UNIVERSIDAD NACIONAL AGRARIA

LA MOLINA

FACULTAD DE ECONOMÍA Y PLANIFICACIÓN

"IDENTIFICACIÓN DE CLIENTES EN CAMPAÑAS PARA UNA ENTIDAD FINANCIERA USANDO EL MÉTODO STACKING"

TESIS PARA OPTAR TÍTULO DE

INGENIERO ESTADÍSTICO E INFORMÁTICO

JULIO CÉSAR ALVAREZ CHANCASANAMPA

LIMA – PERÚ

2022

Ouriginal by Turnitin

Document Information

Analyzed document	Tesis_final_jalvarez_UNALM.pdf (D142538650)
Submitted	2022-07-30 18:22:00
Submitted by	HERMELINDA ALVAREZ CHANCASANAMPA
Submitter email	halvarez@lamolina.edu.pe
Similarity	10%
Analysis address	halvarez.unalm@analysis.urkund.com

Sources included in the report

W	URL: https://comunidad.iebschool.com/bigdata/2015/05/13/la-mineria-de-datos-entre-la-estadistica-y-la-inteligencia-artificial/ Fetched: 2022-07-30 18:23:00	5
W	URL: https://empresas.blogthinkbig.com/que-algoritmo-elegir-en-ml-aprendizaje/ Fetched: 2022-07-30 18:23:00	3
W	URL: https://jllopisperez.com/2013/12/19/test-de-hosmer-y-lemeshow/ Fetched: 2022-07-30 18:23:00	1
W	URL: https://analisisydecision.es/medir-la-importancia-de-las-variables-con-random-forest/ Fetched: 2022-07-30 18:23:00	3
SA	**M2.878_20212_PEC4.1 - Redacci&#65533;n de la memoria (1a Entrega)_17709253.txt** Document M2.878_20212_PEC4.1 - Redacci&#65533;n de la memoria (1a Entrega)_17709253.txt (D138333674)	1
SA	**M0.156_20212_PAC 2_17838978.txt** Document M0.156_20212_PAC 2_17838978.txt (D140740664)	2
W	URL: https://www.cise.ufl.edu/~adobra/papers/a-exam.pdf Fetched: 2022-07-30 18:23:00	1
W	URL: https://www.sciencedirect.com/science/article/abs/pii/S0378779620307021. Fetched: 2022-07-30 18:23:00	1
W	URL: https://www.scirp.org/html/9-1240025_8072.htm Fetched: 2022-07-30 18:23:00	1
W	URL: https://www.knime.com/blog/from-modeling-to-scoring-confusion-matrix-and-class-statistics Fetched: 2022-07-30 18:23:00	1
SA	**1568256739_365__Proyecto_de_Deteccion_de_retrasos_de_pagos._Arriola,_Castelo,_Paredes,_Vinueza_F.pdf** Document 1568256739_365__Proyecto_de_Deteccion_de_retrasos_de_pagos._Arriola,_Castelo,_Paredes,_Vinueza_F.pdf (D55581247)	1
SA	**M0.156_20212_PAC 2_17842835.txt** Document M0.156_20212_PAC 2_17842835.txt (D140768647)	1

UNIVERSIDAD NACIONAL AGRARIA LA MOLINA

FACULTAD DE ECONOMÍA Y PLANIFICACIÓN

"Identificación de clientes en campañas para una entidad financiera usando el método Stacking"

TESIS PARA OPTAR TÍTULO DE
INGENIERO ESTADÍSTICO E INFORMÁTICO

Presentado por:
JULIO CÉSAR ALVAREZ CHANCASANAMPA

M.A. Fernando Kené Rosas Villena.
PRESIDENTE

MS. Grimaldo José Febres Huamán.
MIEMBRO

Mg. Sc Iván Dennys Soto Rodríguez
MIEMBRO

Mg. Sc. Clodomiro Fernando Miranda Villagómez.
PATROCINADOR

DEDICATORIA

A mis padres, hermanos y sobrinos, quienes me apoyaron moralmente para cumplir con el objetivo.

A seguir remando, para ir por más.

AGRADECIMIENTO

A Dios por haber iluminado en todo mi desarrollo. También a las personas que estuvieron a mi lado en esta travesía larga amig (os), Doctores e Ingenieros.

Un agradecimiento especial, a mi madre, por su apoyo moral para seguir continuando y lograr el objetivo.

Un agradecimiento especial, a mi hermana la Dra. Hermelinda, por su apoyo en esta travesía del proceso de la tesis.

Un agradecimiento especial, a mi hermano Ing. Jurgen, por su apoyo y criticas para lograr el objetivo.

Un agradecimiento especial, a mis demás Herman(os), por su apoyo en esta travesía del proceso de la tesis que fue duro.

Un agradecimiento especial, a la Asistenta Social de economía Elizabeth Gereda Martínez por su apoyo moral para seguir en mi lucha y el apoyo que me brindo en mi época universitaria.

Un agradecimiento especial, al profesor Dr. Fernando Rosas Villena por el apoyo del ordenamiento para la culminación de mi tesis. Fue una gran ayuda, estaré eternamente agradecido

Un agradecimiento especial, al profesor Mg. Sc. Fernando Miranda Villagómez por ser mi asesor de tesis y apoyar en esta larga travesía y lucha constante.

Un agradecimiento especial, a la secretaria de la facultad de economía y planificación Merajad Dúmet Montoya por decirme las cosas claras en esta travesía.

Un agradecimiento especial, al profesor Dr. Felipe De Mendiburu Delgado por darme sugerencia al inicio y en el proceso de cómo conseguir el objetivo, ya que la lucha es constante.

Un agradecimiento especial, a mis amig(os) Ingenieros Jhonatan Catpo, José Arizaca, Cinthya Arrollo, Ruby Ordoñez y César Guevara. Por su apoyo moral, en el largo tiempo que les conocí por sus consejos y amistad que perdura.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

N° TABLAS **TÍTULO** **Pág.**

ÍNDICE DE FIGURAS

N° FIGURA **TÍTULO** **Pág.**

ÍNDICE DE ANEXOS

N° ANEXO **TÍTULO** **Pág.**

RESUMEN

La presente investigación, tiene como objetivo general determinar si el método de ensamble Stacking predice con mayor precisión a los clientes potenciales a quienes se les otorgará o desembolsará préstamos en las ofertas de campaña de una entidad financiera, que los algoritmos de aprendizaje supervisado de Machine Learning: Random Forest, Regresión Logística y Árbol de Decisión. La evaluación se realizó comparando el método Stacking con los modelos individuales de Regresión Logística, Árbol de Decisión y Random Forest. Para dicha evaluación se usaron los indicadores Auc, Gini, Logloss y Kolmogorov. Los resultados de sensibilidad en orden de importancia que se obtuvieron con los modelos estadísticos fueron lo siguiente: Regresión Logística 88.9%, seguido del método Stacking con 87.9%, luego el Árbol de Decisión con un 84% y por último Random Forest con un 82.7%. Mientras, que al evaluar la especificidad el de mayor importancia fue el modelo de Random Forest con un 84.8%, Árbol de Decisión 82.8%, método Stacking 81.6% y por último Regresión Logística 78.4%. Respecto a los indicadores evaluados, el que presentó mayor Auc es el método Stacking 0.9117, seguido de Random Forest con un 0.9074, la Regresión Logística reportó un 0.9064 y Árbol de Decisión 0.9074. Con respecto al indicador Gini, el que tiene mayor Gini es el método de Stacking con 0.8235, seguido de Random Forest con un 0.8148, la Regresión Logística cuyo resultado fue de 0.8128 y en última posición el Árbol de Decisión con 0.7885. Con relación al indicador Logloss, el que mostró mejor desempeño fue el método Stacking con 0.3177, seguido de Random Forest con 0.3435, Árbol de Decisión 0.3886 y Regresión Logística 0.3959. Finalmente, con respecto al indicador Kolmogorov, el que tiene mejor resultado es el método Stacking con 0.7124, seguido por Random Forest con 0.7028, Árbol de Decisión 0.6907 y por último la Regresión Logística con 0.6751.

Palabras clave: Método Stacking, Auc, Gini y Logloss.

ABSTRACT

The general objective of this research is to determine if the Stacking assembly method predicts with greater precision the potential clients who will be granted or disbursed loans in the campaign offers of a financial institution, than the Machine Learning supervised learning algorithms. : Random Forest, Logistic Regression and Decision Tree. The evaluation was carried out by comparing the Stacking method with the individual models of Logistic Regression, Decision Trees and Random Forest. For this evaluation were used the indicators Auc, Gini, Logloss and Kolmogorov. The results obtained with the confusion matrix were: the Logistic Regression presents greater sensitivity with 88.9%, followed by the Stacking method with 87.9%, then the Decision Tree with 84% and finally Random Forest was 82.7%. While, in the specificity the highest is Random Forest 84.8%, Decision Tree 82.8%, the stacking method 81.6% and finally Logistic Regression 78.4%. Respect to the evaluated indicators, the one with the greatest Auc is the method Stacking 0.9117, followed Random Forest with a 0.9074, Logistic regression reported a 0.9064 y Decision tree with a 0.9074. Also, the one with the greatest Gini is the method of Stacking with 0.8235, followed Random Forest with a 0.8148, the Logistic regression whose result was 0.8128 and Decision tree result 0.7885. Regarding, the indicator Logloss showed better performance is the stacking method 0.3177, then Random Forest 0.3435, Decision tree 0.3886 y Logistic regression 0.3959. Finally, the Kolmogorov indicator, the one with the best indicator is the stacking method 0.7124, followed by Random Forest 0.7028, Decision tree 0.6907 and finally Logistic regression 0.6751.

Keywords: Method Stacking, Auc, Gini y Logloss.

I. INTRODUCCIÓN

Las entidades financieras brindan diversos servicios y productos a los clientes, tales como: transacciones, cuentas sueldo, cuentas de ahorro, créditos hipotecarios, tarjetas de crédito, tarjetas de débito etc., donde la prioridad de toda la gama de actividades que realizan es no permitir que sus carteras envejezcan y se queden sin clientes, por lo que es importante la captación de nuevos clientes. Las entidades financieras invierten grandes volúmenes de sumas de dinero para ampliar su base de clientes, pero a la hora de ver los resultados, el costo por contacto resulta muy alto y la efectividad baja. Una de las formas de captación de nuevos clientes es la realización de llamadas, llamadas que a veces son innecesarias en clientes que no desean ofertas de campaña.

Ferraro (2013) señala que en la actualidad una de las formas muy utilizada por los bancos para captar clientes es la realización de constantes llamadas realizadas a partir de una base de datos.

Las entidades financieras al comunicarse con personas que no tienen interés con los servicios que se les ofrece, gastan tiempo, dinero y recursos. Si la oferta que se les ofrece resulta negativa, se puede calificar como un proceso ineficaz por parte de las entidades bancarias, obteniendo pocos resultados en comparación con las inversiones que realizan. Sin embargo, si consiguen que un cliente acepte un préstamo por esta vía, significa una gran conquista para los bancos, porque asegura la fidelidad del cliente, por lo menos hasta vencerse el plazo del mismo, como así también un gran esfuerzo sobre todo económico.

En las ofertas de campañas se busca la compra de un determinado producto en un segmento determinado de la cartera de clientes al que se les puedes ofertar servicios diferenciados por su valor comercial. También en las ofertas de campañas se busca realizar préstamos económicos y así obtener mayores utilidades en el negocio.

Si la entidad financiera no realiza una buena campaña se originan problemas, reflejándose esto en la disminución de la cartera de clientes al no captar nuevos clientes, produciéndose entonces la disminución de los ingresos en la entidad financiera.
Algunas campañas resultan un fracaso debido a que estas no fueron implementadas con conocimiento anterior el comportamiento de compra de los clientes.

Por esta razón es primordial saber con antelación, a cuántos clientes la entidad financiera desembolsará u otorgará un préstamo en un periodo determinado.

En la actualidad la Minería de Datos, ofrece potentes algoritmos de predicción. Las iteraciones de aprendizaje automático que se producen entre los distintos modelos que surgen hacen que estos tipos de modelos sean robustos, éstos son los denominados Métodos de Ensamble, donde el objetivo final es obtener una eficiencia óptima en la predicción.

Campo y Cruz (2017) indican que las compañías están incursionando en la utilización de modelos predictivos para conocer con anterioridad el comportamiento de sus clientes y un mejor rendimiento en las predicciones se obtiene usando interacciones entre modelos estadísticos. Uno de ellos, es el método Stacking introducido por Wolpert en 1992.

Beltrán et al. (2012) en su investigación para predecir a qué clientes otorgar un crédito bancario, concluyen que el método Stacking es superior a la Regresión Logística y Random Forest, obteniendo un 80.8%, 78.7% y 80.2% respectivamente en la clasificación general o accuracy. Con respecto al área bajo la curva (AUC) con el método de Stacking también se obtuvo un resultado superior a los modelos mencionados, siendo éstos: 0.893 para el método de Stacking, 0.867 para la Regresión Logística y 0.865 para Random Forest.

Padmapani et al. (2018) en el estudio que se realizó para clasificar una opinión como positiva o negativa con respecto al rendimiento de laptops, el método Stacking dió resultados altos con un accuracy de 92.53%, seguido por máquina de soporte vectorial (SVM) 72.25%, Naive Bayes (Nb) 60.1% y el algoritmo de Vecinos más cercanos (Knn) 9.6% de exactitud.

Padmapani et al. (2018) demuestran que el rendimiento del método Stacking es mejorado al combinar clasificadores heterogéneos.

Por lo tanto, el presente trabajo de investigación tiene como objetivo general determinar si el método de ensamble Stacking predice con mayor precisión a los clientes potenciales a quienes se les otorgará o desembolsará préstamos en las ofertas de campaña de una entidad financiera, que los algoritmos de aprendizaje supervisado de Machine Learning: Random Forest, Regresión Logística y Árbol de Decisión.

- La identificación de clientes potenciales para campañas de una entidad financiera usando el método Stacking con los algoritmos de Regresión logística, Árbol de Decisión y Random Forest.

- Comparación del método Stacking propuesto con los algoritmos de Regresión logística, Árbol de Decisión y Random Forest mediante los indicadores Auc, Gini, Logloss, Kolmogorov.

II. REVISIÓN DE LITERATURA

2.1. Antecedentes

Kotsiantis et al. (2007) realizaron el estudio de detección de fraude de clientes en una entidad financiera aplicando algoritmos individuales y de ensamble para las predicciones. El método ensamble Stacking tuvo el mayor accuracy 95.1, seguido del algoritmo C4.5 con 91.2, Regresión Logística 75.3 y Random Forest 73.4. Con respecto al recall: Stacking presentó 90.2, C4.5 85.2, Random Forest 86.3 y regresión Lineal 36.6.

Portugal y Carrasco (2006) explican que el método Stacking combina múltiples modelos que han sido entrenados para una tarea de clasificación, es decir, combina varios clasificadores para inducir un clasificador de nivel más alto con un mejor rendimiento.
Realizaron un estudio de detección de fraude financiero en donde el mayor porcentaje de clientes correctamente clasificados fue con el método Stacking (81.2%), seguido de Naive Bayes (72.2%) y luego el Árbol de Decisión con 79.8% de clasificación correcta.

Beltrán et al. (2012) en la investigación de predicción de clientes para el otorgamiento de un crédito bancario, concluyen que Stacking es superior a la Regresión Logística y Random Forest en la clasificación general o accuracy, logrando un 80.8% respecto a la Regresión Logística que obtuvo un 78.7% y Random Forest 80.2%. Con respecto al área bajo la curva (AUC) con el método de Stacking también se obtuvo un resultado superior a los modelos mencionados, siendo esto 0.893, 0.867 para la Regresión Logística y 0.865 para Random Forest.

Padmapani et al. (2018) también mencionan que el método Stacking combina clasificadores heterogéneos y ofrece un rendimiento mejorado.

En el estudio para clasificar una opinión como positiva o negativa con respecto al rendimiento de laptops, el método Stacking dio resultados altos con un accuracy de 92.53%, seguido por máquina de soporte vectorial (SVM) 72.25%, Naive Bayes (Nb) 60.1% y el algoritmo de Vecinos más cercanos (Knn) e exactitud.

2.2. Bases Teóricas

2.2.1. Machine Learning

Rome (1999) menciona que los modelos predictivos se utilizan con frecuencia en estadística junto con algoritmos de machine learning y se basan en el análisis de la información de datos actuales e históricos para realizar las predicciones sobre eventos futuros. Además, muestran las relaciones entre las diversas variables permitiendo capturar información potencial a un conjunto de condiciones, guiando así en la toma de decisiones. Es así por ejemplo que en el mundo de negocios se explotan los patrones de comportamiento encontrados en los clientes para poder identificar riesgos y oportunidades.

Los modelos predictivos también permiten conocer que productos son más propensos a ser adquiridos por el cliente, basándose para ello en el comportamiento histórico de los mismos. Los productos más propensos pueden ser: una línea de crédito, monto de consumo en establecimientos asociados a la corporación, score de bancos, etc.

Las raíces de machine learning son la inteligencia artificial y la estadística. La estadística se encarga del estudio de poblaciones, como la variabilidad que permite la modelización de los fenómenos, a la vez como de métodos para la síntesis de la información contenida en los datos.

Podríamos distinguir dos tipos de estadística: la exploratoria (Data Analysis) y la inferencial (Data Modelling) no es fácil establecer el límite entre ambas, se considera que la estadística exploratoria es la antesala de la estadística inferencial.

A la vez se puede informar que la inteligencia artificial, se preocupa de establecer soluciones algorítmicas a costes razonables, y la estadística se ocupa de generalizar los resultados obtenidos, trasladando los datos a situaciones más generales que la estudiada (López, 2015). Según Heller (2019) el aprendizaje automático es una rama de la inteligencia artificial que incluye métodos o algoritmos para crear automáticamente modelos a partir de los datos. Los algoritmos de aprendizaje automático aprenden de los datos para dar una solución a problemas que son demasiado complejos para resolverlos con la programación convencional. A diferencia de un sistema que realiza una tarea siguiendo reglas explícitas, un sistema de aprendizaje automático aprende de la experiencia. Mientras que un sistema basado en reglas realizará una tarea de la misma manera cada vez (para bien o para mal) el rendimiento de un sistema de aprendizaje automático se puede mejorar mediante el entrenamiento, al exponer el algoritmo a más datos.

A la vez Heller (2019) nos informa que los algoritmos de aprendizaje automático a menudo se dividen en supervisados (los datos de entrenamiento están etiquetados con las respuestas) y no supervisados (cualquier etiqueta que pueda existir no se muestra en el algoritmo de entrenamiento).

a. Algoritmos de aprendizajes supervisados

En el aprendizaje supervisado, los algoritmos trabajan con datos "etiquetados", intentado encontrar una función que, dadas las variables de entrada (input data) les asigne la etiqueta de salida adecuada. El algoritmo se entrena con un "histórico" de datos y así "aprende" a asignar la etiqueta de salida adecuada a un nuevo valor, es decir, predice el valor de salida (Simeone, 2018).

Según Beunza et al. (2019) se refiere a un tipo de modelos de machine learning que se logran entrenar con un conjunto donde los resultados de salida son conocidos. Los modelos logran aprender de esos resultados conocidos y realizan ajuste para adaptarse a los datos de entrada.

Según Mariñas (2009) el aprendizaje supervisado cuando la máquina recibe una lista de salidas deseadas y_1, y_2, y_3, …, y el objetivo de la máquina es aprender a producir la salida correcta dada una nueva entrada. Es decir, va aprendiendo cómo son los miembros que

componen las distintas clases hasta que sea capaz de formar un ‘modelo’ o ‘prototipo’ de los miembros de cada clase.

Según Alvarado (2003) los algoritmos supervisados estiman una función f que mejor asocia a un conjunto de datos X (variables independientes) con un conjunto de datos Y (variables dependientes).

b. Algoritmos de aprendizajes no supervisados

Según Simeone (2018) el aprendizaje no supervisado tiene lugar cuando no se dispone de datos “etiquetados” para el entrenamiento. Sólo se conoce los datos de entrada, pero no existen datos de salida que correspondan a un determinado input. Por tanto, sólo podemos describir la estructura de los datos, para intentar encontrar algún tipo de organización que simplifique el análisis. Por ello, tienen un carácter exploratorio.

Según Alvarado (2003) nos informa que dado un conjunto de variables aleatorias x_1, x_2, x_3,.., para los cuales no existe ninguna variable Y que clasifique a estas variables.

Tabla 1: Tipos de aprendizaje de Machine Learning

SUPERVISADOS	NO SUPERVISADOS
Árbol de Decisión Regresión Logística Random Forest SVM Redes Neuronales	Segmentación Agrupamiento"Clustering" Reglas de asociación

Fuente: Elaboración propia.

2.2.2. Regresión Logística

Abraira y Pérez (1996) los modelos de Regresión Logística son modelos que permiten estudiar si una variable binomial depende, o no, de otra u otras variables (no necesariamente binomiales). Si una variable binomial de parámetro *p* es independiente de otra variable *X*, se cumple $p = p|X$, por consiguiente, un modelo de regresión es una función de *p* en *X* que a través del coeficiente de *X* permite investigar la relación anterior

Abraira y Pérez (1996) indica que el proceso es binomial cuando sólo tiene dos posibles resultados: "éxito" y "fracaso", siendo la probabilidad de cada uno de ellos constante en una serie de repeticiones. A la variable número de éxitos en *n* repeticiones se le denomina variable binomial. A la variable definida de un sólo ensayo *y*, con sólo dos valores: 0 para fracaso y 1 para éxito, se le denomina binomial puntual.

Un proceso binomial está caracterizado por la probabilidad de éxito, representada por p (es el único parámetro de su función de probabilidad) la probabilidad de fracaso se representa por q y, evidentemente, ambas probabilidades están relacionadas por $p + q = 1$. En ocasiones, se usa el cociente p/q , denominado "odds", y que indica cuánto más probable es el éxito que el fracaso, como parámetro característico de la distribución binomial, aunque, evidentemente, ambas representaciones son totalmente equivalentes (Abraira y Perez (1996)).

Fiuza y Rodríguez (2000) es uno de los instrumentos estadísticos más expresivos y versátiles de que se dispone para el análisis de datos. Su origen se remonta a la década de los sesenta (Confield, Gordon y Smith 1961); su uso se universaliza y expande desde principios de los ochenta debido, especialmente, a las facilidades informáticas con que se cuenta desde entonces.

La Regresión Logística se utiliza cuando queremos investigar si una o varias variables explican una variable dependiente que toma un carácter cualitativo.

c. El modelo logístico

Según Salas (1996) la Regresión Logística de respuesta binaria tiene una variable dependiente Y=0 o Y=1, y p número de variables independientes (X_j) que pueden ser continuas o discretas. El tipo de variable Y que admite sólo dos valores se denomina Dicotómica y tendrá una distribución de Bernoulli, donde:

$$P(Y = 1) = \pi \;\; y \;\; P(Y = 0) = 1 - \pi$$

Por lo que:

$$E[Y] = 0xP(Y = 0) + 1xP(y = 1)$$

$$E[Y] = P(Y = 1)$$

Entonces, el valor esperado es la probabilidad de que *Y*=1 es:

$$E\left[\frac{y}{x}\right] = P(Y = 1) = \beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P$$

Por lo tanto, se tiene:

$$E\left[\frac{y}{x}\right] = P(Y = 1) = \pi = X\beta = \beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P \qquad (2.1)$$

Para la Regresión Logística binaria Logit se realiza una transformación de forma lineal y se obtienen resultados que cumplan con el rango de la probabilidad:

$$\pi(x) = \frac{e^{\beta_0 + \sum \beta_j X_j}}{1 + e^{\beta_0 + \sum \beta_j X_j}} \qquad (2.2)$$

Fiuza y Rodríguez (2000) expresan que el objetivo de la Regresión Logística es expresar la probabilidad de que ocurra un hecho en función de ciertas variables.

El modelo de Regresión Logística puede escribirse como:

$\pi(x) = p$

$$\log\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_i \qquad (2.3)$$

donde *p* es la probabilidad de que ocurra el evento de interés (en nuestro caso tener éxito). Dado el valor de las variables independientes, podemos calcular directamente la estimación de la probabilidad de que ocurra el evento de interés de la siguiente forma:

$$\hat{p} = \frac{e^{suma}}{1+e^{suma}} \qquad (2.4)$$

donde: $suma = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_i$

La función $f(p) = \log\left(\frac{P}{1-P}\right)$ es llamada *logit*.

d. Estimación Máximo Verosímil de los Parámetros

Sea *i = 1, 2, ..., n,* una muestra de *n* observaciones independientes (x_i, y_i) donde y_i es denotado como un valor de la variable respuesta dicotómica y x_i el valor de la variable independiente para la i-ésima observación. La variable de respuesta está codificada como 0 o 1, representando la ausencia o presencia respectivamente de la variable respuesta. Entonces dado el modelo de Regresión Logística en la ecuación (2.2) se requiere que se estime los parámetros desconocidos (β_0, β_1).

Según Nieto (2015) la función de verosimilitud expresa la probabilidad de los datos observados como una función de los parámetros desconocidos y está expresado por la siguiente ecuación:

$$l(\beta) = \prod_{i=1}^{n} \pi(x_i)^{y_i}(1 - \pi(x_i)_i^{y})^{1-y_i} \qquad (2.5)$$

En donde, $\pi(x_i)^{y_i}(1 - \pi(x_i))^{1-y_i}$,es la contribución a la función de verosimilitud para el par de observación (x_i, y_i).

La expresión de la ecuación (2.5) es una productoria dado que las observaciones se asumen como independientes.

La estimación Máximo Verosímil de los Parámetros (β) significa que para la estimación de β, debemos usar el principio de máxima verosimilitud es decir el valor que maximiza la expresión en la ecuación (2.5). Sin embargo, es mucho más fácil matemáticamente trabajar con el logaritmo de dicha ecuación:

$$L(\beta) = \ln\big(l(\beta)\big) = \sum_{i=1}^{n}\{y_i \ln\big(\pi(x_i)\big) + (1 - y_i)\ln\big(1 - P_i(x_i)\big)\} \qquad (2.6)$$

Esta expresión es denomina log-verosimilitud

Entonces para encontrar el valor de β que maximiza L(β) diferenciamos L(β) con respecto a β_0 y β_1, resultando entonces dos expresiones (ecuaciones de verosimilitud) que se iguala a cero:

$$\sum_{i=1}^{n}[y_i - \pi(x_i)] = 0 \qquad (2.7)$$

y

$$\sum_{i=1}^{n} x_i[y_i - \pi(x_i)] = 0 \qquad (2.8)$$

Por lo que el valor de β dado para las expresiones (2.7) y (2.8) son los llamados estimadores de máxima verosimilitud y se denota como $\hat{\beta}$.

Para ver un estimador de máxima verosimilitud ($\hat{\beta}$) para la ecuación (2.7) es:

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} \hat{\pi}(x_i) \qquad (2.9)$$

Indicando que la suma de los valores observados de *Y* es igual a la suma de los valores predichos esperados.

"Por lo tanto, los estimadores resultantes para la Regresión Logística binaria son los que están estrechamente relacionados con los datos observados. Si *Y* está codificada como 0 o 1 entonces la expresión para $\pi(x_i)$ dada en la ecuación (2.2) nos provee (para valores arbitrarios de $\beta = (\beta_0, \beta_1)$ el vector de parámetros) la probabilidad condicional de que *Y* es

igual a 1 dado *x* ósea P (*Y* = 1|X = *x*) y la probabilidad condicional que *Y* es igual a cero (1 − $\pi(x)$) dado x será P (*Y* = 0|X = *x*).

Por lo tanto, para los pares (x_i, y_i) , donde $y_i = 1$, la contribución a la función de verosimilitud es $\pi(x_i)$, y para el par donde $y_i = 0$, la contribución a la función de verosimilitud es $1 - \pi(x_i)$."

e. Pruebas del modelo de regresión logística

- **Prueba de Razón de Verosimilitud**

  Silva y Molina (2016) definen a la razón de verosimilitud o likelihood ratio (LR) como la razón entre la posibilidad de observar un resultado en pacientes con cierta enfermedad versus la posibilidad de ese resultado en pacientes sin la enfermedad. Su uso constituye una herramienta de gran utilidad para la toma de decisiones clínicas frente a la solicitud de algún *test* diagnóstico y su cálculo se deriva de probabilidades condicionadas en base al teorema de Bayes, pero además ésta se puede estimar en base a los parámetros de sensibilidad y especificidad de la siguiente manera:

$$LR(+) = \frac{\text{Sensibilidad}}{1 - \text{Especificidad}} = \frac{\text{Tasa de verdaderos positivos}}{\text{Tasa de falsos positivos}}$$

$$LR(-) = \frac{1 - \text{Sensibilidad}}{\text{Especificidad}} = \frac{\text{Tasa de falsos negativos}}{\text{Tasa de verdaderos negativos}}$$

  Entonces, un LR (+) de mayor magnitud (>10) significa que es importante utilizar un test diagnóstico, puesto que permite confirmar con certeza la presencia de enfermedad, y un LR (−) con un valor bajo (<0.1) indica que se descarta la enfermedad y por lo tanto no es necesario realizar el *test* diagnóstico.

  Salas (1996) nos menciona que para evaluar la significación global de un modelo se debe utilizar el estadístico de razón de verosimilitud (ERV) y se define como:

$$ERV = -2\left[ln\frac{L(R)}{L(MV)}\right]$$

$$= -2\{ln[L(R)] - ln[L(MV)]\}$$

$$= \{-2ln[L(R)]\} - \{-2ln[L(MV)]\} \qquad \sim(\chi^2_{k-1})$$

Donde:

L(MV) es la función de verosimilitud para el modelo formulado y L(R) la función de verosimilitud para el modelo restringido en el que únicamente se considera al término independiente o constante.

El ERV sigue una distribución chi-cuadrado con (k-1) grados de libertad donde k es el número de parámetros incluidos en el modelo formulado que han sido estimados por máxima verosimilitud.

- **Prueba de Wald**

Según Larrañaga et al. (2005) el test de Wald es otra forma de realizar prueba de hipótesis de parámetros, sin embargo, tan solo puede ser usado para realizar la prueba de un único parámetro.

El test de Wald usa el denominado estadístico de Wald para la variable en estudio (x_j).

Entonces para dicha j-ésima variable de interés, el estadístico de Wald es $\frac{\widehat{\beta_J}}{\widehat{S_{\beta_J}}}$

Donde: $\widehat{\beta_J}$ son las estimaciones máximo verosímiles y $\widehat{S_{\beta_J}}$ es su correspondiente desviación estándar.

Además, se cumple que $\frac{\widehat{\beta_J}}{\widehat{S_{\beta_J}}} \sim N(0,1)$ o lo que es equivalente $(\frac{\widehat{\beta_J}}{\widehat{S_{\beta_J}}})^2 \sim \chi_1^2$

La prueba de hipótesis que se plantea es la siguiente:

$$H_0: \beta_j = 0$$

$$H_1: \beta_j \neq 0$$

Por lo que la distribución del estadístico de Wald para el j-ésimo parámetro sirve para aceptar o rechazar la hipótesis nula establecida.

- **Prueba de Hosmer - Lemeshov**

Según Ramírez y Rodriguez (2014) la prueba Hosmer - Lemeshow es utilizado para evaluar la bondad de ajuste del modelo de la regresión logística. El ajuste será bueno cuando un valor alto de la probabilidad predicha (p) se relaciona con el resultado 1 de la variable binomial dependiente, mientras que un valor bajo de la probabilidad predicha p (próximo a cero) se relaciona en la mayoría de las ocasiones con el resultado *Y = 0* de la variable dependiente.

La prueba consiste en calcular, para cada una de las observaciones del conjunto de datos las probabilidades que predice el modelo para la variable dependiente, luego las ordena, agrupa para calcular a partir de ellas las frecuencias esperadas y compararlas con las observadas mediante la prueba de χ^2, considerando además que los valores esperados nulos o muy pequeños (menores de cinco) no se computa.

Según Llopis (2014) la prueba Hosmer - Lemeshow es un test donde se evalúa la distancia entre un valor observado O_i y un valor esperado e_i planteándose la siguiente prueba de hipótesis:

H_0: El modelo de Regresión Logística se ajusta a los datos.

H_1: El modelo de Regresión Logística No se ajusta a los datos.

$$\sum \frac{(O_i - e_i)^2}{e_i}$$

Buscándose finalmente que la prueba no sea significativa, lo contrario de lo habitual.

f. Modelización

Gill (2007) menciona que la selección de variables cumple un rol importante al momento de desarrollar un modelo, debido a que al adicionar variables que no contribuyen ocasionaría distorsión en el modelamiento.

Se menciona dos métodos para seleccionar las variables que deben entrar en el modelo, estas son:

- Backward: Esta técnica inicia cuando el modelo contiene a todas las variables regresoras y en cada paso se elimina las variables menos influyentes.

- Fordward: Esta técnica inicia teniendo el modelo sin ninguna variable y se va ingresando en cada paso las variables con mayor importancia.

2.2.3. Random Forest

Según Carranza (2019) Random Forest o bosque aleatorio es un método de aprendizaje conjunto o ensemble learning para la clasificación o regresión.

Es una técnica que consiste en combinar varios predictores con el objetivo de obtener uno más "fuerte" que pueda realizar mejores predicciones. En el caso de clasificación, consiste en una colección de clasificadores con estructura de árbol (Carranza ,2019).

$$\{h(x, \theta_k), k = 1, \dots\}$$

Donde:

- $\{\theta_k\}$, son vectores aleatorios independientes e idénticamente distribuidos, y representa los parámetros para la construcción del k- esimo árbol.

$h(x, \theta_k)$, es un clasificador donde *x* es el vector de entrada.

Luego, dada una entrada *x*, cada árbol emite un único voto para la elección de la clase más popular para *x*.

Gislason (2006) expresa que Breiman et al. 1984 indica que el algoritmo Random Forest crea múltiples Árboles CART cada uno entrenado en una muestra bootstrapped de los datos de entrenamiento originales y para dividir cada árbol en nodos lo hace a través de una búsqueda de un subconjunto seleccionado al azar de variables de entrada buscando siempre minimizar la correlación entre los árboles del conjunto. Asimismo, para determinar la clasificación de una instancia se hace a través del voto mayoritario y que cada árbol emite un solo voto.

El algoritmo Random Forest puede manejar datos dimensionales altos y usar una gran cantidad de árboles en conjunto.

También expresa que Breiman muestra que el tiempo computacionalmente para el análisis de Random Forest esta dado por:

$$cT\sqrt{M}Nlog(N)$$

Donde:

- c, es una constante.
- T, es el número de árboles en el conjunto.
- M, es el número de variables.
- N, es el número de muestras en el conjunto de datos.

Cabe señalar que, aunque Random Forest no son computacionalmente intensivos, requieren de una buena cantidad de memoria ya que almacenan una matriz N x T en memoria.

Se puede estimar la importancia de la variable m al permutar aleatoriamente todos los valores de la variable m^{th} out of bag en las muestras para cada clasificador. Si se produce un mayor out-of-bag error, eso es una indicación de la importancia de esa variable.

Según Strobl et al. (2008) argumentaban que se está volviendo cada vez más popular el algoritmo Random Forest en muchos campos científicos ya que puede hacer frente a problemas como: base de datos pequeños, base de datos con muchas variables, interacciones complejas e incluso variables predictoras altamente correlacionadas (situación que a menudo ocurre en bioinformática) así como por su alta precisión predictiva sugiriendo en la selección de variables predictoras relevantes para el análisis de datos de microarrays, secuenciación de ADN y otras aplicaciones. Sin embargo, advierte que en la determinación de las variables importantes se muestra un sesgo hacia las variables predictoras correlacionadas.

Lo mencionado en el párrafo anterior se confirma con lo expuesto por Espinosa (2020) quien argumenta que entre los métodos de aprendizaje automático más populares son el Random Forest y ello es debido al balance que ofrece entre complejidad y resultados, es decir puede trabajar con base de datos grandes, así como con cientos de variables ofreciendo un rendimiento similar en comparación con técnicas más complejas.

A la vez indica Carranza (2019) la técnica conocida como Random Forest, construye árboles basados en un subconjunto de variables de entrada elegidas al azar. Cada árbol es construido siguiendo el siguiente algoritmo:

- Si el número de muestras en el conjunto de entrenamiento es P, muestrear N casos aleatoriamente - pero con reemplazo, a partir de los datos originales. Esta muestra va a ser el conjunto de entrenamiento para la construcción del árbol.

- Si hay M variables de entrada, se especifica un número m << M, constante durante el crecimiento del bosque o Forest, tal que en cada nodo se seleccionen m variables al azar de las M. Posteriormente se eligen entre las m variables aquellas que mejor dividan al nodo, es decir, aquellas que generen al final un árbol compacto y simple.

- Cada árbol se construye hasta su máxima extensión posible. No hay pruning(poda).

a. Número de árboles

Según Brown (2016) menciona que a mayor cantidad de árboles es mejor porque permite detectar anomalías que se presentan en los datos. Pero debemos tener en cuenta que al ser un área de retornos decrecientes, cada árbol que se adiciona tendrá menos beneficios que los árboles que se construyeron antes de él. Eventualmente se estabilizará y a mayores árboles que se generan posteriormente, no aportarán mucho en la mejora del modelo.

Además, Breiman (2001) menciona que el tiempo de ejecución de 1000 árboles en comparación al de 100, la parte computacional tendrá un trabajo mayor en la ejecución.

b. Error out of bag (OOB)

Según Cutler et al. (2012) en el proceso de seleccionar una muestra bootstrap de los datos algunas observaciones o datos no son elegidos, esto son los denominados "out-of-bag data" (OOB) y son usados para estimar el error de generalización, así como para detectar la importancia de las variables.

El error de generalización para la regresión con pérdida de error al cuadrado, se estima utilizando out-of-bag del error cuadrático medio (MSE):

$$MSE_{oob} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f_{oob}}(x_i))^2$$

Donde: $\hat{f_{oob}}(x_i)$ es la predicción out-of-bag para la observación i.
La tasa de error de generalización para la clasificación con cero pérdidas, se estima usando la tasa de error out-of-bag:

$$E_{oob} = \frac{1}{N}\sum_{i=1}^{N} I(y_i \neq \hat{f_{oob}}(x_i))$$

Esto indica que se promedia las interacciones (I) donde la variable respuesta y_i es diferente a la predicción out-of-bag para la observación i.

Una idea errónea común es que la tasa de error out-of-bag para el forest se obtiene al promediar la tasa de error out-of-bag para cada árbol. Lo que realmente se usa es la tasa de error out-of-bag de las predicciones, esto nos permite obtener una tasa de error por clase, y un "matriz de confusión out-of-bag " mediante la tabulación cruzada de y_i y $\hat{f}_{oob}(x_i)$.

Para entender la tasa de error de predicciones out-of-bag, primero tenemos que tener en cuenta que, si los árboles son grandes, las predicciones para la variable respuesta de las observaciones que están dentro de los datos de entrenamiento usando todos los árboles serán demasiado optimistas. Por esta razón, la predicción de la variable respuesta para las observaciones que están en el conjunto de entrenamiento se hace usando árboles para los cuales la observación está fuera de bolsa (out-of-bag). Estas predicciones son llamadas predicciones out-of-bag.

El OOB se puede usar en la determinación del número de árboles de un algoritmo que usa la técnica Boosting en una construcción, tal como el Random Forest. Esta relación se puede observar en la grafica de la Figura 3, experimento desarrollado por Mease,D. y Wyner, A. (2008) en donde se infiere que existe una relación estrecha entra la tasa de error en los datos de prueba y la tasa de error out-of-bag, es decir que a medida que el número de árboles se incrementa la tasa de error para ambas medidas mencionadas se hace más pequeña y coincidentes.

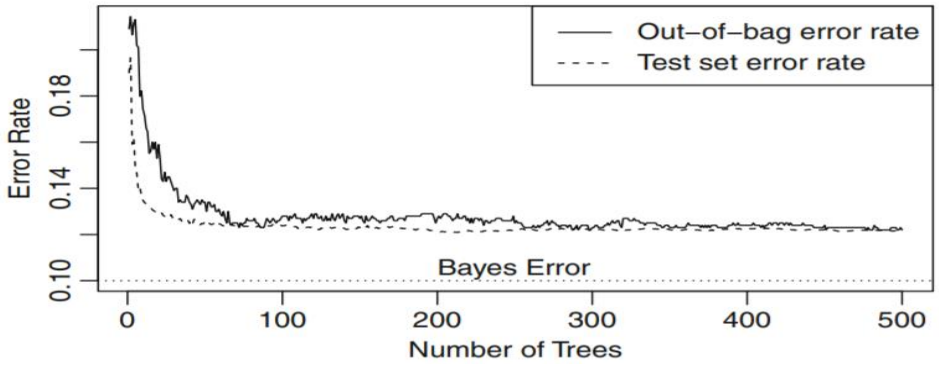


Figura 1: Out-of-bag error y Test set error rate.
Fuente: Mease, D. y Wyner, A. (2008).

Según Mitchel (2011) el parámetro número de variable muestreadas (m-try) tiene el mayor efecto sobre la capacidad predictiva real, mientras que los otros parámetros en estudio tales como el tamaño de la muestra o si la muestra es con o sin reemplazo, tuvieron poco efecto sobre la capacidad predictiva real en la mayoría de los casos. Sin embargo, estos parámetros tienen un gran efecto en la estimación del error OOB, que para ciertos parámetros causa un sesgo positivo severo. Este sesgo se reduce en gran medida mediante submuestreo sin reemplazo y eligiendo la misma proporc e observaciones de cada grupo para las muestras en la bolsa.

Todavía hay un pequeño sesgo positivo restante que resulta de la selección de variables, y realizar una validación cruzada puede refinar aún más la estimación del error. Sin embargo, dado que el sesgo es bajo, uno puede preferir simplemente informar el error OOB como un límite superior esperado del error real.

c. Importancia de variables

Se puede conocer la importancia de variables para el modelo mediante los indicadores mean decrece accuracy y mean decreace Gini.

- MDA (Mean Decreace accuracy)

  El indicador MDA mide en cuánto se reduce el error de clasificación cuando se incluye un predictor en el modelo, es decir este indicador se basa en la contribución de las variables al error de predicción.

- MGD (Mean DecreaseGini)

  Según Vaquerizo (2011) la medida MGD indica que mayor valor significa mayor importancia en los modelos creados, ya que valores próximos a 0 para el índice de Gini implican un mayor desorden y valores próximos a 1 implican un menor desorden. Entonces si computamos una medida de “decrecimiento” del índice de Gini significa que cuanto mayor sea esta medida más variabilidad aporta a la variable dependiente.

En el software R, la función random**VarImpsRF** permite determinar la importancia de las variables empleadas en los modelos, pero esto también implica mayores tiempos de ejecución.

d. Algoritmo Random Forest

Cutler et al. (2012) presenta a $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ como el conjunto de los datos de entrenamiento, con $x_i = (x_{i,1}, \ldots., x_{i,p})^T$, donde p son los predictores y *yi* la variable respuesta.

Los ensambles f son construidos en términos de una colección de los llamados "base learnings" $h_1(x), \ldots h_J(x)$ y estos "base learnings" (aprendices básicos) se combinan para dar el "predictor del ensamble" f_x. En regresión, los aprendices básicos se promedian:

$$f(x) = \frac{1}{J}\sum_{j=1}^{J} h_j(x)$$

mientras que, en clasificación f_x es la clase predicha de mayor frecuencia ("votación")

$$f(x) = arg\ max_{y \epsilon \mathcal{Y}} \sum_{j=1}^{J} I(y = h_j(x))$$

En Random Forests, el j-ésimo aprendizaje básico es un árbol denominado $h_j(X, \ominus_j)$, donde $\ominus_j$ es una colección de variables aleatorias y las $\ominus_j$'s son independientes para j=1,… J.

Entonces, Cutler et al. (2012) definen el algoritmo Random Forest en los siguientes pasos: Para j=1 a J:

1) Tomar una muestra bootstrap Dj de tamaño N de D.

2) Usando la muestra bootstrap Dj como datos de entrenamiento, se ajusta un árbol usando particionamientos binarios recursivos.

   a) Comience con todas las observaciones en un solo nodo.
   b) Repita los siguientes pasos de forma recursiva para cada nodo no dividido hasta que se cumple el criterio de parada:
      (i) Seleccione *m* predictores al azar de los *p* predictores disponibles.
      (ii) Encuentre la mejor división binaria entre todas las divisiones binarias de los m predictores del paso anterior (i).
      (iii) Divida el nodo en dos nodos descendientes utilizando la división del paso anterior (ii).

Para obtener una predicción en un nuevo punto *x*, se aplica las siguientes fórmulas:

$\hat{f}(x) = \frac{1}{J}\sum_{j=1}^{J} \hat{h_j}$, para regresión y

$\hat{f}(x) = \arg max_y \sum_{j=1}^{J} I(\hat{h_j}(x) = y)$ para clasificación

Donde $\hat{h_j}(x)$ es la predicción de la variable de respuesta en x usando el j-ésimo árbol

e. Descripción de la librería Random forest

- Definiciones de algunos Parámetros:

  Ntree: Número de árboles que se genera.
  Mtry: El número de variables que serán elegidas de manera aleatoria para la división de cada nodo. El valor predeterminado es la raíz cuadrada de donde p es el número de variables x.
  Replace: El comando indica si desea que el muestreo sea con reemplazo o sin reemplazo.
  Classwt: El comando permite dar prioridad a una clase. Esto se aplica para la variable respuesta.

Nodesize: Indica el mínimo tamaño de nodos terminales.

Maxnodes: Máximo tamaño de nodos terminales que pueden tener los árboles en el bosque.

Importance: TRUE activa el cálculo de la importancia de las variables, es decir muestra las variables que son más importantes para el modelo.

2.2.4. Árbol de Decisión

Bouza y Santiago (2019) indican que los Árboles de Decisión determinan una regla de decisión, resultando ser una herramienta de clasificación muy potente y que su popularidad se debe a la facilidad de entendimiento de sus resultados para cualquier usuario. "La técnica de árboles de decisión permite la:

- **Segmentación:** Establecer que grupos son importantes para clasificar un cierto ítem.
- **Clasificación:** Asignar ítems a uno de los grupos en que está particionada una población.
- **Predicción:** Establecer reglas para hacer predicciones de ciertos eventos.
- **Reducción de la dimensión de los datos:** Identificar que datos son los importantes para hacer modelos de un fenómeno.
- **Identificación-interrelación:** Identificar que variables y relaciones son importantes para ciertos grupos identificados a partir de analizar los datos.
- **Recodificación:** Discretizar variables o establecer criterios cualitativos perdiendo la menor cantidad posible de información relevante".

Según Solarte y Soto (2011) sobre las propiedades de los Árboles de Decisión explican que una propiedad importante de esta técnica es que permite la organización eficiente del conjunto de datos, y esto se debe a que los árboles son construidos a partir del primer nodo raíz evaluándole y de acuerdo al valor que adopta se va descendiendo en las ramas hasta llegar al final del camino que son las hojas del árbol, donde las hojas representan las clases y el nodo raíz representa todos los patrones.

Diaz (2012) argumenta que para ajustar árboles CART con los valores predeterminados de los parámetros en la librería rpart se utiliza la instrucción rpart ($Y \sim X_1 + X_2 + \cdots + X_p$) donde *Y* es la variable respuesta y $X_1, X_2, .., X_p$son las variables predictoras. Si *Y* es discreta la función ajusta un árbol de clasificación y si es continua un árbol de regresión. En el estudio de simulación realizado en este trabajo se tiene solo una variable predictora *X*, por tanto, la instrucción utilizada para ajustar los árboles de regresión es rpart (Y ~ X).

a. Clases de algoritmo de árboles de clasificación

Moreno et al. (2016) indica que los algoritmos más conocidos que generan árbol de clasificación son:

- CART: Genera solo árboles binarios, es decir de cada nodo se desprende exactamente dos ramas.

  El árbol CART es un método de regresión usado para predecir valores de variables continuas, pero cuando los supuestos para aplicar este modelo no se cumplen sus conclusiones pueden ser erróneas. Los árboles de regresión CART es un método muy fácil de interpretación de resultados. Los CART utilizan datos históricos, los cuales se usan para construir árboles de regresión que permiten la clasificación y la predicción de nuevos datos, estos tienen como ventaja que pueden manipular con facilidad variables numéricas.

  Sus principales características son: la robustez a outliers o valores atípicos, la invariancia en la estructura de sus árboles de clasificación a transformación monótonas de las variables independientes, y la interoperabilidad.

  Según Loh (2011) CART usa una generalización de la varianza binomial llamada índice de Gini. Primero crecen un árbol demasiado grande y luego lo podan a un tamaño más pequeño para minimizar una estimación del error de clasificación errónea empleando a la vez validación cruzada 10 veces. CART está implementado en R *system*7 como RPART.

- CHAID: La variable (dependiente) puede ser continua y categórica. Sin embargo, las variables predictoras(independientes) son variables categóricas solamente (pueden ser más de dos categorías).

- C4.5: Según Loh (2011) la técnica C4.5 es un algoritmo desarrollado por Ross Quinlan para generar un árbol de decisión y es una extensión del algoritmo ID3. Para su función de impureza usa la entropía. Los árboles de decisión generados por C4.5 pueden ser usados para clasificación, y por esta razón, casi siempre es referido como un clasificador estadístico.

- C5: Joaquín (2017) C5 es el algoritmo sucesor de C4.5, ambos desarrollados por Quinlan, con el objetivo de crear árboles de clasificación. Se destaca por la capacidad de generar árboles de predicción simples, generar modelos basados en reglas, ensembles basados en la técnica boosting y la asignación de distintos pesos a los errores. Todas estas capacidades mencionadas son accesibles mediante el paquete C50.

b. Árbol de clasificación CART

Según Breiman et al. (1984) define al algoritmo CART como un método no paramétrico de segmentación binaria donde el árbol es construido dividiendo repetidamente los datos. En cada división los datos son partidos en dos grupos mutuamente excluyentes. El nodo inicial es llamado nodo raíz o grupo madre y se divide en dos grupos hijos o nodos, luego el procedimiento de partición es aplicado a cada grupo hijo por separado. Para dividir los datos se requiere un criterio de particionamiento el cual es determinado por la medida de impureza.

Las divisiones se seleccionan de modo que “la impureza” de los hijos sea menor que la del grupo madre y estas están definidas por un valor de una variable explicativa
(Deconinck et al. 2006).

Según Timofeev (2004) el análisis de árboles de clasificación y regresión (CART) generalmente consiste en tres pasos:

1. Construcción del árbol máximo.
2. Poda del árbol.
3. Selección del árbol óptimo mediante un procedimiento de validación cruzada ("cross-validation").

c. Construcción de árbol de clasificación

El árbol máximo es construido utilizando un procedimiento de partición binario, comenzando en la raíz del árbol. Este árbol es un modelo que describe el conjunto de entrenamiento (grupo de datos original) y generalmente es sobreajustado, es decir, contiene gran cantidad de niveles y nodos que nos proporciona una mejor clasificación y puede ser demasiado complejo. Cada grupo es caracterizado por la distribución (respuesta categórica) o por la media (respuesta numérica) de la variable respuesta, el tamaño del grupo y los valores de las variables explicativas que lo definen. Gráficamente, el árbol se representa con el nodo raíz (los datos sin ninguna división) al iniciar y las ramas y hojas debajo (cada hoja es el final de un grupo).

Calidad del Nodo (Función de Impureza)

La función de impureza es una medida que permite determinar la calidad de un nodo, esta será denotada por $i(t)$.

Existen varias medidas de impureza (criterios de particionamiento) que nos permiten analizar varios tipos de respuesta, las tres medidas más comunes presentadas por Breiman et al. (1984) para árboles de clasificación son:

- **El índice de información o entropía el cual se define como:**

$$i(t) = \sum_j p(j|t) \ln p(j|t)$$

El objetivo es encontrar la partición que maximice $\Delta i(t)$ en la ecuación:

$$\Delta i(t) = -p(j|t) \ln p(j|t)$$

donde j = 1, ..., k es el número de clases de la variable respuesta categórica y $p(j|t)$ la probabilidad de clasificación correcta para la clase j en el nodo t.

- **El índice Gini tiene la forma**

$$i(t) = \sum_{i \neq j} p(j|t) \ln p(i|t)$$

Encontrar la partición que maximice Δi(t) en:

$$\Delta i = -\sum_{j=1}^{k} [p_j(t)]^2$$

Este índice es el más utilizado. En cada división el índice Gini tiende a separar la categoría más grande en un grupo aparte, mientras que el índice de información tiende a formar grupos con más de una categoría en las primeras decisiones.

- **El índice "Towing".**

A diferencia del índice Gini, Towing busca las dos clases que juntas formen más del 50 % de los datos, esto define dos "super categorías" en cada división para las cuales la impureza es definida por el índice Gini. Aunque el índice towing produce árboles más balanceados, este algoritmo trabaja más lento que la regla de Gini (Deconinck et al., 2006). Para usar el índice towing se selecciona la partición que maximice:

$$\frac{p_L p_R}{4} \left[\sum_j |p(j|t_L) - p(j|t_R)| \right]^2$$

Donde:

- $t_L \ y \ t_R$ representan los nodos hijos izquierdo y derecho respectivamente.
- P_L y P_R representan la proporción de observaciones en t que pasaron a $t_L \ y \ a \ t_R$ en cada caso.

d. Poda del árbol

El árbol que se obtiene con el algoritmo esta generalmente sobreajustado por lo tanto es necesario ser podado, cortando sucesivamente ramas o nodos terminales hasta encontrar el tamaño "adecuado" del árbol.

Según Breiman et al. (1984) introducen algunas ideas básicas para resolver el problema de seleccionar el mejor árbol. Una forma es buscar una serie de árboles anidados de tamaños decrecientes (Death & Fabricius, 2000) cada uno de los cuales es el mejor de todos los árboles de su tamaño. Estos árboles pequeños son comparados en función de costo complejidad R_α(T) para determinar el árbol óptimo.

Según Deconinck et al. (2006) para cada árbol *T*, la función costo - complejidad se define como:

$$R_\alpha(T) = R(T) + \alpha\left|\tilde{T}\right|$$

Donde:

- R(T) es el promedio de la suma de cuadrados entre los nodos, puede ser la tasa de mala clasificación total o la suma de cuadrados de residuales total dependiendo del tipo de árbol.
- $\left|\tilde{T}\right|$, es la complejidad del árbol, definida como el número total de nodos del sub-árbol.
- α , es el parámetro de complejidad.

El parámetro α es un número real mayor o igual a cero. Cuando $\alpha = 0$ se tiene el árbol más grande y a medida que α se incrementa se reduce el tamaño del árbol.

La función R_α(T) siempre será minimizada por el árbol más grande, por tanto, se necesitan mejores estimaciones del error, para esto Breiman et al. (1984) proponen obtener estimadores "honestos" del error por "validación cruzada". Computacionalmente el procedimiento es exigente pero viable, pues solo es necesario considerar un árbol de cada tamaño, es decir, los árboles de la secuencia anidada.

e. Selección del árbol optimo

Serna (2009) explica que es necesario seleccionar el árbol óptimo a partir de la secuencia de árboles anidados obtenidos y para ello no es efectivo la comparación o penalización de la complejidad en la elección del árbol óptimo (De'ath & Fabricius, 2000) entonces lo que se requiere es estimar con precisión el error de predicción y generalmente esta estimación se obtiene utilizando el procedimiento de validación cruzada.

El procedimiento de validación cruzada que se implementa dado que no se cuenta con suficientes datos es la:

- **Validación cruzada con partición en V, (v-fold Cross validation).**

  La Validación cruzada consiste en extraer de la muestra de aprendizaje una muestra de prueba que será utilizado como datos nuevos. La muestra de aprendizaje es usada para calcular los estimadores del modelo y el subconjunto extraído ósea la muestra de prueba es usada para verificar el desempeño de los estimadores. El error en la predicción es usado para conocer el desempeño del modelo, el cual es acumulado para obtener el error medio absoluto de la muestra de prueba.

  La metodología CART generalmente utiliza Validación Cruzada con partición en V (v-fold cross validation) teniendo V = 10 y el proceso es el siguiente:

- Se divide la muestra en diez grupos mutuamente excluyentes de tamaño aproximadamente igual.

- Luego se extrae un grupo por vez y se construye el árbol con los datos de los grupos restantes. El árbol construido es usado para predecir la respuesta del grupo extraído o eliminado.

- Con el paso anterior se calcula el error estimado y esta se repite para las 10 extracciones y construcción de los árboles, obteniéndose por lo tanto la estimación de 10 errores.

- Por último, se selecciona el árbol con el menor error estimado (menor tasa de mala clasificación).

  El flujograma del algoritmo CART se muestra en la Figura 1.

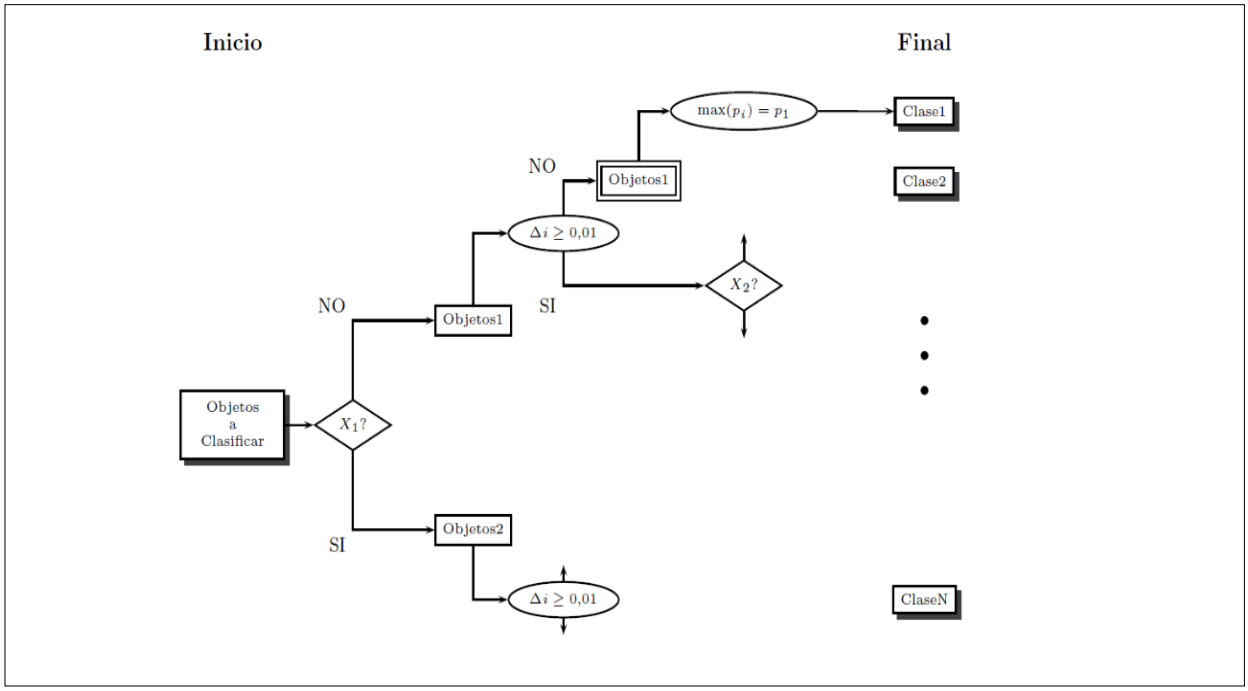


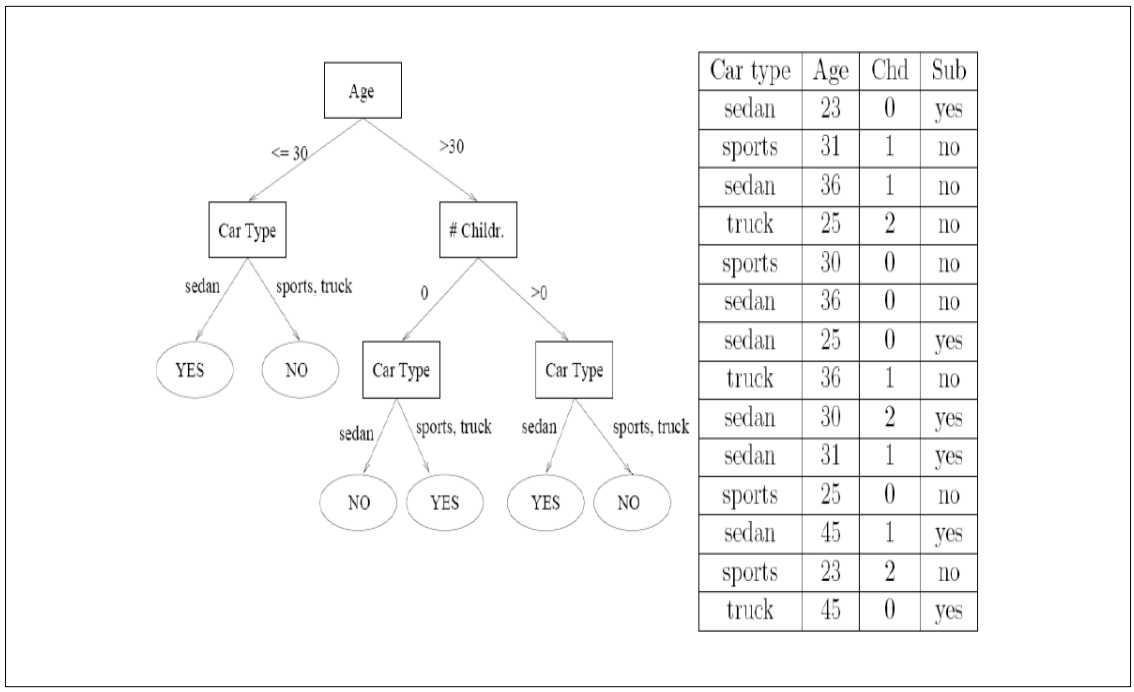
Figura 2: Diagrama de flujo del algoritmo CART**.**
Fuente: De'ath & Fabricius (2000).

Como ejemplo de funcionamiento del algoritmo CART se tiene el árbol y los datos en la Figura 2, donde se quiere determinar un conjunto de reglas que indiquen si un conductor vive o no en los suburbios.

Concluyéndose que:

- Si Age ≤ 30 y Car Type = Sedan, entonces el conductor Si vive en los suburbios.
- Si Age ≤ 30 y CarT ype = truck/Sports, entonces el conductor No vive en los suburbios.
- Si Age > 30, Children = 0 y Car Type = Sedan, entonces el conductor No vive en los suburbios.

- Si Age > 30, Children = 0 y Car Type = truck/Sports, el conductor Si vive en los suburbios.
- Si Age > 30, Children > 0 y CarT ype = Sedan, entonces el conductor Si vive en los suburbios.
- Si Age > 30, Children > 0 y CarT ype = truck/Sports entonces el conductor No vive en los suburbios.



Car type	Age	Chd	Sub
sedan	23	0	yes
sports	31	1	no
sedan	36	1	no
truck	25	2	no
sports	30	0	no
sedan	36	0	no
sedan	25	0	yes
truck	36	1	no
sedan	30	2	yes
sedan	31	1	yes
sports	25	0	no
sedan	45	1	yes
sports	23	2	no
truck	45	0	yes

Figura 3: Árbol de clasificación**.**
Fuente: Dobra (2002)

2.2.5. Método de Ensamble Stacking

Campo y Cruz (2017) mencionan que el método Stacking es un tipo de ensamble que es definido en el ajuste de múltiples modelos de diferentes tipos para obtener un modelo final (supervisor) que aprende a combinar las predicciones de los modelos primarios. La superioridad de los resultados con el método Stacking es por tres motivos: la reducción del sesgo, disminución de la varianza y la probabilidad baja de sobre-ajuste.

Además, Boruel (2012) indica que el método Stacking combina varios estimadores provenientes de distintos métodos de aprendizaje. A partir de toda la muestra de

entrenamiento, se obtienen *M* modelos, las cuales pueden ser: $g_1, g_2, \dots, g_M$ proveniente de los *M* algoritmos de aprendizaje. Finalmente, para clasificar un nuevo dato se utilizan los modelos $g_1, g_2, \dots, g_M$. La predicción final sobre un nuevo dato es la siguiente:

$$f(x) = g(x, g_1(x), \dots, g_M(x))$$

Asimismo, Ting y Witten (1999) definen al método Stacking, como la combinación de múltiples clasificadores generados por diferentes algoritmos de aprendizaje $L_1, \dots, L_N$ en un solo conjunto de datos S, que está compuesto por un vector de características $s_i = (x_i, yi)$.

Según Caffe et al. (2011) el método de ensamble Stacking es una combinación de clasificadores heterogéneos que aprende de los errores de los métodos anteriores para tener predicciones más estables.

Según Clarke (2003) la principal idea en Stacking es combinar $f_1, f_2, \dots, f_m$ capas por la técnica de cross-validation. La idea es que los modelos estén en capas fi con pesos α_i. Además, define a los vectores:

$$z_j = \left(z_{1j}, \dots, z_{mj}\right) = (f_1^{-j}(x_j), \dots, f_m^{-j}(x_j))$$

En el que el superíndice − j significa que la j^{th} observación (y_i, x_j) no se utiliza para estimar los coeficientes en cada f_i que luego se evalúan en la eliminación x_j. El α se elige para minimizar

$$L = \sum_j (y_j - \sum_i \alpha_i z_{i,j})^2$$

Hay varias opciones, que conduce a diferentes técnicas para optimizar *L* y numerosas variantes en la validación cruzada de una salida (Clarke, 2003).

Segrera y Moreno (2006) definen el método Stacking como la combinación de múltiples clasificadores generados por distintos algoritmos de aprendizaje $L_1, L_2, \dots, L_n$ para un mismo conjunto de datos *S*. En la primera fase, genera clasificadores del nivel $C_1, \dots, C_n$, donde

$C_i = L_i(S)$. En la segunda fase, un clasificador en el meta-nivel combina las salidas provenientes de los clasificadores del nivel base que trabajan en paralelo.

Entonces el principal éxito del método Stacking, se debe a la capacidad de combinar las predicciones del conjunto de clasificadores.

Según Padmapani et al. (2018) el ensemble Learning consiste en la combinación de varios modelos por medio de diferentes técnicas como Stacking. Por lo tanto, para obtener una mejor predicción se combina diferentes familias de minería de datos.

La combinación de clasificadores en la actualidad es un área activa de investigación en el aprendizaje automatizado y el reconocimiento de patrones. Se han publicado numerosos estudios teóricos y empíricos que demuestran las ventajas del paradigma de combinación de clasificadores por encima de los modelos individuales (Díaz et al., 2015).

Según Beltrán et al. (2012) el método Stacking combina múltiples clasificadores a través de diferentes algoritmos de aprendizaje. Los algoritmos de aprendizaje de la primera fase pueden ser Árbol de Decisión, Redes neuronales, Máquinas de vectores soporte, Regresión Logística, Random Forest, etcétera. En una segunda fase otro clasificador combina predicciones. Este esquema funcionará cuando todos los modelos utilizados tienen una precisión aceptable.

Villarino (2015) explica que se proponen métodos de ensamble de clasificadores mediante la técnica de Stacking porque el objetivo es mejorar la precisión alcanzada por los modelos individuales de clasificación y reducir la varianza de los errores cometidos.

Campo y Cruz (2017) defienen a Stacking como un tipo de ensamble que consiste en el ajuste de múltiples modelos de diferentes tipos presentando un modelo final denominado “supervisor” quien es el que ha aprendido a combinar las predicciones de los modelos primarios.

Stacking combina múltiples clasificadores en dos fases a través de diferentes algoritmos de aprendizaje. Los algoritmos de aprendizaje de la primera fase pueden ser árboles de decisión,

redes neuronales, máquinas de soporte vectorial, regresión logística, etcétera. En la segunda fase otro clasificador combina las salidas de los modelos iniciales, siendo estos por voto mayoritario. Este esquema se observa en la figura 5 y funcionará bien si todos los modelos utilizados tienen una precisión aceptable (Beltrán, 2012).

Cada clasificador del nivel de base predice una distribución de probabilidad (DP) sobre los posibles valores de la clase. La predicción del clasificador de base C aplicado al ejemplo *x* será: $p^c(x) = (p^c(c_1\backslash x), \ldots, p^c(c_m\backslash x))$

Donde:

- {c1, c2, …, cm}, es el conjunto de posibles valores de la clase.
- $p^c(c_i\backslash x)$, es la probabilidad estimada (y predicha) por el clasificador.
- C, indica pertenencia de la variable *x* a la clase c_i.

La clase c_j con la mayor probabilidad es la que predice $p^c(c_j\backslash x)$ el clasificador C.
Los atributos del meta-nivel son las probabilidades predichas para cada posible clase por cada uno de los clasificadores del nivel base.

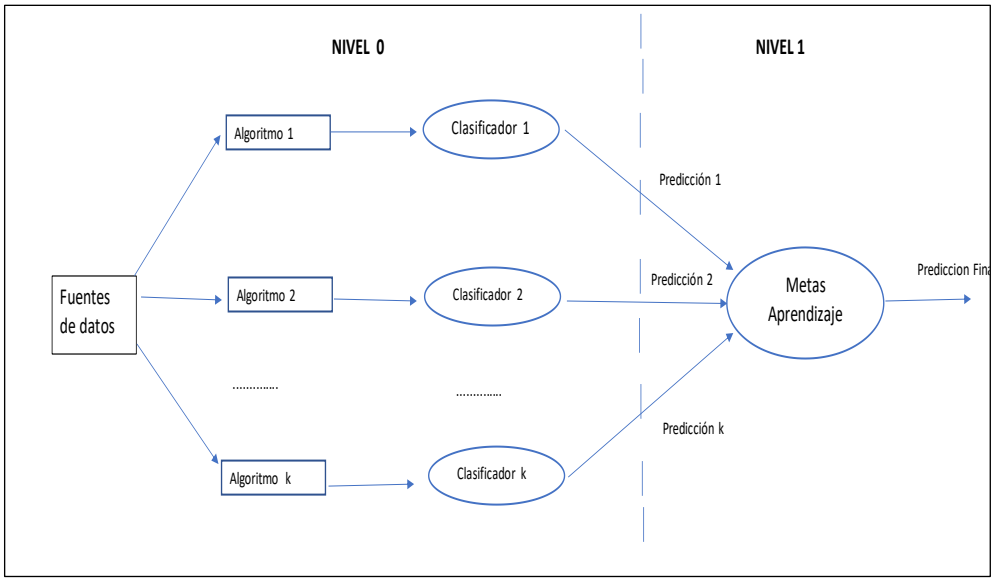


Figura 4: Estructura del método Stacking (Proceso del método

Fuente**:** Beltrán et al. (2012).

a. Algoritmo Stacking

Según Campo et al. (2017) define el algoritmo en los siguientes pasos:

The Stacking Algorithm

Imput: $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$

Primer nivel algoritmo de aprendizaje $L_1, \dots., L_T$;

Segundo nivel algoritmo de aprendizaje L.

Process:

For t=1,...,T:

$h_t = L_t(D)$ % train firts-level h_t individual learner aplicando the firts-level

End; %learning L_t algorithm de original data set D

$D' = O$; %generate a new data set

For $i = 1, \dots., m$:

For $t = 1, \dots., T$:

$z_{it} = h_t(x_i)$ % Use h_t to classify the training example x_i

End;

$D' = D' \cup \{\big((z_{i1}, z_{i2}, \dots, z_{iT}), y_i\big)\}$

End;

$h' = L(D')$ % Train the second-level learner h' by applying the second-level

% learning algorithm L to the new data set D'

Output:

$$H(x) = h'(h_1(x), \dots \dots, h_T(x))$$

2.2.6. Indicador de comparación

a. Receiver Operating Characteristic (ROC)

Según Jiménez (2012) la curva ROC (Receiver Operating Characteristic) es una metodología desarrollada para analizar un sistema de decisión. Consiste en una representación gráfica.

También Fan et. al (2006) el área bajo la curva ROC (Auc) es ampliamente reconocida como la medida de la discriminación discriminatoria de una prueba diagnóstica.

El valor máximo para el Auc es 1.0, por lo que indica una prueba (teóricamente) perfecta (es decir, 100% sensible y 100% específico).

En la curva ROC se sitúa en el eje de las abscisas el valor de uno menos la especificidad y en el eje de las ordenadas el valor de la sensibilidad. Por esto, la situación ideal es estar cerca del vértice superior izquierdo, ya que en este caso habría mucha sensibilidad y especificidad. La curva ROC nos muestra el desempeño del clasificador en todo su rango operativo. Puede ser usada para visualizar y seleccionar el mejor clasificador, aquel que por un lado maximice los verdaderos positivos y negativos, por el otro que minimice los falsos positivos. En función del tipo de problema puede cambiar el criterio usado para el clasificador.

El área bajo la curva ROC representa la probabilidad de que un clasificador ordene una instancia positiva elegida aleatoriamente más alta que una negativa. El Auc nos permite comparar clasificadores.

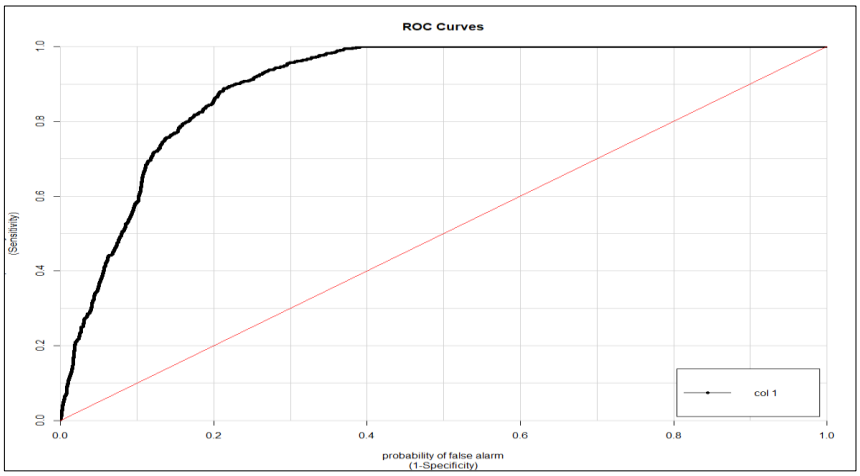


Figura 5: Curva ROC.

Fuente: Elaboración propia.

En el Tabla 2, se aprecia una tabla de doble entrada, la cual representa la clase real y la clase precedida. La cual se producen 4 posibles resultados.

Tabla 2: Matriz de confusión

		Clase Predecida	
		Negativo	**Positivo**
Clase Real	**Negativo**	Verdadero Negativo **VN**	Falso Positivo **FP**
	Positivo	Falso Negativo **FN**	Verdadero Positivo **VP**

Según Widmann (2019):

VP (Verdaderos positivos): Instancias correctamente reconocidas por el sistema.
FN (Falsos negativos): Instancias que son positivas y que el sistema dice que no lo son.
FP (Falsos positivos): Instancias que son negativas pero el sistema dice que no lo es.
VN (Verdaderos negativos): Instancias que son negativas y correctamente reconocidas como tales.

Indiciadores relacionados a la curva ROC son:

- **Sensibilidad:**

  Según Widmann (2019) sensibilidad viene a ser la proporción de casos positivos reales que se identifican correctamente. También, puede ser definido como qué tan apto es el modelo para detectar eventos en la clase positiva. Por lo tanto, dado que los correos electrónicos no deseados son la clase positiva, entonces la sensibilidad cuantifica cuántos de los correos electrónicos no deseados reales se predicen correctamente como spam.

$$sensitivity = \frac{VP}{VP + FN}$$

- **Especificidad:**

  Según Widmann (2019) especificidad es la proporción de casos negativos reales que se identifican correctamente. También se puede definir a la vez la especificidad como qué tan exacta mide la asignación a la clase positiva, en este caso, una etiqueta de spam asignada a un correo electrónico.

$$specificity = \frac{VN}{FP + VN}$$

- **Correcta clasificación (*Accuracy*):**

  Según Guerrero (2018) la proporción de casos correctamente clasificados (VP y VN) como positivos y negativos N=VP+FN+FP+VN.
  Tasa de acierto:

$$Accuracy = \frac{VP+VN}{FN+FP+VP+VN}$$

- **Error de clasificación:**

  Según Guerrero (2018) la proporción de casos que no fueron clasificados correctamente (FP y FN) a partir del criterio establecido respecto a toda la clasificación. Suponiendo que N es el número del conjunto de datos de entrenamiento, N=VP+FN+FP+VN.

  La Tasa de error es:

$$\frac{FP + FN}{FN + FP + VP + VN}$$

Otros indicadores:

- **Kolmogorov Smirnov (**KS**):**

  KS o Kolmogorov-Smirnov es una medida del grado de separación entre las distribuciones positiva y negativa, valorando así el rendimiento de los modelos de clasificación. KS es 100, si las puntuaciones dividen a la población en dos grupos separados en los que un grupo contiene todos los positivos y el otro todos los negativos.

  Por otro lado, KS sería 0, si el modelo no puede diferenciar entre positivos y negativos, entonces es como si el modelo seleccionara casos de la población al azar.

  En la mayoría de los modelos de clasificación, KS estará entre 0 y 10, entonces cuanto mayor sea el valor, mejor será el modelo para separar los casos positivos de los negativos (Doob, 2016).

- **LogLoss:**

  Mide el rendimiento de un modelo de clasificación, donde la entrada de predicción es un valor de probabilidad del modelo. Entonces el indicador mide que tan cercana en promedio esta la probabilidad a la clase real de la variable respuesta. (Buja et al.,2005).

b. Desbalanceo de datos

Las técnicas más comunes que se usa frente a estos tipos de problema, es decir específicamente cuando el target presenta un desbalanceo son: oversampling, undersampling y Smote. Estos balanceos se realizan comúnmente cuando el target tiene un porcentaje de datos de interés menor o igual al 10%.

- **Undersampling**

  Según Wah et al. (2014) la técnica de balanceo undersampling consiste en descartar muestras de la clase mayoritaria para modificar la distribución de clases, hasta igualar a la clase minoritaria y así equilibrar las muestras para el entrenamiento del modelo.

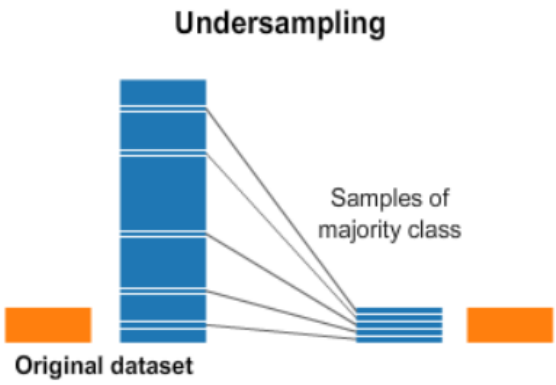


Figura 6: Funcionamiento de Undersampling.
Fuente: Fawcett (2016).

- **Oversampling**

Según Wah et al. (2014) la técnica de balanceo oversampling replica aleatoriamente instancias minoritarias (clase menor en proporción del target) para aumentar su población y así equilibrar a la clase mayoritaria.

La desventaja de esta técnica es que puede llevar a un sobreajuste, ya que hace copias exactas de la minoría.

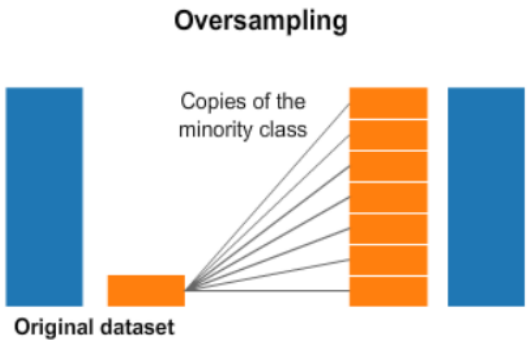


Figura 7: Funcionamiento Oversampling.
Fuente: Fawcett (2016).

- **SMOTE (Syntetic Minority Over-sampling Technique)**

SMOTE es un algoritmo de oversampling que genera instancias "sintéticas" o artificiales para equilibrar la muestra de datos basado en la regla del vecino más cercano. La generación se realiza extrapolando nuevas instancias en lugar de duplicarlas como hace el algoritmo de Resampling. Para cada una de las instancias minoritarias se buscan las instancias minoritarias vecinas (más cercanas) y se crean N instancias entre la línea que une la instancia original y cada una de las vecinas. El valor de N depende del tamaño de oversampling deseado. Para un caso del 200% por ejemplo, por cada instancia de la clase minoritaria deben crearse dos nuevas instancias genéricas (Moreno et al .,2009).

También Moreno et al. (2009) define a SMOTE como un algoritmo de sobre-muestreo de ejemplos utilizado para la clase minoritaria. Crea ejemplos sintéticos en lugar de hacer un sobre-muestreo con reemplazo y opera en el espacio de atributos feature space, en lugar del espacio de datos data space. Crea un ejemplo sintético a lo largo de los segmentos de línea que unen alguno o todos los k vecinos más cercanos de la clase minoritaria. Se eligen algunos de los k vecinos más cercanos de manera aleatoria (no se utilizan todos). SMOTE utiliza típicamente k = 5.

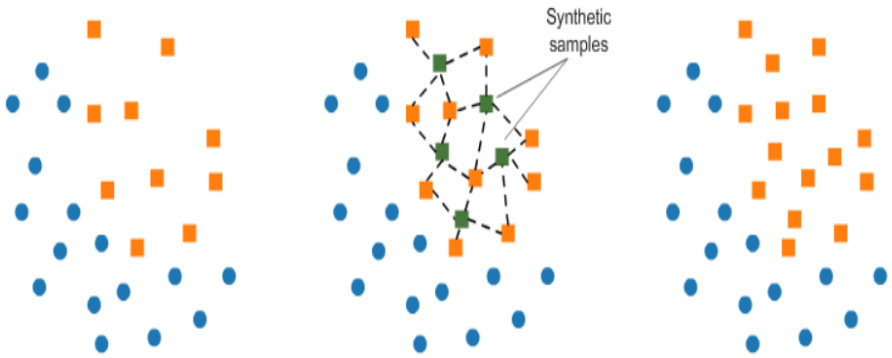


Figura 8: Funcionamiento del SMOTE.
Fuente: Fawcett (2016).

c. Cross Validation

Espinar (2018) explica que probablemente el método más simple y más usado para estimar el error de predicción es validación cruzada (Cross Validation). Esta estrategia implica dividir el conjunto de observaciones en dos partes, una de ellas será usada como conjunto de entrenamiento y la otra parte será usada como conjunto de validación o también conocido como conjunto hold-out.

Validación Cruzada con partición en *k*-subconjuntos es un método que consiste en dividir aleatoriamente el conjunto de observaciones en k subconjuntos de igual tamaño aproximadamente. El primer subconjunto se usa como conjunto de validación, del cual ya se puede calcular el error cuadrático medio MSE_1, y el resto como conjunto de entrenamiento para ajustar el modelo. Este procedimiento es repetido k veces, cada vez con un subconjunto

diferente para el conjunto de validación y se obtiene $MSE_1, \ldots, MSE_K$. errores cuadráticos medios.

Por lo tanto, para calcular la estimación del error del modelo usamos:

$$CV = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

Según Gupta (2017) en la validación cruzada de K Fold, los datos se dividen en k subconjuntos. El método consiste en retener uno de los k subconjuntos para ser el conjunto de test /conjunto de validación y juntándose los otros k-1 subconjuntos para formar un conjunto de entrenamiento en donde se formará el modelo, entonces en sí el método de retención se repetirá k veces, tal como se observa en la figura 9. Para conocer la efectividad total del modelo se promedia las estimaciones de los errores en todos los k ensayos.

Por lo tanto, en la validación cruzada de K Fold se observa que cada punto de datos llega a estar en un conjunto de validación exactamente una vez y llega a estar en un conjunto de entrenamiento k-1 veces, reduciendo esto r significativamente el sesgo, ya que usamos la mayoría de los datos para el ajuste y también reduce significativamente la varianza, ya que la mayoría de los datos también se usan en el conjunto de validación. El intercambio de conjuntos de entrenamiento y prueba también aumenta la efectividad de este método. Como regla general y evidencia empírica, generalmente se prefiere K = 5 o K = 10, pero nada es fijo y puede tomar cualquier valor.

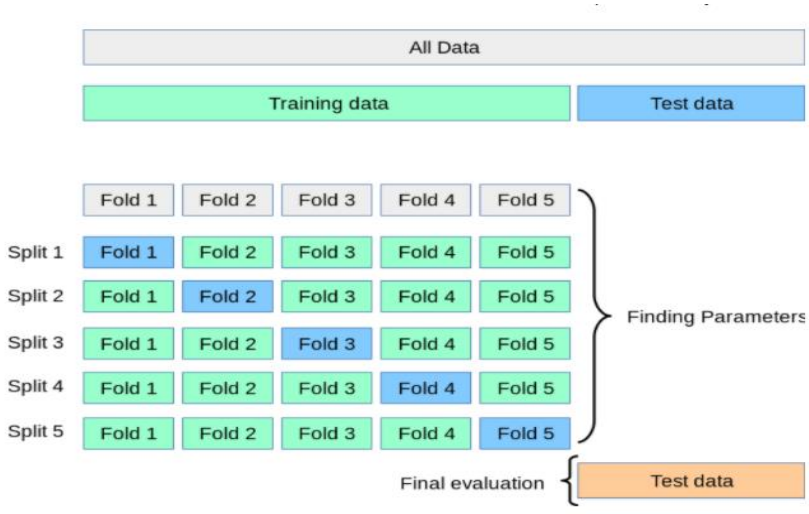


Figura 9: Cross Validation.

Fuente: Gupta (2017).

2.3. Definición De Términos Básicos

- **Desembolso:**

Desembolso es un contrato por el cual una entidad financiera pone dinero a disposición del cliente hasta un límite señalado y por un plazo determinado percibiendo periódicamente los intereses de las cantidades dispuestas y las comisiones fijadas.

Según Pedrosa (2016) define que un préstamo de una operación financiera por la cual una entidad financiera (prestamista) otorga mediante un contrato o acuerdo entre las partes, un activo (normalmente una cantidad de dinero) a otra persona (prestatario), a cambio de la obtención de un interés (precio del dinero).

Betancourt (2009) el tipo de operaciones, los bancos adoptan una posición acreedora frente a sus clientes. Por un lado, otorgan préstamos a sus clientes, acordando con ellos una retribución que pagarán en forma de intereses, en función al riesgo y costes operativos asumidos; por otro lado, también realizan inversiones con la intención de obtener una rentabilidad.

El préstamo o mutuo, el mutuo es un contrato por el cual el mutuante, prestamista o prestador se obliga a entregar al mutuatario, mutuario o prestario una cantidad de dinero o bienes consumibles a cambio de que se le devuelvan otros de la misma especie, calidad o cantidad. En el préstamo bancario, el cliente recibe del Banco (en un solo acto) una determinada cantidad de dinero con el compromiso de devolver su importe más los intereses y comisiones convenidos en las fechas pactadas (Betancourt, 2009).

- **Contactibilidad:**

  Es un proceso que consiste en cómo el banco logra contactar con sus clientes para ofrecerles los productos que crea convenientes, por ejemplo, el canal de telemarketing (Tello, 2017).

- **Ensamble:**

  Dietterich, (2000) el ensamble es una técnica de combinación de dos o más algoritmos sean similares o diferentes llamados aprendices de base. Esto se hace generalmente, para hacer un sistema más robusto que incorpore las predicciones de todos los aprendices base.

  Caruana et al. (2004) define ensemble como una colección de modelos cuyas predicciones se combinan mediante promedios ponderados o votaciones.

  Según Kutmar et al. (2021) los modelos Ensemble ML son metaalgoritmos que crean un grupo de varios enfoques ML y los combinan de manera inteligente en un modelo predictivo para reducir la varianza y el sesgo. Se probaron una serie de algoritmos, incluidos el refuerzo adaptativo, el refuerzo categórico, el refuerzo extremo, el refuerzo ligero, el bosque aleatorio y los árboles adicionales, para encontrar sus tasas de detección y falsos positivos. Se empleó un método de preprocesamiento de datos para mejorar el rendimiento de detección.

- **Overffiting**:

  Overffiting es el efecto de sobreentrenar un algoritmo de aprendizaje con unos ciertos datos, es decir se entrena demasiado el conjunto de datos que ocasiona que el algoritmo de aprendizaje pueda quedar ajustado a unas características muy específicas de los datos de entrenamiento que no tienen relación causal con la función objetivo. Entonces durante la fase de sobreajuste el éxito al responder a las muestras de entrenamiento sigue incrementándose mientras que su actuación con muestras nuevas va empeorando (Hawkins, 2003).

III. MATERIALES Y MÉTODOS

3.1. Lugar de ejecución

La investigación se llevó a cabo en la Universidad Nacional Agraria La Molina (UNALM).

3.2. Materiales y Métodos

3.2.1. Materiales

Los materiales y equipos utilizados en la presente tesis son los siguientes:

- Una computadora marca Toshiba con un procesador Intel Core (TM) I7 @2.50 GHZ, con una memoria RAM DE 4.00 GB y un sistema operativo Windows 10 pro de 64 bits.
- Software R versión 3.6.
- Datos: La base de datos que se empleo está conformada por los clientes que tienen campaña en los meses diciembre 2016, enero 2017 y febrero 2017.
- Usb de16 gb.
- Hojas bond 1000.
- Internet.

3.2.2. Población

La población está conformada por 28 080 personas que formaron parte de la campaña "monto ofrecido al cliente para que desembolse" realizada en los meses de diciembre 2016, enero 2017 y febrero 2017.

3.3. Metodología de la investigación

3.3.1. Tipo de investigación

El Tipo de investigación es aplicada, puesto que el propósito es dar solución a situaciones o problemas concretos e identificables, necesitándose para ello una aplicación innovadora, es decir un método o modelo (Bunge, 1971).

3.3.2. Diseño de la investigación

El diseño de la investigación es no experimental, debido a que no se realizaron manipulaciones de variables, y es de corte transversal, ya que se recolectaron los datos en un solo momento y analiza su incidencia o interrelación en un momento dado (Agudelo et al. 2008).

3.3.3. Formulación de la hipótesis

El método de ensamble Stacking predice con mayor precisión a los clientes potenciales a quienes se les otorgará o desembolsará préstamos en las ofertas de campaña de una entidad financiera, que los algoritmos de aprendizaje supervisado de Machine Learnig: Random Forest, Regresión Logística y Árbol de decisión.

3.3.4. Definición operacional de variables

En la aplicación del método Stacking, se consideró los modelos regresión Logística, Árboles de decisión y Random forest, donde la variable dependiente (Y) es de naturaleza

dicotómica y las variables dependientes fueron extraída en un intervalo de tres campañas de tres meses.

Las variables que se utilizaron en la investigación son:

- Desembolsado: Variable de tipo numérica con medida cualitativa nominal, si el cliente desembolsó (1=Si) y si no desembolsó (0=No).
- Monto de consumo por campaña: Variable de Tipo numérica con medida cuantitativa discreta. Es el Monto que se ofrece al cliente para su posible desembolso.
- Edad: Variable de tipo numérica con medida cuantitativa discreta que registra la edad del cliente.
- Género: Variable de tipo numérica con medida cualitativa nominal que registra el sexo del cliente: F (Femenino) y M (Masculino).
- Estado Civil: Variable de tipo numérica con medida cualitativa nominal que registra el estado civil del cliente: soltero, conviviente, separado, viudo o sin estado civil.
- Número de entidades con las que trabaja el cliente: Variable de tipo numérica con medida cuantitativa discreta que registra el número de entidades con las que trabaja el cliente, mostrándose así su situación financiera.
- Saldo del RCC: Variable de tipo numérica con medida cuantitativa continua que registra en el RCC.
- Saldo de la empresa que debe 1: Variable de tipo numérica con medida cuantitativa continua que registra en el RCC.
- Saldo de la empresa que debe 2: Variable de tipo numérica con medida cuantitativa continua que registra en el RCC.
- Saldo de la empresa que debe 3: Variable de tipo numérica con medida cuantitativa continua que registra en el RCC.
- Nombre del departamento.
- Nombre del distrito.
- Nombre de la provincia.
- Nombre de la agencia.

3.4. Metodología Aplicada

Los pasos que corresponden a la contrastación de hipótesis de la presente investigación son:

- Paso 1: Análisis exploratorio.
- Paso 2: Aplicación del modelo de Regresión Logística.
- Paso 3: Aplicación del modelo de Árbol de Decisión.
- Paso 4: Aplicación del modelo de Random Forest.
- Paso 5: Comparación de los algoritmos.

IV. RESULTADOS Y DISCUSIÓN

4.1. Paso 1: Análisis exploratorio de los datos

4.1.1. Descripción de los datos

La población de este análisis son personas que tuvieron campaña ("monto ofrecido al cliente para que desembolsen") en los meses diciembre 2016, enero 2017 y febrero 2017 en el producto fuerza de ventas de la entidad financiera.

4.1.2. Análisis univariado de los datos

En la tabla 3 se observa las estadísticas descriptivas para las variables cuantitativas de los clientes que están en la base de clientes a quienes se le ofrece una oferta de campaña. En el cuadro se observa que no presentan valores *missing*, por lo que no es necesario realizar ninguna imputación. A la vez se observa que los datos de las variables son asimétricos, la cual se puede asumir que hay sesgo en los datos.

Tabla 3: Análisis univariado de variables cuantitativas

Variable	Valores missing	N	Mìnimo	Maximo	Media	Desv. Tip.	Asimetria
nMonto_Consumo	0	28080	1000	6000.00	3753.78	1628.24	-0.45
Edad	0	28080	1	50.00	20.90	11.48	0.23
SaldoTotalRCC	0	28080	0	468769.22	7058.39	11345.98	0.99
SaldoEmp1	0	28080	0	467059.29	5708.21	9893.81	0.93
SaldoEmp2	0	28080	0	40149.68	1110.85	2393.32	1.39
SaldoEmp3	0	28080	0	20000.00	212.59	792.45	0.80
SaldoEmp4	0	28080	0	5347.08	26.61	198.45	0.40

Fuente: Elaboración propia.

En el Tabla 4 se observa las estadísticas descriptivas para las variables cualitativas de los clientes que están en la base de clientes a quienes se les ofrece una oferta de campaña. Se observa que existe valores missing de 354 datos en las variables departamento, provincia y distrito representando el 1.2% cada uno de ellos. También 6266 registros de la variable Agencias representado el 22.3% y 56 datos en la actividad económica representando el 0.19% del total de datos. Las variables Genero, estado civil y desembolso no tienen valores missing.

Tabla 4: Análisis univariado de variables cualitativas

Nombre de la Variable	Valores Missing	Numero de niveles	Moda	Porcentaje Moda
Departamento	354	24	Lima	40.57
Provincia	354	186	Lima	28.45
Distrito	354	818	ATE	5.40
Agencias	6266	28	Null	22.31
Genero	0	2	M	54.92
Estado Civil	0	5	Sin estado	55.00
Actividad Económica	56	2	Independiente	50.43
Desembolso	0	2	No moroso	91.08

Fuente: Elaboración propia.

- Departamento:

  La variable tiene 24 niveles las cuales son:

  Amazonas, Ancash, Apurímac, Arequipa, Ayacucho, Cajamarca, Callao, Cusco, Huancavelica, Huánuco, Ica, Junín, La Libertad, Lambayeque, Lima, Loreto, Madre de Dios, Moquegua, Pasco, Piura, Puno, San Martín, Tacna, Tumbes, Ucayali
- Provincia: Las provincias son 186.
- Distrito: Los distritos son 354.
- Agencias:

  Las agencias son:

  Chincha, Ceres, Puente piedra, Higuereta, Canto grande, Comas, Chorrillos, Villa el salvador, SJM, Huancayo, Lima, Huaral, Ica, San Borja, La merced, El tambo, Los olivos, Pichanaqui, Huacho, Tarma, Pangoa, Mala, Miraflores, Imperial, San Vicente, Satipo, Zarate y San miguel.

- Género: Masculino y femenino.
- Estado civil: Son 5 niveles: casado, viudo, separado y sin estado.
- Actividad económica: Dos niveles: Dependiente e independiente.

4.1.3. Análisis bivariado de los datos

- Cuadros de clientes que desembolsaron el crédito.

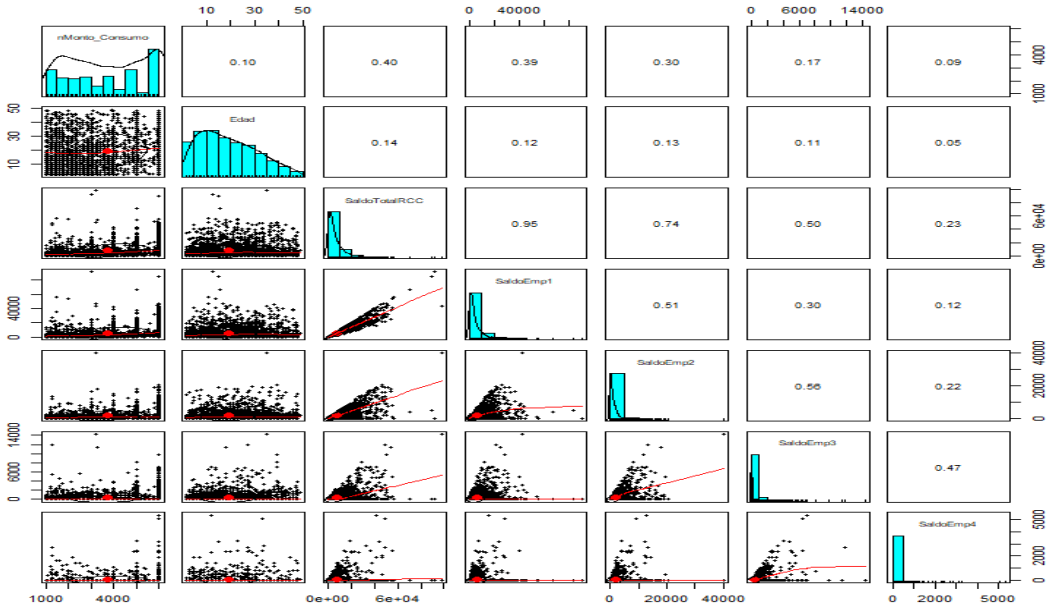


Figura 10: Gráfico de dispersión para clientes que desembolsaron el crédito**.**

Fuente: Elaboración propia.

En la Figura 11 se puede observar las correlaciones entre las variables cuantitativas para los clientes que realizaron desembolsos.

Se observa que las variables que tienen correlación alta son saldo total Rcc vs SaldoEmp1 con una correlación de 0.95, luego Saldo total Rcc vs SaldoEmp1 con una correlación de 0.74; las demás parejas de variables tienen correlaciones bajas menores a 0.52.

- Cuadros de clientes que no desembolsaron el crédito.

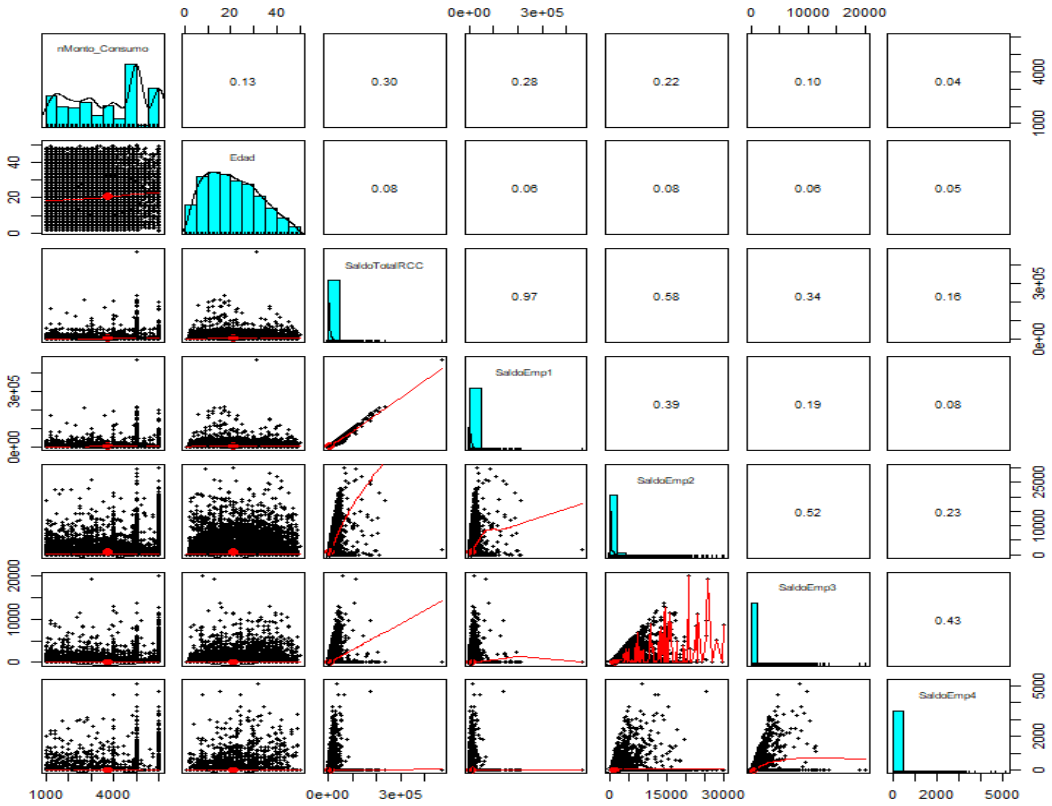


Figura 11: Grafico de dispersión para clientes que no desembolsaron el crédito**.**

Fuente: Elaboración propia.

En la Figura 12 se puede observar las correlaciones entre las variables cuantitativas para los clientes que no realizaron desembolsos.

Se observa que las variables que tienen correlaciones altas son saldo total Rcc vs SaldoEmp1 con 0.97, Saldo total Rcc vs SaldoEmp1 con 0.58; lós demás pares de variables tienes correlaciones bajas menores a 0.55.

4.1.4. Consideraciones en el procedimiento

Para la recopilación de la base de datos se consideró los meses de diciembre del año 2016, enero del 2017 y febrero del 2017 en el producto fuerza de ventas de la entidad financiera. La distribución de la variable respuesta “Desembolso”, es de 8.92% para los que realizaron el desembolso y para los que no desembolsan es de 91.08%. Al tener una de las categorías una cantidad mayor de información, es necesario trabajar alguna técnica de balanceo.

4.1.5. Consideraciones en el procedimiento

Para el desarrollo del modelo se dividió la base de datos: 70% para el tratamiento y 30% para la prueba o testeo.

Tabla 5: Base de datos

Total :28080
Desembolso: 2504
No desembolso: 25576

70%

Entrenamiento
Total :19 656
Desembolso: 1753
No desembolso: 17903

30%

Test
Total :8424
Desembolso: 751
No desembolso: 7673

4.2. Paso 2: Aplicación del modelo de Regresión Logística

4.2.1. Selección de variables

En el modelo de Regresión Logística se aplicó el método de backwards para seleccionar las variables que están más relacionadas con la variable desembolso (Ver Anexo 2).
En la salida de resultados se muestra las variables significativas para el modelo de regresión logística.

Logit[P(Desem=1)]= -6.002e-02 -9.028e-03*Edad-1.008e-05 * SaldoEmp1+2.517e+00 *EstadoCivilCONVIVIENTE+2.908e+00*EstadoCivilSEPARADO-1.882e+01*EstadoCivilSINESTADO+2.099e+00*EstadoCivilSOLTERO+3.004e+00*EstadoCivil VIUDO +3.004e+00* EstadoCivilVIUDO-5.047e-01*GeneroM

```
summary(modelo12)

Call:
glm(formula = Desembolsado ~ Edad + SaldoEmp1 + EstadoCivil +
    Genero, family = binomial() data = smote_sample_train_data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.29509  -0.00009   0.18797   0.62632   1.86886

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)            -6.002e-02  1.916e-01  -0.313  0.75412
Edad                   -9.028e-03  3.592e-03  -2.513  0.01196 *
SaldoEmp1              -1.008e-05  3.792e-06  -2.658  0.00786 **
EstadoCivilCONVIVIENTE  2.517e+00  1.337e-01  18.831  < 2e-16 ***
EstadoCivilSEPARADO     2.908e+00  2.475e-01  11.749  < 2e-16 ***
EstadoCivilSIN ESTADO  -1.882e+01  2.325e+02  -0.081  0.93550
EstadoCivilSOLTERO      2.099e+00  8.916e-02  23.548  < 2e-16 ***
EstadoCivilVIUDO        3.004e+00  4.396e-01   6.833 8.29e-12 ***
GeneroM                -5.047e-01  7.169e-02  -7.041 1.91e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersión parameter for binomial family taken to be 1)

    Null deviance: 9720.7  on 7011  degrees of freedom
```

```
Residual deviance: 4889.6  on 7003  degrees of freedom
AIC: 4907.6

Number of Fisher Scoring iterations: 18
```

4.2.2. Curva ROC

En la Figura 12 la curva ROC para el modelo de Regresión Logística. Se observa que la curva ROC está por encima de la diagonal, más precisamente se encuentra en la parte superior izquierda con un valor de 0.9064, lo que indica que el modelo presenta mayor tasa de verdaderos positivos.

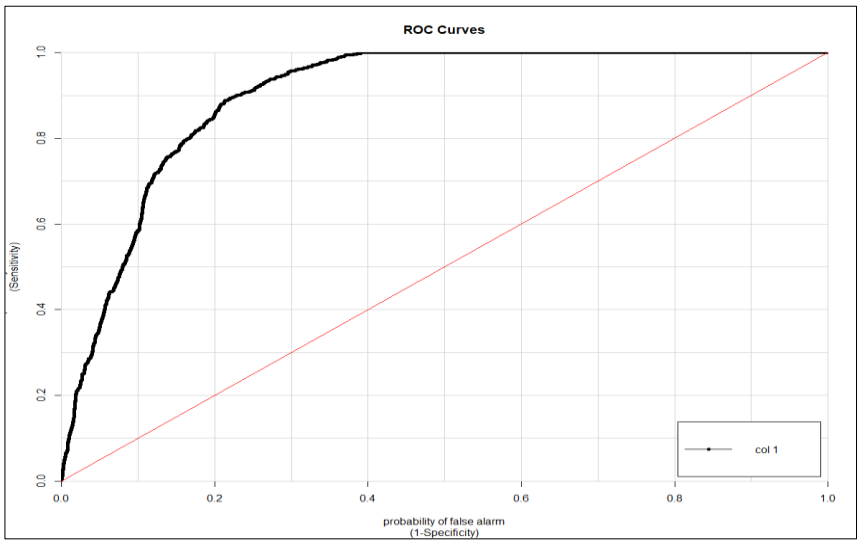


Figura 12: Curva ROC con regresión Logística.

Fuente: Elaboración propia.

4.2.3. Tabla de clasificación e indicadores

En el Tabla 6 se aprecia la matriz de confusión o tabla de clasificación para el modelo de Regresión Logística, donde la precisión es de 79.38 %, la sensibilidad 88.95% y especificidad de 78.44 %.

Tabla 6: Matriz de confusión Regresión Logística

Observado	Pronóstico		
	No	**Si**	**Porcentaje Correcto**
No	6019	1654	78.44%
Si	83	668	88.95%
Porcentaje global			79.38%

Fuente: Elaboración propia.

4.3. Paso 3: Aplicación del modelo de Árbol de Decisión

4.3.1. Selección de variables

La técnica de Árbol de decisión, mediante la librería Rpart en R Projects, tiene el mecanismo incorporado para determinar la importancia de las variables.

4.3.2. Curva ROC

En la Figura 13 la curva ROC para el Árbol de decisión. Se observa que su curva ROC está por encima de la diagonal, más precisamente se encuentra alejada de la esquina izquierda superior con un valor de 0.8943, lo que indica que el modelo presenta una alta tasa de verdaderos positivos y a la vez la tasa de falsos positivos es mayor.

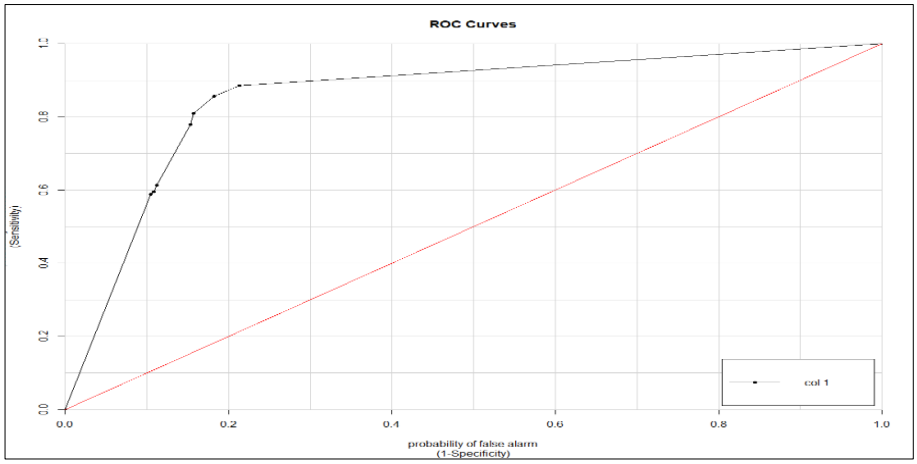


Figura 13: Curva ROC con Árboles de Decisión.

Fuente: Elaboración propia.

4.3.3. Tabla de clasificación e indicadores

En el Tabla 7 se observa la matriz de confusión o tabla de clasificación para el Árbol de Decisión, presentando esta una precisión de 82.9%, sensibilidad de 84% y una especificidad de 82.8%.

Tabla 7: Matriz de confusión Árbol de Decisión

Observado	Pronóstico		Porcentaje Correcto
	No	Si	
No	6356	1317	82.8%
Si	120	631	84.0%
Porcentaje global			82.9%

Fuente: Elaboración propia.

4.4. Paso 4: Aplicación del modelo de Random Forest

4.4.1. Selección de variables

La técnica Random Forest, mediante la librería Random forest en R Project, tiene incorporado el mecanismo para determinar la importancia de las variables según el indicador mean decrease accuracy y mean decrease Gini como se observa en la Figura 14.

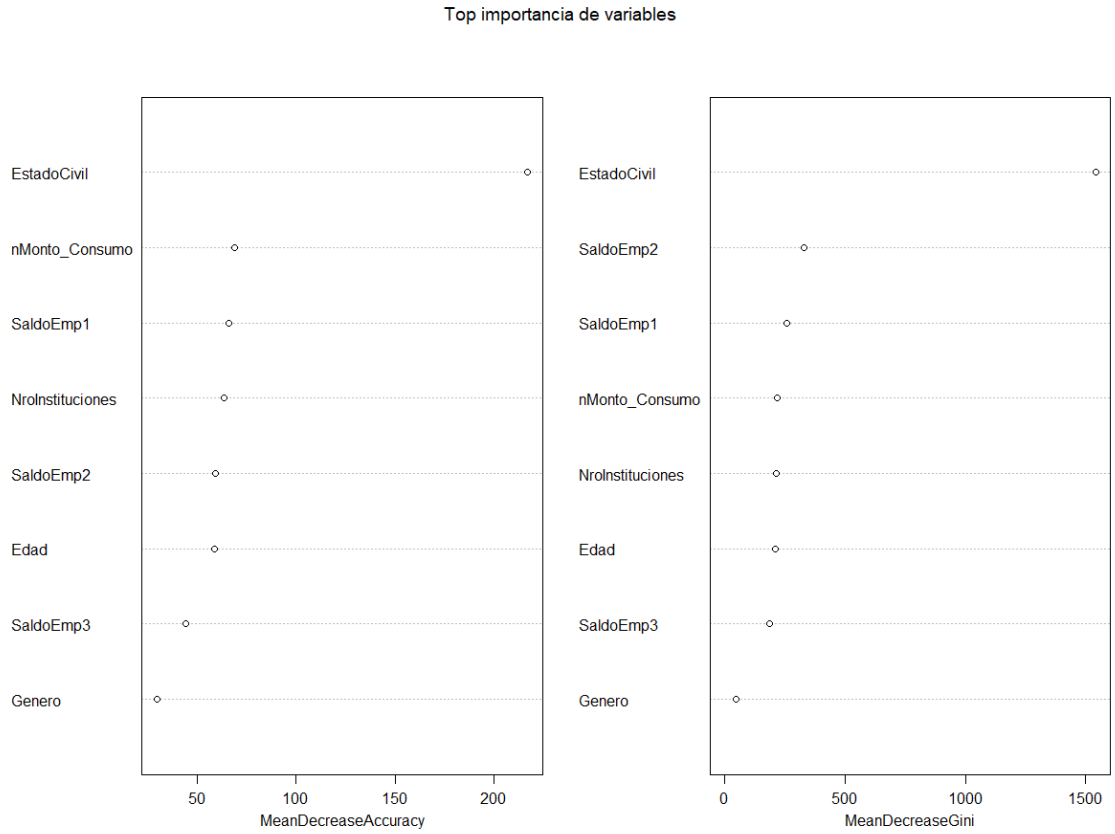


Figura 14: Selección de variables con Random Forest**.**

Fuente: Reporte de R Project.

En la Figura 15, se presenta los errores en la clasificación de las categorías de interés de la variable objetivo, así como el error acumulado.

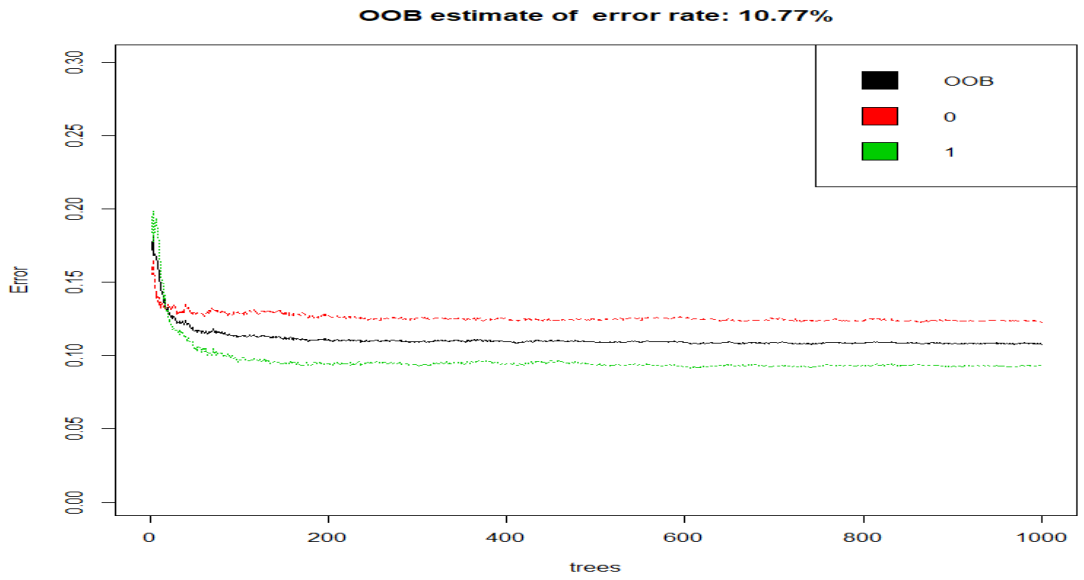


Figura 15**:** OOB con Random Forest.

Fuente: Elaboración propia.

Las líneas de color verde corresponden al OOB de la categoría “desembolso”; el OOB de la categoría “no desembolsado”, está representado por las líneas de color rojo y el OOB del modelo general es de color negro.

Observando la gráfica tenemos que las líneas de los OOB no se encuentran superpuestas, entonces nos indica que tienen el mismo error y que no es necesario identificarlo.

4.4.2. Curva ROC

En la Figura 16, se observa la curva ROC para la técnica Random Forest. Esta se encuentra por encima de la diagonal, pero alejada de la esquina izquierda, mostrando un valor de 0.9074, indicando entonces una alta tasa de verdaderos positivos y a la vez una mayor la tasa de falsos positivos.

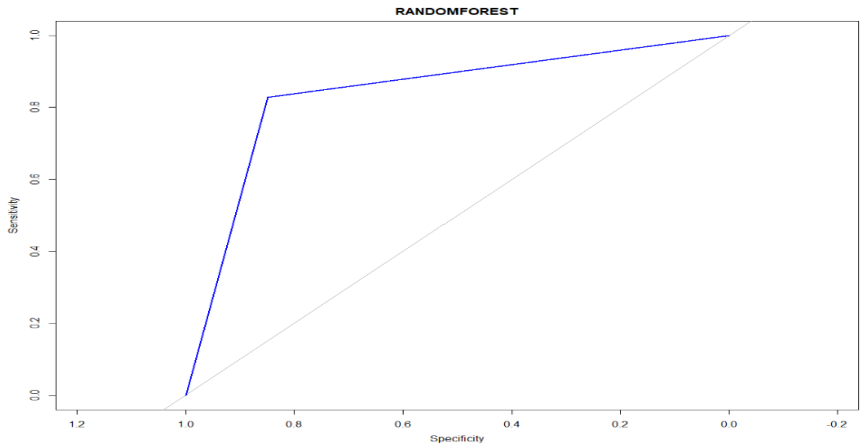


Figura 16: Curva ROC con Random Forest.

Fuente: Elaboración propia.

4.4.3. Tabla de clasificación e indicadores

En la Tabla 8 se observa la matriz de confusión o tabla de clasificación para la el método de ensamble Random Forest, donde la precisión es de 84.6%, la sensibilidad de 82.7% y la especificidad de 84.8%.

Tabla 8: Matriz de confusión Random Forest

Observado	**Pronóstico**		
	No	**Si**	**Porcentaje Correcto**
No	6509	1164	84.8%
Si	130	621	82.7%
Porcentaje global			84.6%

Fuente: Elaboración propia.

4.5. Paso 5: Comparación de los algoritmos

4.5.1. Curva ROC

En la Figura 17 la curva ROC para el método Stacking. La curva se encuentra por encima de la diagonal y alejada de la esquina izquierda, con un área de 0.9117, indicando una alta tasa de verdaderos positivos y a la vez una mayor tasa de falsos positivos.

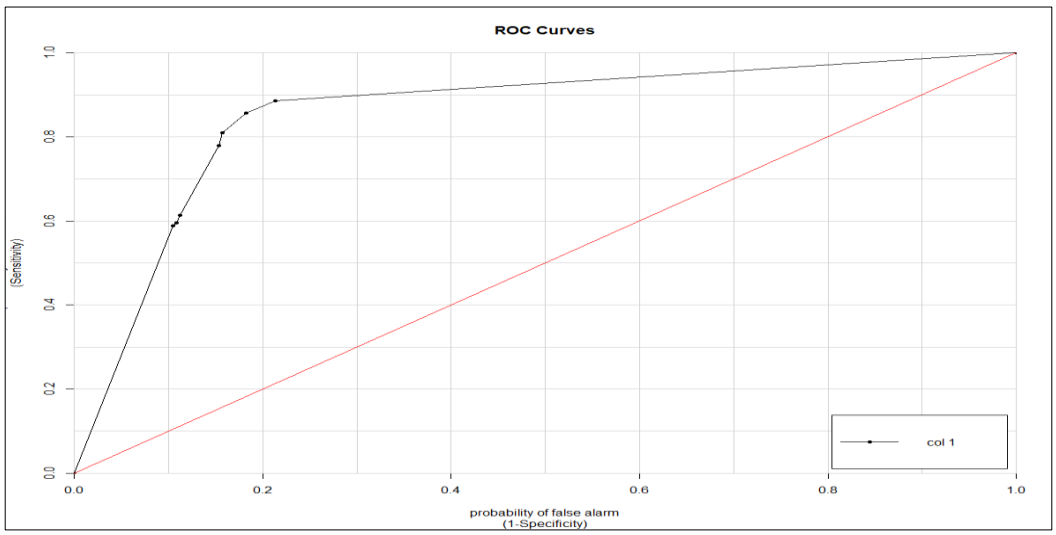


Figura 17: Curva ROC Stacking.

Fuente: Elaboración propia.

4.5.2. Tabla de clasificación e indicadores

En la Tabla 9 se observa la matriz de confusión o tabla de clasificación para el método Stacking donde la precisión es de 82.2%, sensibilidad de 87.9% y especificidad de 81.6%.

Tabla 9: Matriz de confusión Stacking

Observado	Pronóstico		
	No	Si	Porcentaje Correcto
No	6264	1409	81.6%
Si	91	660	87.9%
Porcentaje global			82.2%

Fuente: Elaboración propia.

4.5.3. Comparación de los modelos

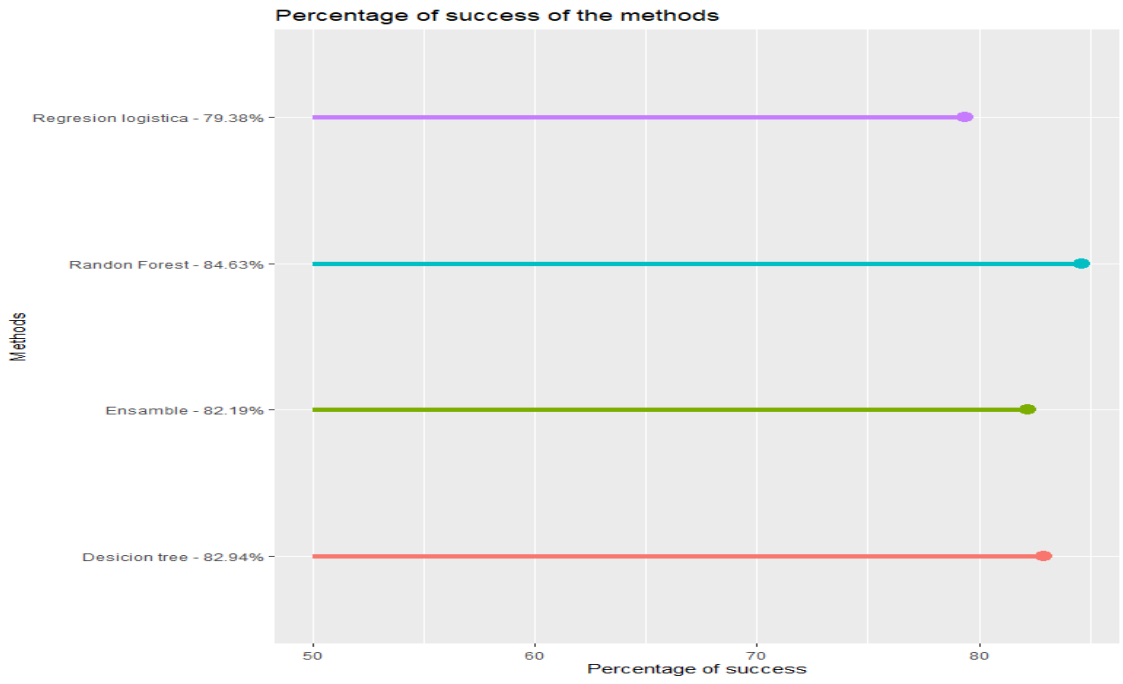


Figura 18: Comparación de clasificación general

Fuente: Reporte de R Project

En la Figura 18, se observa que el modelo de clasificación que tiene mayor accuracy es Random Forest con un porcentaje exitosa de clasificación de 84.63%, seguido del Árbol de Decisión con 82.94%, método Stacking 82.19% y por último Regresión Logística con 79.38%.

En la Tabla 10, se comparan los indicadores de la curva ROC.

Tabla 10: Comparación de sensibilidad y especificidad de la curva ROC

Modelo	Regresión Logística	Árbol de Decisión	Random Forest	Stacking de Modelos
Especificidad	78.4%	82.8%	84.8%	81.6%
Sensibilidad	88.9%	84.0%	82.7%	87.9%

Fuente: Elaboración propia.

Se observa que el modelo de clasificación que tiene mayor sensibilidad es Regresión Logística 88.9%, seguido de método Stacking con 87.9%, luego Árbol de Decisión con 84.0% y por último Random Forest con 82.7%.

Respecto a la especificidad, el modelo de clasificación que tiene mayor especificidad es Random Forest con 84.8%, seguido del Árbol de Decisión 82.8%, luego el método Stacking de modelos con 81.6% y por último la Regresión Logística con 78.4%.

El análisis estadístico con respecto a otras métricas de clasificación tales como Kolmogorov, AUC, Gini y Logloss, se observa en la Tabla 11.

Tabla 11: Comparación de los indicadores

	Regresión Logística	Árbol de Decisión	Random Forest	Stacking de Modelos
Kolmogorv	0.6751	0.6907	0.7028	0.7124
Auc	0.9064	0.8943	0.9074	0.9117
Gini	0.8128	0.7885	0.8148	0.8235
Logloss	0.3959	0.3886	0.3435	0.3177

Fuente: Elaboración propia.

En la Tabla 11, se observa que para los resultados de la métrica Kolmogorov, el que tiene mayor indicador es el método Stacking con un valor de 0.7124, seguido por Random Forest con 0.7028, Árbol de Decisión 0.6907 y por último la Regresión Logística con 0.6751.

En cuanto al Auc, el modelo de clasificación que tiene mayor valor de AUC es el método Stacking con 0.9117, seguido de Random Forest con 0.9074, Regresión Logística de 0.9064 y finalmente Árbol de Decisión con 0.8943.

Con respecto al indicador Gini, el modelo de clasificación que presenta mayor valor es el método Stacking con 0.8235, seguido de Random Forest con 0.8148, Regresión Logística 0.8128 y finalmente Árbol de Decisión con 0.7885.

Y por último con respecto al indicador Logloss, el modelo de clasificación que indica un mayor desempeño es el método Stacking con 0.3177, seguido de Random Forest con 0.3435, Árbol de Decisión 0.3886 y finalmente el algoritmo de Regresión Logística con 0.3959.

4.6. Proceso computacional

En el proceso computacional de ejecución de los modelos de Regresión Logística, Árbol de decisión y Random Forest; se observó que el modelo que presenta mayor tiempo de ejecución es Random Forest en comparación a los otros modelos.

En Random Forest se observa que mientras mayor sea el número de árboles, mayor será el tiempo de ejecución. Para nuestro caso, se generó 1000 árboles y la demora fue de 30 minutos.

El tiempo de ejecución de los otros modelos fue menor a 10 minutos y en cuanto al método Stacking su procesamiento su tiempo de ejecución fue de 5 minutos.

V. CONCLUSIONES

- El método de ensamble Stacking predice con mayor precisión a los clientes potenciales a quienes se les otorgará o desembolsará préstamos en las ofertas de campaña de una entidad financiera, que los algoritmos de aprendizaje supervisado de Machine Learning: Random Forest, Regresión Logística y Árbol de Decisión.

- Se logró identificar a los clientes potenciales para campañas de una entidad financiera usando el método Stacking con los algoritmos de Regresión logística, Árbol de Decisión y Random Forest. Cuyos resultados fueron :87.9%, 88.9%, 84.0% y 82.7% respectivamente.

- Con respecto a la comparación del método Stacking con los algoritmos de Regresión logística, Árbol de Decisión y Random Forest mediante los indicadores Auc, Gini, Logloss, Kolmogorov. Se obtuvo el mejor desempeño con el método Stacking, los resultados fueron 0.9117, 0.8235, 0.3177 y 0.7124 respectivamente.

- Computacionalmente la Regresión Logística procesa en menos tiempo, seguido de árbol de decisión. Mientras, que en la técnica de Random Forest el tiempo de procesamiento depende de la cantidad de árboles que se genere, es decir a mayor número de árboles el tiempo de ejecución es mayor. En cambio, en el método ensamble Stacking su procesamiento demora segundos, tiempo mucho menor en comparación de los métodos individuales.

VI. RECOMENDACIONES

- Se recomienda utilizar en futuras investigaciones otros indicadores de comparación como F1 Score, Cohen's Kapa, entre otros.

- Utilizar otros algoritmos tales como: KNN, Análisis de soporte vectorial, entre otros en el desarrollo del método Stacking.

- Comparar los resultados obtenidos en la presente tesis con otros algoritmos no estudiados.

- En la implementación del mejor modelo obtenido en la presente tesis se recomienda crear grupos de perfiles de clientes mediante la segmentación score.

VII. BIBLIOGRAFÍA

ABRAIRA V Y PEREZ A 1996. Métodos Multivariantes en Bioestadística.

AGUDELO, G; AIGNEREN, M.; RUIZ, J. 2008. Diseños de investigación experimental y no-experimental. Centros de estudios de opinión.

ALVARADO, JUAN 2003. Algoritmos de minería de datos.

BELTRÁN, M.; MUÑOZ. A.; MUÑOZ, M. 2012. Un nuevo clasificador de préstamos bancarios a través de la minería de datos. Disponible en: https://www2.uned.es/dpto-economia-aplicada-y-estadistica/SEIO2012.pdf. Consultado el 13 de Mayo del 2018.

BETANCOURT, C. 2009. "Las Operaciones Bancarias Activas en el Perú". Disponible: http://www2.congreso.gob.pe/sicr/cendocbib/con4_uibd.nsf/05EDEE22BF2868E005257A940076FB5B/$FILE/contratos_bancarios.pdf.

BEUNZA, J.; PUERTAS, E.; CONDES, E . 2019. Manual práctico de inteligencia artificial en entornos sanitarios. Disponible en: https://books.google.es/books?hl=es&lr=&id=88nSDwAAQBAJ&oi=fnd&pg=PA35&dq=algoritmo+supervisado&ots=6Q7fLHReSW&sig=guh9K1O7dQi3pAn9s4vZmQgkhOM#v=onepage&q=algoritmo%20supervisado&f=false

BORUEL, M. 2012. Métodos de agregación de modelos y aplicaciones. Memoria de Trabajos de Difusión Científica y Técnica, núm. 10 (2012). Consultado el 12 de julio del 2018.

BOUZA, C Y SANTIAGO, A. 2012. La minería de datos: árboles de decisión y su aplicación en estudios médicos. Universidad de la habana. Cuba. Universidad Autónoma de Guerrero, México. Consultado el 29 de Julio del 2020.

BUJA, A.; STUETZLE, W.; SHEN, Y. 2005. Loss Functions for Binary Class Probability Estimation and Classification: Structure and Applications.

BUNGE, M.1996. La investigación científica. Ed Ariel. Barcelona. 1996. Epistemología. Ed. Ariel. Barcelona. 1981.

BREIMAN, L.; FRIEDMAN, J.; OLSHEN, R.; STONE, C. G. 1984. Classification and Regression Trees. Wadsworth International Group, Belmont, California, USA.

BREIMAN, L. 1992. Stacked Regression. Statistics Department, University of California, Berkeley, CA 94720. Machine Learning, 24, 49-64 (1996) Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.. Consultado el 10 de Junio del 2018.

BREIMAN, L. 2001. Random Forests.

BROWN,E. 2016. Machine learning with Random Forest and Decisiones Trees.

CAMPO, Y. Y CRUZ, C. 2017. Modelos Apilados y factores que pueden afectar la eficiencia. Universidad Santo Tomas Facultad de Estadística. Consultado el 15 de julio del 2018.

CAFFE, M; SANTORO, P; BARANAUSKAS, J. 2011. Avaliação do Algoritmo de Stacking em Dados Biomédicos. Consultado el 27 de Octubre del 2020.

CARUANA, R.; NICULESCU,A; CREW,G.; KSIKES,A. 2011 Ensemble Selection from Libraries of Models . Consultado el 07 de agosto del 2020.

CARRANZA, R. 2019. Reconocimiento de caracteres en imágenes no estructuradas.

CLARKE, B. 2003. Comparing Bayes Model Averaging and Stacking When Model Approximation Error Cannot be Ignored. Department of Statistics University of British

Columbia Vancouver, BC V6T 1Z2, Canada.Journal of Machine Learning Research 4 (2003) 683-71. Consultado el 2 de agosto del 2018.

CUTLER, A.; CUTLER, D; STEVENS, J. 2012 Random forest. Consultado el 07 de agosto del 2020.

DE'ATH, G Y FABRICIUS, K. E. 2000. Classification and regression trees: A powerful yet simple technique for ecological data analysis. Ecology, 81 (11) 3178–3192.

DECONINCK, E., ZHANG, M. H., COOMANS, D., & HEYDEN, Y. V. 2006. Classification tree models for the prediction of blood-brain barrier passage of drugs. Journal of Quemical Information and Modeling, , 46 (3) 1410–1419.

DIAZ, H.; ALEMAN, Y.; CABRERA, L.; MORALES, A.; CHAVEZ, M Y CASAS, G.2015. Algoritmos de aprendizaje automático para clasificación de Splice Sites en secuencias genómicas Machine Learning algorithms for Splice Sites classification in genomic sequences. Editorial "Ediciones Futuro". Revista Cubana de Ciencias Informáticas. Vol. 9, No. 4, octubre-diciembre, 2015. Pág. 155-170. Consultado el 13 de Julio del 2018.

DIAZ, J. 2012. Comparación entre Árboles de Regresión CART y Regresión Lineal. Consultado EL 22 agosto 2020.

DIETTERICH, T. 2000. Ensemble Methods in Machine Learning

DOBRA, A. 2002. Classification and regression tree construction. Thesis proposal, Departament of Computer Science, Cornell University, Ithaca NY. Consultado el 20 abril 2021.https://www.cise.ufl.edu/~adobra/papers/a-exam.pdf

DOOB, J. 2016. HEURISTIC APPROACH TO THE KOLMOGOROV-SMIRNOV THEOREMS'. Consultado el 10 de agosto del 2020.

ESPINAR, R. 2018. Modelos de Clasificación con datos no balanceados. Consultado el 12 de agosto del 2020.

FAN, J.; UPADHYE, S.; WORSTER, A. 2006. Understanding receiver operating characteristic (ROC) curves.

FIUZA, D Y RODRÍGUEZ, J. 2000. La Regresión Logística es una herramienta versátil. Consultado el 20 abril 2019.https://www.revistanefrologia.com/es-la-regresion-logistica-una-herramienta-versatil-articulo-X0211699500035664.

FAWCETT, T. 2016. Learning from Imbalanced Classes. Consultado el 03 abril 2020. https://www.svds.com/learning-imbalanced-classes/

GISLASON, P.; BENEDIKTSSON, J.; SVEINSSON, J. 2005. Random Forests for land cover classification. Consultado el 05 de agosto del 2020.

GUERRA, L. 2017.Prediccion de fuga de clientes en una corredora de seguros utilizando Regresión Logística y el algoritmo Random Forest. Consultado el 13 de agosto del 2020.

GUPTA, P. 2017. Cross-Validation in Machine Learning. Consultado el 12 de agosto del 2020. Disponible en: https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f.

HAWKINS, D. 2003. The problema overffiting.

Heller, M.; 2019. What is machine learning? Intelligence derived from data. Consultado el 16 de agosto del 2020. Disponible en: https://www.infoworld.com/article/3214424/what-is-machine-learning-intelligence-derived-from-data.html.

JIMÉNEZ, A. 2012. Insights into the area under the receiver operating characteristic curve (AUC) as a discrimination measure in species distribution modellinggeb_683 498

JOAQUIN, R. 2017.Árboles de predicción: Random forest ,gradient boosting y c5.0. Consultado el 30 de Julio del 2020. Disponible en: https://www.cienciadedatos.net/documentos/33_árboles_de_prediccion_bagging_random_forest_boosting#C50

KOTSIANTIS, S.; KOUMANAKOS, E.; TZELEPIS, D.; TAMPAKAS, V. 2007. Forecasting Fraudulent Financial Statements using Data Mining.

KUTMAR,S Y SARKAR,D. 2021. Ensemble machine learning models for the detection of energy theft. Disponible en: https://www.sciencedirect.com/science/article/abs/pii/S0378779620307021.

LARRAÑAGA, P.; IÑAKI, Y.; MOUJAID, A. 2005. Regresión Logística. Departamento de Ciencias de la Computación e Inteligencia Artificial. Consultado el 15 de agosto del 2020.

LOPEZ,R.2015 . La minería de datos, entre la estadística y la inteligencia artificial

Disponible en: https://comunidad.iebschool.com/bigdata/2015/05/13/la-mineria-de-datos-entre-la-estadistica-y-la-inteligencia-artificial/.

LOH,W. 2011. Classification and regression tres.

LLOPIS, J 2014. La Estadística: Una Orquesta Hecha Instrumento. Consultado el 15 de agosto del 2020. Disponible en: https://jllopisperez.com/2013/12/19/test-de-hosmer-y-lemeshow/.

MARIÑAS, G.2009. Evaluación de algoritmos supervisados de extracción de características para clasificación de texturas

MEASE, D Y WYNER, A. 2008. Evidence Contrary to the Statistical View of Boosting. Journal of Machine Learning Research 9 pp.

MORENO, J.; RODRIGUEZ, D.; SICILIA, M.; RIQUELME, J; RUIZ, R. 2009. SMOTE-I: mejora del algoritmo SMOTE para balanceo de clases minoritarias. Consultado el 11 de agosto del 2020.

MORENO, A.; VICENTE, P.; GALINDO, P. 2016. Aprendizaje basado en árboles de decisión: un estudio crítico desde Weka, RapidMiner y SPSS Modele. Consultado el 29 de Julio del 2020.

NIETO, A 2015. Estimación de la probabilidad de egreso de estudiantes de licenciatura en ciencias de la BUAP usando Regresión Logística. Consultado el 13 de agosto del 2020.

PADMAPANI, P.; SUNIL, G.; RATNADEEP, R. 2018. Stacking ensemble model for polarity classification in feature based opinion mining. Indian Journal of Computer Science and Engineering (IJCSE). Consultado el 13 de Marzo del 2018.

PEDROSA,J. 2016.Economipedia.

PORTUGAL, R Y CARRASCO, M. 2006. ensamble de algoritmos bayesianos con árboles de decisión, una alternativa de clasificación.

RAMÍREZ, W Y RODRIGUEZ, Y. 2014. La Regresión Logística aplicada a un programa de salud en Medicina Veterinaria - The logistic regression applied to a health program in Veterinary Medicine. Consultado el 14 de agosto del 2020.

ROME ,1999. I 2009 CRMI SI2008.Memoria-analisis

SALAS, M. 1996. La regresión logística. Una aplicación a la demanda de estudios universitarios. Consultado el 15 de agosto del 2020.

SERNA, S. 2009. Comparación de Árboles de Regresión y Clasificación y regresión logística. Escuela de Estadística Facultad de Ciencias Universidad Nacional de Colombia Sede Medellín. Consultado el 15 de agosto del 2020.

SEGRERA, S Y MORENO, M. 2006. Multiclasificadores: Métodos y arquitecturas. Informe Técnico – Technical Report DPTOIA-IT-2006-001 Marzo, 2006. Departamento de Informática y Automática Universidad de Salamanca. Consultado el 13 de Mayo del 2018.

SILVA, B. y MOLINA, M. 2016. Likelihood ratio (razón de verosimilitud): definición y aplicación en Radiología. Consultado el 14 de agosto del 2020.

SIMEONE, O 2018. A Very Brief Introduction to Machine Learning With Applications to Communication Systems. Consultado el 05 de agosto del 2021.https://empresas.blogthinkbig.com/que-algoritmo-elegir-en-ml-aprendizaje/

SOLARTE, G Y SOTO, J. 2011. Árboles de decisiones en el diagnóstico de enfermedades cardiovasculares. Universidad tecnológica de Pereira. Colombia. Consultado el 29 de Julio del 2020.

STROB, C.; BOULESTEIX, A.; KNEIB, T.; THOMAS, A; ACHIM, Z. 2008. Conditional variable importance for random forests. Consultado el 05 de agosto del 2020. https://link.springer.com/article/10.1186/1471-2105-9-307

TELLO, L. 2017. Analisis y mejora de la Contactibilidad a traves de los procesos de telemarketing en una entidad bancaria. Consultado el 27 de abril del 2020.

TIMOFEEV, R. 2004. Classification and regression trees (cart). theory and applications. Master thesis, CASE - Center of Applied Statistics and Economics. Humboldt University, Berlin.

MITCHEL, M. 2011. Bias of the Random Forest Out-of-Bag (OOB) Error for Certain Input Parameters. Consultado el 05 de agosto del 2020. https://www.scirp.org/html/9-1240025_8072.htm

VAQUERIZO, R. 2011. Medir la importancia de las variables con Random Forest. Consultado el 07 de agosto del 2020. https://analisisydecision.es/medir-la-importancia-de-las-variables-con-random-forest/.

VILLAMARINO, G. 2015. Metodología de minería de datos para el estudio de tablas de siniestralidad víal. Facultad de estudios estadísticos. Universidad Complutense de Madrid. Consultado el 23 de junio del 2018.

WAH, B.; KHATIJAHHUSNA, A.; HEZLIN, A.; SIMON, F; ZURAIDA, K.; NIK, A. 2014. An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets. Consultado el 11 de agosto del 2020.

WIDMANN, M. 2019. From Modeling to Scoring: Confusión Matrix and Class Statistics. Consultado el 10 de agosto del 2020. https://www.knime.com/blog/from-modeling-to-scoring-confusion-matrix-and-class-statistics

VIII. ANEXOS

ANEXO 1: Análisis previos del modelo.

```
data<-
read.csv('E:/2020/TESIS_PREGRADO_UNALM/DATA_TESIS/DATA_DESEMBOLSO_N.csv',
header=T)
str(data)
data[,"Desembolsado"]<-as.factor(data[,"Desembolsado"])
data<-data[,-c(1:4)]
a<-subset(data,Genero=="F")
b<-subset(data,Genero=="M")
dim(a)
dim(b)
c<-subset(data,Genero==" ")
d<-subset(data,Genero=="NULL")
dim(c)
dim(d)
datos<-subset(data,Genero=="F"| Genero=="M")
str(datos)

row.names(datos)<-1:dim(datos)[1]
datos[,"Genero"]<-as.character(datos[,"Genero"])
table(datos[,"Desembolsado"])
datos1<-subset(datos,Desembolsado==0)
datos2<-subset(datos,Desembolsado==1)
dim(datos1)
dim(datos2)



#datos3<-
datos[c(sample(as.numeric(row.names(datos1))2504,F)as.numeric(row.names(d
atos2)))]
```

```
datos[,"Genero"]<-as.factor(datos[,"Genero"])
datos[,"Edad"]<-as.numeric(as.character(datos[,"Edad"]))
attach(datos)
str(datos)
# NomDepart NomDistr "NomProvin","Emp1","Emp2","Emp3","Emp4")]
#datos3<-datos3[,-c(7,8,9,20,21,22,23)]
#datooo<-subset(datos,select=-c("NomDepart","NomDistr",
#
"NomProvin","Agencia","Emp1","Emp2","Emp3","Emp4"
#                               ))

datos<-datos[,-c(7,8,9,10,20,21,22,23)]
datos[,"EstadoCivil"]<-as.character(datos[,"EstadoCivil"])

row.names(datos)<-1:dim(datos)[1]
head(datos$EstadoCivil)
datos[row.names(subset(datos,EstadoCivil=="Soltero"))"EstadoCivil"]<-
"SOLTERO"
datos[row.names(subset(datos,EstadoCivil=="Casado"))"EstadoCivil"]<-
"CASADO"
datos[row.names(subset(datos,EstadoCivil=="Viudo"))"EstadoCivil"]<-
"VIUDO"
datos[row.names(subset(datos,EstadoCivil=="Divorciado"))"EstadoCivil"]<-
"SEPARADO"
datos[row.names(subset(datos,EstadoCivil=="DIVORCIADO"))"EstadoCivil"]<-
"SEPARADO"
datos[row.names(subset(datos,EstadoCivil=="Separado"))"EstadoCivil"]<-
"SEPARADO"
datos[row.names(subset(datos,EstadoCivil=="Sin
informacion"))"EstadoCivil"]<-"SIN ESTADO"
datos[row.names(subset(datos,EstadoCivil=="NULL"))"EstadoCivil"]<-"SIN
ESTADO"


datos[,"EstadoCivil"]<-as.factor(datos[,"EstadoCivil"])
#datos3[,"TieneMonto"]<-as.factor(datos3[,"TieneMonto"])
#datos3[,"DesembolsoRefinanciado"]<-
as.factor(datos3[,"DesembolsoRefinanciado"])

str(datos)
a<-table(datos$Desembolsado)
```

```
round(prop.table(a)4)*100

# 0     1
# 25576  2504
# 0          1
# 91.08  8.92
library(mlr)
summarizeColumns(datos)

###:::::::particion
library(caTools)
set.seed(123)
split = sample.split(datos$Desembolsado, SplitRatio =0.7)
data.train = subset(datos, split == TRUE)
data.test = subset(datos, split == FALSE)

c<-table(data.train$Desembolsado)
# 0    1
# 17903  1753
round(prop.table(c)4)*100
# 0     1
# 91.08  8.92
d<-table(data.test$Desembolsado)
# 0    1
# 7673  751
round(prop.table(d)4)*100
# 0     1
#91.08  8.92

##:: balanceo
library(DMwR)
smote_sample_train_data <- SMOTE(Desembolsado ~ ., data = data.train,
perc.over = 100, perc.under=200)
print('Number of transactions in train dataset after applying SMOTE
sampling method')
print(table(smote_sample_train_data$Desembolsado))
# 0    1
# 3506 3506
library(caret)
#validacion cruzada
ctrl<- trainControl(
```

```
method = "repeatedcv",
number = 10,
repeats = 10)
```

ANEXO 2: Análisis de la Regresión Logística

- Selección de variables

Modelo 1:

```
f1<c("nMonto_Consumo","Edad","NroInstituciones","SaldoTotalRCC","SaldoEmp
1","SaldoEmp2","SaldoEmp3","EstadoCivil","Genero")
Desembolsado<-'Desembolsado'
modelo1<train(smote_sample_train_data[,f1],smote_sample_train_data[,Desem
bolsado],method='glm',trControl=ctrl,tuneLength=10,family = binomial())
summary(modelo1)

Call:
NULL

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.56331  -0.00010   0.08265   0.61627   2.32013

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              3.561e-01  2.232e-01   1.596 0.110558
nMonto_Consumo          -2.197e-04  2.385e-05  -9.211  < 2e-16 ***
Edad                    -1.096e-02  3.656e-03  -2.996 0.002733 **
NroInstituciones         1.571e-01  4.165e-02   3.770 0.000163 ***
SaldoTotalRCC            2.412e-04  1.852e-04   1.303 0.192667
SaldoEmp1               -2.575e-04  1.852e-04  -1.391 0.164377
SaldoEmp2               -1.022e-04  1.849e-04  -0.552 0.580655
SaldoEmp3               -3.416e-04  2.129e-04  -1.604 0.108649
EstadoCivilCONVIVIENTE   2.580e+00  1.369e-01  18.848  < 2e-16 ***
EstadoCivilSEPARADO      2.963e+00  2.510e-01  11.808  < 2e-16 ***
EstadoCivilSIN ESTADO   -1.884e+01  2.284e+02  -0.082 0.934260
EstadoCivilSOLTERO       2.076e+00  9.140e-02  22.709  < 2e-16 ***
EstadoCivilVIUDO         3.021e+00  4.450e-01   6.789 1.13e-11 ***
GeneroM                 -4.827e-01  7.301e-02  -6.612 3.80e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4733.9  on 6998  degrees of freedom
AIC: 4761.9

Number of Fisher Scoring iterations: 18
```

Modelo 2:

```
f2<c("nMonto_Consumo","Edad","SaldoTotalRCC","SaldoEmp1","SaldoEmp2","Sal
doEmp3","EstadoCivil","Genero")  #,"NroInstituciones"
Desembolsado<-'Desembolsado'
modelo2<train(smote_sample_train_data[,f2],smote_sample_train_data[,Desem
bolsado],method='glm',trControl=ctrl,tuneLength=10,family = binomial())
summary(modelo2)

Call:
NULL

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.66492  -0.00010   0.07149   0.62691   2.26355

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              6.365e-01  2.105e-01   3.023  0.00250 **
nMonto_Consumo          -2.147e-04  2.378e-05  -9.028  < 2e-16 ***
Edad                    -1.097e-02  3.654e-03  -3.002  0.00269 **
SaldoTotalRCC            3.224e-04  1.904e-04   1.693  0.09044 .
SaldoEmp1               -3.388e-04  1.904e-04  -1.779  0.07520 .
SaldoEmp2               -1.631e-04  1.906e-04  -0.856  0.39216
SaldoEmp3               -3.831e-04  2.196e-04  -1.744  0.08112 .
EstadoCivilCONVIVIENTE   2.558e+00  1.364e-01  18.761  < 2e-16 ***
EstadoCivilSEPARADO      2.973e+00  2.507e-01  11.862  < 2e-16 ***
EstadoCivilSIN ESTADO   -1.887e+01  2.288e+02  -0.082  0.93427
EstadoCivilSOLTERO       2.077e+00  9.126e-02  22.763  < 2e-16 ***
EstadoCivilVIUDO         3.044e+00  4.440e-01   6.857 7.05e-12 ***
GeneroM                 -4.894e-01  7.287e-02  -6.716 1.87e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4748.2  on 6999  degrees of freedom
AIC: 4774.2

Number of Fisher Scoring iterations: 18
```

Modelo 3:

```
f3<-
c("nMonto_Consumo","Edad","SaldoTotalRCC","SaldoEmp1","SaldoEmp3","Estado
Civil","Genero")  #,"NroInstituciones" ,"SaldoEmp2"
Desembolsado<-'Desembolsado'
modelo3<-
train(smote_sample_train_data[,f3],smote_sample_train_data[,Desembolsado]
,method='glm',trControl=ctrl,tuneLength=10,family = binomial())
summary(modelo3)

Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6426  -0.0001   0.0799   0.6267   2.2647

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)             6.347e-01  2.105e-01   3.015  0.00257 **
nMonto_Consumo         -2.141e-04  2.376e-05  -9.011  < 2e-16 ***
Edad                   -1.095e-02  3.654e-03  -2.997  0.00272 **
SaldoTotalRCC           1.606e-04  2.132e-05   7.533 4.97e-14 ***
SaldoEmp1              -1.772e-04  2.295e-05  -7.721 1.16e-14 ***
SaldoEmp3              -2.033e-04  6.636e-05  -3.063  0.00219 **
EstadoCivilCONVIVIENTE  2.555e+00  1.363e-01  18.746  < 2e-16 ***
EstadoCivilSEPARADO     2.969e+00  2.506e-01  11.848  < 2e-16 ***
EstadoCivilSIN ESTADO  -1.887e+01  2.288e+02  -0.082  0.93429
EstadoCivilSOLTERO      2.075e+00  9.116e-02  22.757  < 2e-16 ***
EstadoCivilVIUDO        3.044e+00  4.441e-01   6.855 7.15e-12 ***
GeneroM                -4.883e-01  7.286e-02  -6.702 2.05e-11 ***
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4749.0  on 7000  degrees of freedom
AIC: 4773

Number of Fisher Scoring iterations: 18
```

Modelo4:

```
f4<-
c("nMonto_Consumo","Edad","SaldoTotalRCC","SaldoEmp1","EstadoCivil","Gene
ro")  #,"NroInstituciones" ,"SaldoEmp2" ,"SaldoEmp3"
Desembolsado<-'Desembolsado'
modelo4<-
train(smote_sample_train_data[,f4],smote_sample_train_data[,Desembolsado]
,method='glm',trControl=ctrl,tuneLength=10,family = binomial())
summary(modelo4)

Call:
NULL

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.95511  -0.00010   0.07109   0.62833   2.24193

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)             6.200e-01  2.103e-01   2.949  0.00319 **
nMonto_Consumo         -2.058e-04  2.360e-05  -8.721  < 2e-16 ***
Edad                   -1.104e-02  3.651e-03  -3.024  0.00249 **
SaldoTotalRCC           1.157e-04  1.430e-05   8.086 6.18e-16 ***
SaldoEmp1              -1.315e-04  1.642e-05  -8.009 1.16e-15 ***
EstadoCivilCONVIVIENTE  2.552e+00  1.361e-01  18.745  < 2e-16 ***
EstadoCivilSEPARADO     2.949e+00  2.502e-01  11.786  < 2e-16 ***
EstadoCivilSIN ESTADO  -1.885e+01  2.293e+02  -0.082  0.93449
EstadoCivilSOLTERO      2.071e+00  9.102e-02  22.753  < 2e-16 ***
EstadoCivilVIUDO        3.035e+00  4.437e-01   6.841 7.88e-12 ***
GeneroM                -4.911e-01  7.279e-02  -6.747 1.51e-11 ***
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4758.4  on 7001  degrees of freedom
AIC: 4780.4

Number of Fisher Scoring iterations: 18
```

Modelo 5:

```
f5<-c("Edad","SaldoTotalRCC","SaldoEmp1","EstadoCivil","Genero")
#,"NroInstituciones" ,"SaldoEmp2" ,"SaldoEmp3"  "nMonto_Consumo",
Desembolsado<-'Desembolsado'
modelo5<-
train(smote_sample_train_data[,f5],smote_sample_train_data[,Desembolsado]
,method='glm',trControl=ctrl,tuneLength=10,family = binomial())
 summary(modelo5)

Call:
NULL

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.94079  -0.00009   0.10006   0.62609   2.14647

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)            -7.762e-02  1.929e-01  -0.402  0.68743
Edad                   -1.084e-02  3.619e-03  -2.994  0.00276 **
SaldoTotalRCC           9.812e-05  1.414e-05   6.940 3.91e-12 ***
SaldoEmp1              -1.218e-04  1.656e-05  -7.356 1.89e-13 ***
EstadoCivilCONVIVIENTE  2.519e+00  1.345e-01  18.728  < 2e-16 ***
EstadoCivilSEPARADO     2.916e+00  2.484e-01  11.739  < 2e-16 ***
EstadoCivilSIN ESTADO  -1.879e+01  2.314e+02  -0.081  0.93528
EstadoCivilSOLTERO      2.103e+00  9.001e-02  23.363  < 2e-16 ***
EstadoCivilVIUDO        2.994e+00  4.409e-01   6.789 1.13e-11 ***
GeneroM                -5.037e-01  7.213e-02  -6.983 2.89e-12 ***
```

```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4835.4  on 7002  degrees of freedom
AIC: 4855.4

Number of Fisher Scoring iterations: 18
```

Modelo final:

```
modelo12<-
glm(Desembolsado~Edad+SaldoEmp1+EstadoCivil+Genero,data=smote_sample_trai
n_data,family=binomial())
summary(modelo12)

Call:
glm(formula = Desembolsado ~ Edad + SaldoEmp1 + EstadoCivil +
    Genero, family = binomial() data = smote_sample_train_data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.29509  -0.00009   0.18797   0.62632   1.86886

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)            -6.002e-02  1.916e-01  -0.313  0.75412
Edad                   -9.028e-03  3.592e-03  -2.513  0.01196 *
SaldoEmp1              -1.008e-05  3.792e-06  -2.658  0.00786 **
EstadoCivilCONVIVIENTE  2.517e+00  1.337e-01  18.831  < 2e-16 ***
EstadoCivilSEPARADO     2.908e+00  2.475e-01  11.749  < 2e-16 ***
EstadoCivilSIN ESTADO  -1.882e+01  2.325e+02  -0.081  0.93550
EstadoCivilSOLTERO      2.099e+00  8.916e-02  23.548  < 2e-16 ***
EstadoCivilVIUDO        3.004e+00  4.396e-01   6.833 8.29e-12 ***
GeneroM                -5.047e-01  7.169e-02  -7.041 1.91e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

```
Null deviance: 9720.7  on 7011  degrees of freedom
Residual deviance: 4889.6  on 7003  degrees of freedom
AIC: 4907.6

Number of Fisher Scoring iterations: 18
```

Resumen modelo final:

```
anova(modelo12,test="Chisq")
Analysis of Deviance Table
Model: binomial, link: logit

Response: Desembolsado

Terms added sequentially (first to last)


            Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                         7011     9720.7
Edad         1     11.3      7010     9709.4 0.0007892 ***
SaldoEmp1    1      8.2      7009     9701.2 0.0041688 **
EstadoCivil  5   4761.5      7004     4939.7 < 2.2e-16 ***
Genero       1     50.0      7003     4889.6 1.505e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

pred<-predict(modelo12,data=smote_sample_train_data,type="response")
library(ROCR)
rocpred<-prediction(pred,smote_sample_train_data$Desembolsado)
rocperf<-performance(rocpred,"tpr","fpr")

plot(rocperf,print.cutoffs.at=seq()colorize=TRUE,main="roc:0.8077036")

#curva gini

auc1<-performance(rocpred,measure = "auc")
modelo_auc<-(auc1@y.values [[1]])
gini<-2*(modelo_auc)-1
gini  #0.634145    #0.6414268    rose:0.6160516    aversampling:0.6669034
undersampling:0.7097529  both: 0.6891705
```

```
#0.8079129

head(predtest)
predtest<-predict(modelo12,data.test,type="response")
predclass<-ifelse(predtest>0.5,1,0)
head(predclass)
mc<-table(data.test$Desembolsado,predclass)
mc
# predclass
# 0     1
# 0 6019 1654
# 1    83  668
cl<-sum(diag(mc))/sum(mc)
cl #0.7329193 rose 0.6565217 oversampli:0.7633333  undersampling:0.7204969
both:0.673913
#0.7938034
espec<-mc[1,1]/sum(mc[1,])   #0.7625   rose0.754386   oversampli:0.7612903
undeersampli:0.6923077  both:0.7589286
espec
#0.7844389
sens<-mc[2,2]/sum(mc[2,])      #0.7037037    rose0.5603448    oversampli:
0.7655172  undersampli:0.7571429  both:0.5932203
sens
#0.8894807
library(caret)
predclass <- as.factor(predclass)
caret::confusionMatrix(predclass,data.test$Desembolsado,positive="1")
```

ANEXO 3: Análisis de Árbol de Decisión

```
library(rpart)
arbol.completo<-rpart(Desembolsado~nMonto_Consumo+Edad+NroInstituciones+
                      SaldoEmp1+SaldoEmp2+SaldoEmp3+EstadoCivil+Genero,
                       data= smote_sample_train_data,
                       method="class",
                       cp=0,
                       minbucket=0
)

plotcp(arbol.completo)

printcp(arbol.completo)
Classification tree:
rpart(formula = Desembolsado ~ nMonto_Consumo + Edad + NroInstituciones +
    SaldoEmp1 + SaldoEmp2 + SaldoEmp3 + EstadoCivil + Genero,
    data = smote_sample_train_data, method = "class", cp = 0,
    minbucket = 0)

Variables actually used in tree construction:
[1] Edad             EstadoCivil      Genero           nMonto_Consumo
NroInstituciones SaldoEmp1        SaldoEmp2
[8] SaldoEmp3

Root node error: 3506/7012 = 0.5

n= 7012

           CP nsplit  rel error  xerror      xstd
1  7.0650e-01      0 1.00000000 1.03594 0.0119343
2  4.8488e-03      1 0.29349686 0.29350 0.0084515
3  4.0882e-03      4 0.27780947 0.28694 0.0083726
4  1.9966e-03      7 0.26554478 0.27382 0.0082102
5  1.9015e-03      8 0.26354820 0.26982 0.0081595
6  1.4261e-03     17 0.24301198 0.26212 0.0080601
7  1.3311e-03     19 0.24015973 0.25927 0.0080227
8  1.1409e-03     22 0.23616657 0.25984 0.0080302
9  1.0268e-03     25 0.23274387 0.26212 0.0080601
```

```
10 9.9829e-04     30 0.22760981 0.26212 0.0080601
11 8.5568e-04     32 0.22561323 0.26013 0.0080340
12 7.6060e-04     39 0.21962350 0.26013 0.0080340
13 7.1306e-04     50 0.21049629 0.26098 0.0080452
14 6.1799e-04     55 0.20678836 0.26212 0.0080601
15 5.7045e-04     67 0.19880205 0.26583 0.0081082
16 5.1341e-04     94 0.18254421 0.26554 0.0081046
17 4.7538e-04    100 0.17940673 0.26526 0.0081009
18 4.2784e-04    107 0.17569880 0.26669 0.0081193
19 4.0746e-04    132 0.16457501 0.26754 0.0081303
20 3.8894e-04    149 0.15658871 0.26669 0.0081193
21 3.8030e-04    178 0.14175699 0.26811 0.0081376
22 3.5653e-04    194 0.13519681 0.26897 0.0081486
23 3.4861e-04    201 0.13205933 0.26783 0.0081340
24 3.4227e-04    217 0.12635482 0.26783 0.0081340
25 2.8523e-04    222 0.12464347 0.27981 0.0082851
26 2.4957e-04    427 0.06446092 0.27981 0.0082851
27 2.3769e-04    442 0.05989732 0.28351 0.0083308
28 2.2818e-04    473 0.05105533 0.28494 0.0083483
29 2.1392e-04    478 0.04991443 0.28665 0.0083691
30 1.9015e-04    497 0.04563605 0.29064 0.0084174
31 1.7114e-04    555 0.03422704 0.29464 0.0084651
32 1.4261e-04    565 0.03251569 0.29977 0.0085257
33 9.5075e-05    747 0.00627496 0.30063 0.0085357
34 0.0000e+00    808 0.00028523 0.30519 0.0085887
```

```
plotcp(arbol.completo) #grafico sedimentacion para q veas donde sera el
punto de corte.
> arbol.pruned<-prune(arbol.completo,cp=1.9966e-03 ) #aca se puede
cambiar #0.00172811
> printcp(arbol.pruned)

Classification tree:
rpart(formula = Desembolsado ~ nMonto_Consumo + Edad + NroInstituciones +
    SaldoEmp1 + SaldoEmp2 + SaldoEmp3 + EstadoCivil + Genero,
    data = smote_sample_train_data, method = "class", cp = 0,
    minbucket = 0)

Variables actually used in tree construction:
```

```
[1] Edad              EstadoCivil      nMonto_Consumo   NroInstituciones
SaldoEmp2

Root node error: 3506/7012 = 0.5

n= 7012

         CP nsplit rel error  xerror      xstd
1 0.7065031      0   1.00000 1.03594 0.0119343
2 0.0048488      1   0.29350 0.29350 0.0084515
3 0.0040882      4   0.27781 0.28694 0.0083726
4 0.0019966      7   0.26554 0.27382 0.0082102
> arbol.pruned$variable.importance  #importancia de variables
    EstadoCivil       SaldoEmp2 NroInstituciones            Edad       SaldoEmp1       SaldoEmp3   nMonto_Consumo
     1805.43750        380.06079        264.21892        160.87955        115.82154         86.55434         19.38146



library(rpart.plot)
rpart.plot(arbol.pruned,type=2,extra=101,cex=.7,nn=TRUE)



library(rpart.plot)
rpart.plot(arbol.pruned,type=2,extra=101,cex=.7,nn=TRUE)

#prediccion de un arbol podado


PRED.CART<-predict(arbol.pruned,
                   smote_sample_train_data,
                   type="class"
)

head(PRED.CART)
#calcuando la probabilidad q fugue o no
PROBA.CART<-predict(arbol.pruned
                    ,smote_sample_train_data
                    ,type="prob"
)

head(PROBA.CART)
PROBA.CART=PROBA.CART[,2]
```

```
PROBA.CARTtt<-predict(arbol.pruned
                      ,data.test
                      ,type="prob"
)

proball<-PROBA.CARTtt[,2]
length(proball)
#juntando el archivo
datoscart<-cbind(smote_sample_train_data,PRED.CART,PROBA.CART)

#calculo de error de la clasificacion
error<-mean(PRED.CART!=Desembolsado)
error <- mean(PRED.CART!=Desembolsado)
#Curva ROC y Área bajo la curva

library(pROC)
Desembolsado=smote_sample_train_data$Desembolsado
str(smote_sample_train_data)

areaROC<-auc(roc(Desembolsado,PROBA.CART))
areaROC

ROC<-plot.roc(Desembolsado,PROBA.CART,
              xlab="1-especificidad",
              ylab="sensibilidad",
              col="blue",
              main=paste('area bajo la curva=',round(areaROC,4))
)

#predicted= predict(arbol.completo,data.test)

#predtest<-predict(modelo12,data.test,type="response")

#predict.tree<-predict(arbol.pruned,newdata=data.test,type="class")
comprobando

predict.tree<-predict(arbol.pruned,newdata=data.test,type="class")
conf.tree<-confusionMatrix(table(predict.tree,data.test$Desembolsado))
conf.tree$overall[1]
conf.tree$table
```

```
# mctree<-with(data.test,table(predict.tree,Desembolsado))
# mctree
mctree<-table(data.test$Desembolsado,predict.tree)
mctree
# predict.tree
# 0     1
# 0 6356  1317
# 1    120   631
cg_T<-sum(diag(mctree))/sum(mctree)
cg_T
espe_T<-mctree[1,1]/sum(mctree[1,])
espe_T
sensi_T<-mctree[2,2]/sum(mctree[2,])
sensi_T
```

ANEXO 4: Análisis de Random Forest

```
RANDOM FOREST:
library(randomForest)
model_rf <-
train(Desembolsado~nMonto_Consumo+Edad+NroInstituciones+SaldoEmp1
+SaldoEmp2 +SaldoEmp3+EstadoCivil+Genero,
                  data = smote_sample_train_data, method='rf',trControl
=ctrl )   #15 minutos

> print(model_rf)
Random Forest

7012 samples
   8 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 6311, 6310, 6310, 6311, 6311, 6312, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.8745283  0.7490590
   7    0.8837131  0.7674271
  12    0.8812892  0.7625788

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 7.


> plot(rf)
> plot(rf, ylim = c(0.003,0.3)main="OOB estimate of  error rate: 10.77%")
#para  mostrar y la grafica se muestre de un panorama mas clara
> legend('topright', colnames(rf$err.rate) col = 1:3, fill = 1:3)
> print(rf) #lo q se grafica en esta salida muestra el error

Call:
```

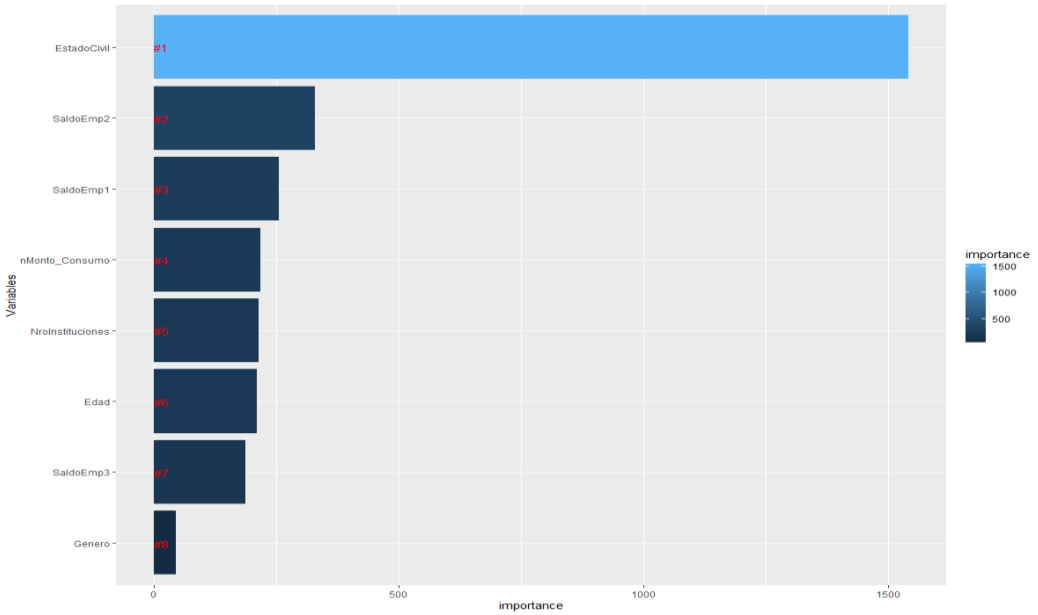
```
randomForest(formula = Desembolsado ~ nMonto_Consumo + Edad +      Genero
+ EstadoCivil + NroInstituciones + SaldoEmp1 + SaldoEmp2 +
SaldoEmp3, data = smote_sample_train_data, mtry = 7, ntree = 1000,
importance = TRUE)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 2

        OOB estimate of  error rate: 10.77%
Confusion matrix:
     0    1 class.error
0 3054  452  0.12892185
1  317 3189  0.09041643
```

```
library(ggplot2)
rankImportance <- varimportance %>%
  mutate(Rank = paste0('#', dense_rank(desc(importance))))
```

```
ggplot(rankImportance, aes(x = reorder(Variables, importance) y =
importance, fill = importance))+
  geom_bar(stat = 'identity')+
  geom_text(aes(x = Variables, y = 0.5, label = Rank) hjust = 0,
vjust=0.55, size = 4, colour = 'red')+
  labs(x = 'Variables')+
  coord_flip()
```

```
> varImp(rf)
                          0         1
nMonto_Consumo    55.16106  55.16106
Edad              42.20242  42.20242
Genero            23.05897  23.05897
EstadoCivil      192.62311 192.62311
NroInstituciones  39.48929  39.48929
SaldoEmp1         39.91222  39.91222
SaldoEmp2         43.37305  43.37305
SaldoEmp3         34.20237  34.20237
```




```
rf <- randomForest(Desembolsado~nMonto_Consumo+Edad+Genero
+EstadoCivil+NroInstituciones+SaldoEmp1  +SaldoEmp2 +SaldoEmp3,
                   data = smote_sample_train_data,
                   mtry = 2,
                   ntree = 1000,
                   importance = TRUE)

 predicrf<-predict(rf,data.test)

library(pROC)
#curva roc
rf.ROC <- roc(predictor=as.integer(predicrf) - 1,
              response=data.test$Desembolsado)

rf.ROC
plot(rf.ROC, col="blue",main="RANDOMFOREST")  #:CURVA ROC=0.8375

# mc2<-with(data.test,table(predicrf,Desembolsado))
# mc2<-with(table(predicrf,Desembolsado)data.test)

mc2<-table(data.test$Desembolsado,predicrf)

c_gRF<-sum(diag(mc2))/sum(mc2)
c_gRF  #0.8484136  #0.8589894 ultimo 0.8362184
# 0.8462726
espc_RF<-mc2[1,1]/sum(mc2[1,])    # 0.8814229 #0.9083447 ultimo
```

```
espc_RF
#0.8481689 mtry2
sens_RF<-mc2[2,2]/sum(mc2[2,])  #0.8218452  #0.8218332 ultimo
sens_RF
#0.8268975 mtry2
```

ANEXO 5: Análisis Stacking

```
# modelo 15.- Stacking de Modelos (LOGISTICA,TREE,RF)

Stacking=data.frame(predtest,proball,proba10);colnames(Stacking)=c("lg","
arb","RF")
Stacking$Stacking=apply(Stacking, 1, mean)
Stacking$response=ifelse(Stacking$Stacking<0.5,0,1)


proba15=Stacking$Stacking

# curva ROC
AUC15 <- roc(data.test$Desembolsado, proba15)
auc_modelo15=AUC15$auc

# Gini
gini17 <- 2*(AUC15$auc) -1

gini17
# Calcular los valores predichos
PREDSTACK <-as.factor(Stacking$response)
PREDSTACK<-as.factor(PREDSTACK)
# Calcular la matriz de confusi?n
tabla=caret::confusionMatrix(PREDSTACK,data.test$Desembolsado,positive="1
")
tabla
# tabla=confusionMatrix(PREDSTACK,data.test$Desembolsado,positive = "1")
#confusionMatrix(PREDSTACK, data.test$Desembolsado)$overall[1] #
0.7080745

# sensibilidad
Sensitivity15=as.numeric(tabla$byClass[1])

# Precision
Accuracy15=tabla$overall[1]

# Calcular el error de mala clasificaci?n
```

```
error15=mean(PREDSTACK!=data.test$Desembolsado)


mc13<-with(data.test,table(PREDSTACK,Desembolsado))
# c_gEnsa<-sum(diag(mc13))/sum(mc13)
# esp_Ens<-mc13[1,1]/sum(mc13[1,])
# esp_Ens
# sensi_Ens<-mc13[2,2]/sum(mc13[2,])
# sensi_Ens

varImp(Stacking$Stacking)
########

#graficos juntos
# Saving prediction percentage of each method
percent_1 <- data.frame(methods=c("Regresion logistica","Desicion
tree","Randon Forest","Ensamble") value=c(0,0,0,0))
#LOGISTICA
( percent_1$value[1] <- sum(diag(mc)) / sum(mc) * 100 )

#desition tree
(percent_1$value[2]<-sum(diag(mctree))/sum(mctree) * 100 )
#RANDOMFOREST
( percent_1$value[3] <- sum(diag(mc2))/sum(mc2) * 100 )
#Ensemble
( percent_1$value[4] <- sum(diag(mc13))/sum(mc13) * 100 )


#Predicting capacity comparison of methods
percent_1$methods <- paste(percent_1$methods, " - " ,
round(percent_1$value,digits = 2) , "%" , sep = "")
visualize_1 <- data.frame(valor=percent_1$value, group=
percent_1$methods)
visualize2_1 <- rbind(visualize_1,data.frame(valor=50, group=
visualize_1$group))


ggplot() +
  geom_point(data = visualize_1, aes(x = valor, y = group, color = group)
size = 4) +
```

```
  geom_path(data = visualize2_1, aes(x = valor, y = group, color = group,
group = group) size = 2) +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 0, vjust = 0.5, hjust = 0.5))
+
  labs(
    x = "Percentage of success",
    y = "Methods",
    title = "Percentage of success of the methods"
  )

#########indicadores


AUC_2     <-
MLmetrics::AUC(proba15,as.numeric(as.character(data.test$Desembolsado)))
GINI_2    <- 2*AUC_2-1
ks_2      <-
KS_Stat(proba15,as.numeric(as.character(data.test$Desembolsado)))
LogLoss_2 <-
LogLoss(proba15,as.numeric(as.character(data.test$Desembolsado)))

AUC_2; GINI_2;ks_2 ;LogLoss_2


# curva ROC
AUC15 <- roc(data.test$Desembolsado, proba15)
auc_modelo15=AUC15$auc

# Gini
gini17 <- 2*(AUC15$auc) -1
```