mayoritaria, reflejando la necesidad de ajustes adicionales, como la modificación del umbral de clasificación o la incorporación de pesos de clase.

Para el ajuste del modelo se implementó una búsqueda en cuadrícula sobre tres hiperparámetros clave: `usekernel`, `adjust` y `laplace`. Se evaluaron configuraciones con y sin suavizado `kernel` (`usekernel` = TRUE/FALSE) para comparar la capacidad de modelar distribuciones

no paramétricas. El parámetro `adjust` se probó con valores de 0.5, 1 y 2, ajustando el grado de suavidad del `kernel` para balancear entre sobreajuste y generalización. Además, se incluyó el suavizado de `Laplace` entre 0 y 1 para mitigar problemas de probabilidad cero ante combinaciones poco frecuentes. Estas combinaciones se comprobaron mediante validación cruzada de 5 pliegues, utilizando el AUC como métrica de referencia.

En cuanto al preprocesamiento, se eliminaron variables con varianza casi nula que podían introducir ruido, y se estandarizaron los predictores mediante centrado y escalado para mejorar la estabilidad del modelo. También se aplicó sobremuestreo (`upsampling`) para corregir el desbalance de clases. Estas decisiones fueron guiadas por un análisis preliminar de la distribución de los datos, buscando optimizar el rendimiento sin comprometer la capacidad de generalización. Así, la combinación de técnicas permitió identificar la mejor configuración del modelo con criterios de precisión, robustez y eficiencia computacional.

## 3.7 CART

El algoritmo CART, que significa Árboles de Clasificación y Regresión, es una técnica propuesta por Breiman y sus colegas en 1984 para construir árboles de decisión. En el contexto de clasificación, su objetivo es predecir una categoría o clase a partir de un conjunto de variables explicativas. Para ello, divide de forma recursiva el espacio de los datos en regiones cada vez más homogéneas respecto a la variable de interés.

El modelo CART fue entrenado utilizando la función `rpart` con un valor por defecto para el hiperparámetro de complejidad (cp) de $\alpha = 0{,}01$ sugerido por Breiman, que permite realizar divisiones solo cuando contribuyen a una mejora significativa en la reducción del error. Adicionalmente, se usaron los valores predeterminados de `minsplit` = 20, que define el número mínimo de observaciones requeridas para intentar una división, y `maxdepth` = 30, que limita la profundidad máxima del árbol para evitar estructuras excesivamente profundas. Se incluyeron variables relevantes como el número de personas del hogar, el nivel educativo máximo, entre otras, y el árbol se podó automáticamente para evitar el sobreajuste.

Posteriormente, se generaron predicciones sobre la base de prueba, utilizando un umbral de 0.5 para clasificar a los hogares como pobres o no pobres. Para mejorar el modelo, se entrenó una versión ajustada mediante validación cruzada de 5 pliegues, utilizando la función `train` y optimizando el hiperparámetro de complejidad (cp). El modelo mostró un buen desempeño en términos de precisión, recall, F1 score y AUC, y se aplicó a la base de prueba para generar las predicciones finales.

## 3.8 Análisis comparativo de modelos

De acuerdo con las predicciones enviadas a Kaggle, se eligieron los mejores modelos predictivos del grupo en función del puntaje F1 Score para predicir si un hogar es pobre o no. El Cuadro 2. presenta los resultados obtenido para las predicciones empleando las metodologías seleccionadas para este documento. Durante el proceso de entrenamiento de los modelos, los principales factores que influyeron en la optimización de los modelos fue el balanceo de clases, las iteraciones para identificar los mejores hiperparámetros y el ajuste de los umbrales. Por último, se podrá consultar mayor información a nivel gráficas y resultados de tablas en la carpeta `04_Views` que se encuentra en el repositorio trabajo, así mismo, en la carpeta `02_Script` se encuentra un archivo en `RMarkdown` con el detalle de todas las estimaciones realizadas.

A partir de la comparación presentada anteriormente, es posible asegurar que la mejor predicción de pobreza se obtuvo con la metodología de Elastic Net (1), por lo tanto, se detalla a cotinuación

Cuadro 2: Comparación de desempeño para los mejores modelos predictivos

| # | Tipo de modelo | Score Kaggle | Métrica usada | Valor |
|---|---|---|---|---|
| 1 | Elastic Net | 0.6358 | Accuracy | 0.8352 |
| | | | AUC | 0.8797 |
| | | | F1 Score | 0.6233 |
| 2 | XGBoosting | 0.6253 | Accuracy | 0.7926 |
| | | | AUC | 0.8886 |
| | | | F1 Score | 0.6161 |
| 3 | XGBoosting | 0.6231 | Accuracy | 0.7958 |
| | | | AUC | 0.8885 |
| | | | F1 Score | 0.6159 |
| 4 | Random Forest (cv, wg, sq) | 0.6137 | Accuracy | 0.7959 |
| | | | AUC | 0.8822 |
| | | | Kappa | 0.4425 |
| 5 | Random Forest (cv, wg) | 0.6137 | Accuracy | 0.7959 |
| | | | AUC | 0.9147 |
| | | | ROC | 0.8857 |
| 6 | Elastic Net (umbral 20 %) | 0.6065 | Accuracy | 0.8489 |
| | | | AUC | 0.8675 |
| | | | F1 Score | 0.5045 |

aquellas características y/o factores que influyeron en la mejora de la predicción. Durante el proceso de entrenamiento se realizaron varias pruebas para identificar el poder preditivo de este método. Se realizaron gráficos y tablas que permitieron comprender el rendimiento del modelo como: importacia de las variables, comparación de métricas con el umbral de deicisión, curva ROC y matriz de confusión.

Como se mencionó en la sección 3.1 se desarrollaron dos entrenamientos para Elastic Net, la principal diferencia de los modelos es la cantidad de variables con las cuales fueron entrenados respectivamente así como el rebalanceo de las clases, resultando en la mejor predicción al entrenar el modelo con 20 variables de las 54 disponibles en la base de `train`. La Figura 1. muestra la importancia en escala de las variables seleccionadas para entrenar el segundo modelo con este método. Este proceso es fundamental para identificar los predictores más relevantes en la clasificación, con una importancia mayor a 50 se encuentra, `ncotizados`, `nt_trab_semanal`, `Nper` y `Nocupados`. Aunque las varaibles restantes en la Figura 1. tienen un importancia menor, la gráfica permitió conocer cómo el modelo prioriza variables socieconómicas y laborales, seguidas de características educativas y de tenencia de vivienda.

El modelo obtuvó el mejor desempeño en términos de la métrica F1 Score con un nivel de regularizarión y una valor de alpha de 0.2 y lamda de 0.0001 como se observa en la Figura 2. Este resultado siguiere que una combinación equilibrada entre pensalización L1 (Lasso) y L2 (Ridge) permite identificar y elimnar adecuadamente las variables irrelevantes.

También, es importante analizar las métricas con el umbral de deicisión del modelo y su comportamiento con distintos valores del umbral, de este modo es posible identificar el punto de equilibrio que optimiza el desempeño general del modelo como lo ilustra la Figura 3. Se observa que el F1 Score alcanza su nivel máximo cerca del umbra de 0.65, que coincide con el punto donde se equilibra presisión y recall. Este punto indica que el modelo logra su mayor balance entre la capacidad de detectar verdaderos positivos y evitar falsos positivos. Esta elección de

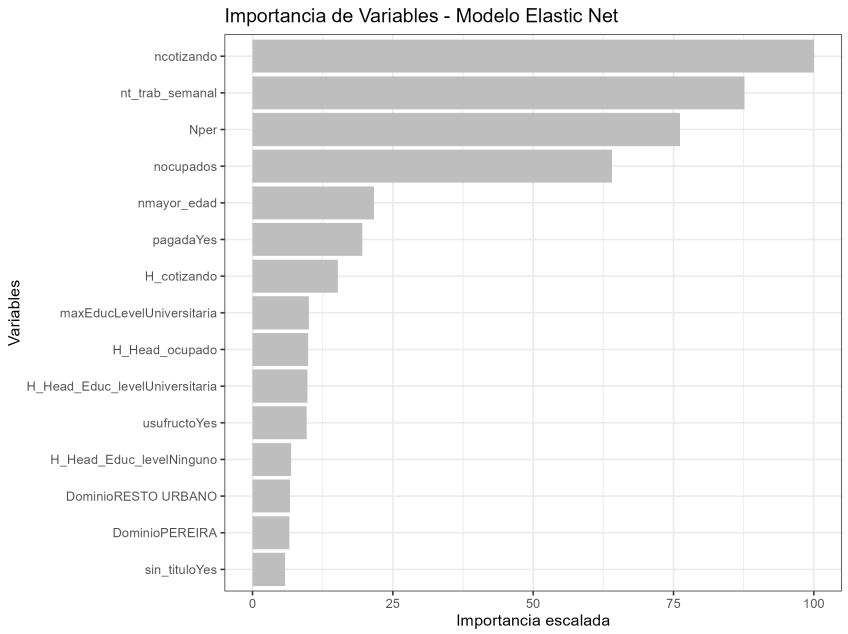


Figura 1: Importancia de variables

umbral es empíricamente sustentada, debido a la optimización del F1 Score, una métrica robusta para problemas con clases desbalanceadas como es el caso.

Del mismo modo, se argumenta la importancia de las variables en función de su comportamiento a lo largo del rango de umbrales, puesto que aquellas variables que aportan una mayor separación entre clases permiten al modelo mantener una alta presición con un recall competitivo, además, las variables más importantes permiten una mejor discriminación de clases, además de reducir el ruido sin sacrificar la generación de predicciones.

La matriz de confusión es una herramienta que permite estimar el desempeño del algoritmo aplicado como se muestra en el Cuadro 3. en donde (TN), el modelo predijo correctamente la clase negativa "predijó no pobre y es no pobre"; (FP), el modeo predijo incorrectamente la clase positiva "predijo pobre y no es pobre"; (FN), el modelo predijo incorrectamente la clase negativa "predijo no pobre y es pobre"; (TP), el modeo predijo correctamente la clase positiva "predijo pobre y es pobre".

Cuadro 3: Matriz de confusión del modelo predictivo

| **Activos negativos** | **Activos positivos** |
|---|---|
| Verdaderos negativos (TN) 116,169 | Falsos positivos (FP) 11,043 |
| Falsos negativos (FN) 15,767 | Verdaderos positivos (TP) 21,981 |

La curva ROC es una métrica relevante para medir la capacidad de rendimiento del modelo en función de verdaderos positivos (TP) y la tasa de falsos positivos (FP). En la Figura 4. se puede observar que el área debajo de la curva (AUC) alcanzó un valor de 0.8797, indicando un alto poder discriminatorio del modelo. Ahora bien, el punto resaltado en rojo indica el umbral óptimo de 0.6465, determinado con base en el equilibrio de sesibilidad y especificidad. Este umbral es coherente con el análisis de F1 Scrore, el cual alcanzó su máximo en torno a este valor.

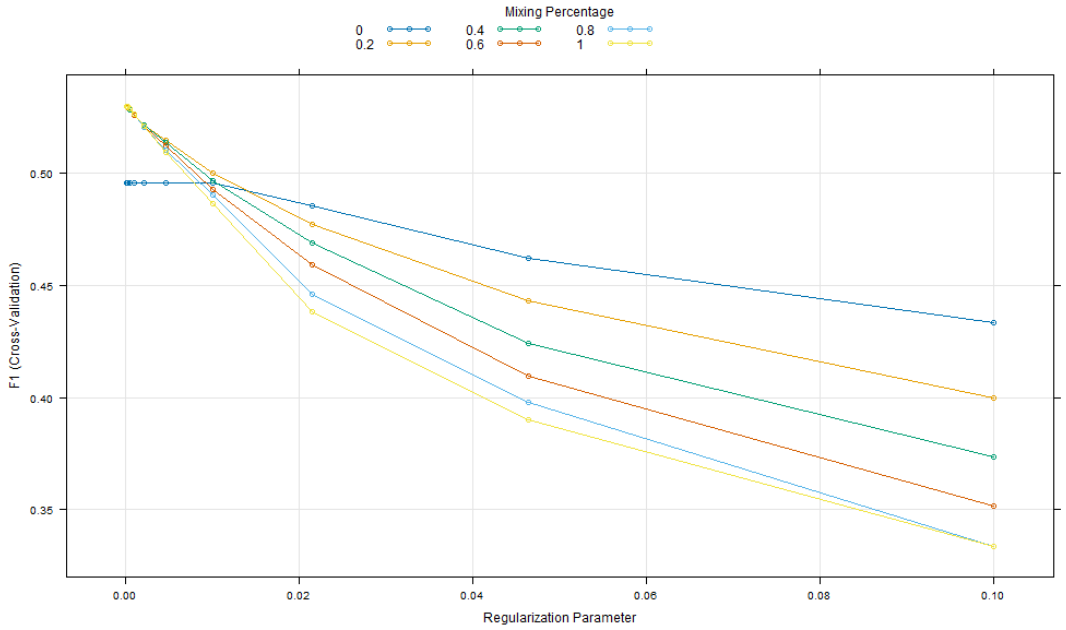


Figura 2: Resultados de Coss-Validation para Elasic Net

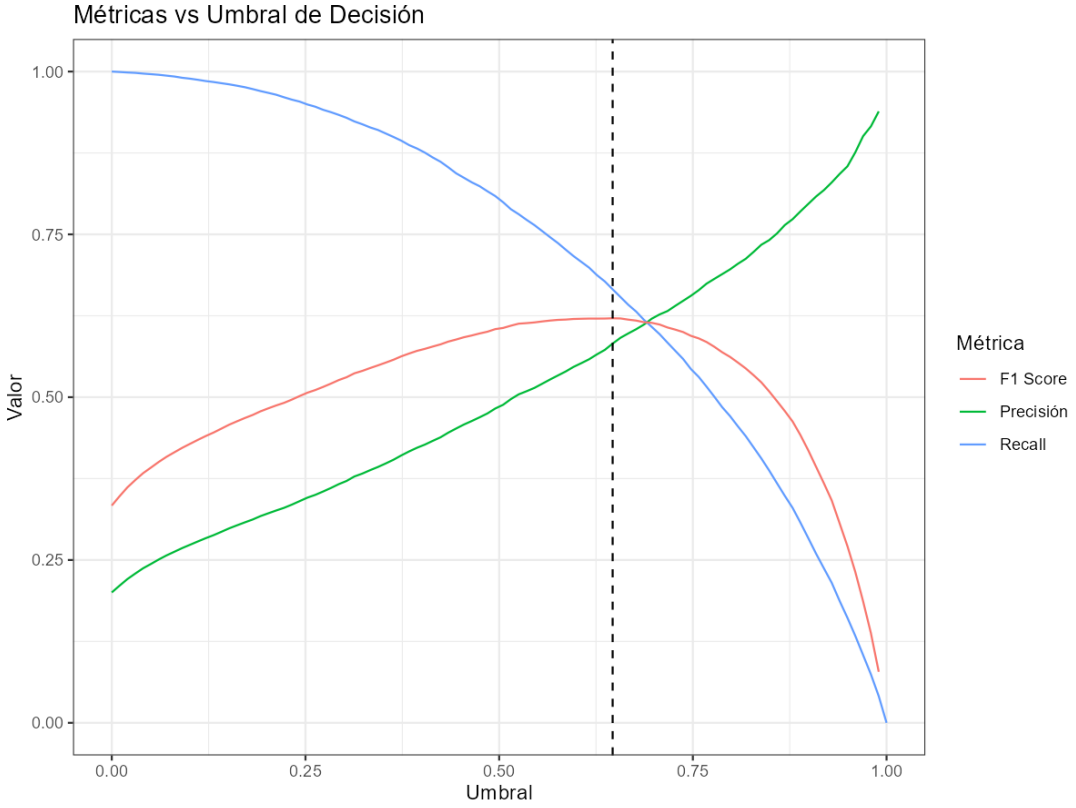


Figura 3: Punto de equilibrio para las métricas vs. umbral de desición

Con lo anterior, un AUC elevado implica que el modelo ha sido capaz de cosntruir una frontera de decisión eficaz, lo cual solo es posible si las varaibles seleccionadas continen suficiente información que aporten a la predicción. Estos resultados permiten validar que el modelo no solo ha identificado un conjunto relevante de variables predictoras, sino que ha sido ajustado con un umbral óptimo que máximiza su rendimiento práctico.

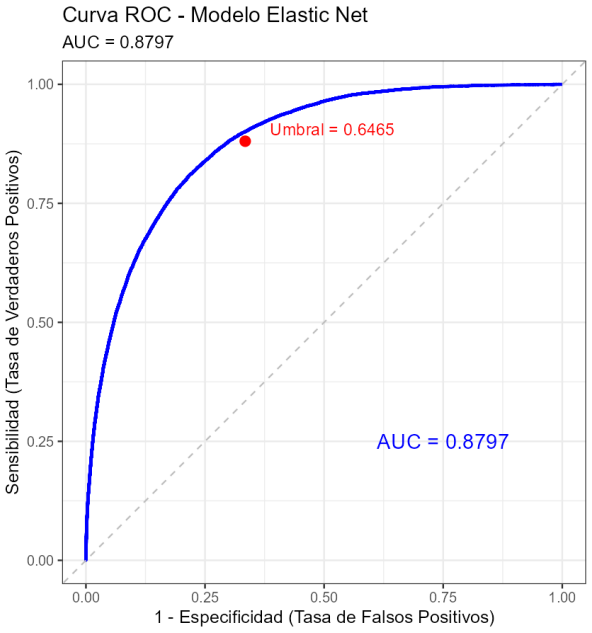


Figura 4: Curva ROC con Elastic Net

# 4. Conclusiones

Comprender el contexto de la información inicial resulta esencial, ya que permite dimensionar adecuadamente el alcance del análisis y proporciona una base sólida para interpretar los resultados obtenidos. En esta línea, el tratamiento de la muestra de entrenamiento se vuelve un paso clave para estructurar y ajustar los modelos bajo diferentes condiciones de entrnamiento.Por lo tanto, es relevante realizar una preselección de variables, conocer las estadísticas descriptivas de la muestra y analizar los modelos con diferentes variables relevantes para evitar el ruido y optimizar los resultados.

Durante el proceso de entrenamiento, se buscó optimizar los estimadores de cada modelo mediante el ajuste cuidadoso de hiperparámetros, la definición de umbrales de clasificación adecuados y el diseño de funciones personalizadas. Estas funciones permitieron balancear la precisión (falsos positivos) y el recall (falsos negativos) a través de la maximización del F1 Score, especialmente relevante en un contexto de datos desbalanceados. En este sentido, las funciones presentadas por R studio son herramientas fundamentales, pero es necesario implementar otras estrategias que lograr mejorar los resultados sin sobre ajustar.

Por último, los resultados del mejor modelo demuestran que las variables predictoras con mayor peso están asociadas a condiciones socioeconómicas y laborales de los hogares, actuando como aproximaciones del ingreso y la estabilidad económica factores relacionados con la condición de pobreza.

# 5. Referencias


1. Galvis Caballero, Á. (2021). ¿Cómo puede contribuir el Machine Learning a la focalización de programas sociales?. Modelo XGBoost para la determinación de pobreza monetaria interpretado mediante Shap Values: Caso Colombia 2019-2020 (Doctoral dissertation, Universidad Autónoma de Bucaramanga UNAB).

2. Brown, C., Ravallion, M., & van de Walle, D. (2018). A poor means test? Econometric targeting in Africa. Journal of Development Economics, 134, 109-124. https://doi.org/10.1016/j.jdeveco.2018.05.004

3. SOHNESEN, T. P.; STENDER, N. Is Random Forest a Superior Methodology for Predicting Poverty? An Empirical Assessment. Poverty & Public Policy, [s. l.], v. 9, n. 1, p. 118–133, 2017. Disponible en: https://research-ebsco-com.ezproxy.uniandes.edu.co/linkprocessor/plink?id=48765633-5416-3407-9047-3a2966a3243e. Acesso en: 28 mar. 2025.

4. Okiabera, J. O. (2020). Using random forest (RF) to identify key determinants of poverty in Kenya (Doctoral dissertation, University of Nairobi).

5. Zhao, X., Yu, B., Liu, Y., Chen, Z., Li, Q., Wang, C., & Wu, J. (2019). Estimation of Poverty Using Random Forest Regression with Multi-Source Data: A Case Study in Bangladesh. Remote Sensing, 11(4) https://doi.org/10.3390/rs11040375

6. Breiman, L. (2001) Random Forests. Machine Learning, 45, 5-32.

7. Louppe, Gilles. (2014). Understanding Random Forests: From Theory to Practice.

8. Segal, M.R. (2004) Machine Learning Benchmarks and Random Forest Regression. Center for Bioinformatics and Molecular Biostatistics, University of California, San Francisco.

9. Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (n.d.). How many trees in a random forest? Department of Computer Science and Mathematics, Faculty of Philosophy, Sciences and Languages at Ribeirao Preto, University of Sao Paulo. Retrieved from http://www.ic.unicamp.br/~wainer/cursos/MO444/papers/osh10a.pdf

10. Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees (illustrated, reprint ed.). Taylor & Francis

11. Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. Computer Science Department, Stanford University.