UNIVERSIDAD NACIONAL DE COLOMBIA

Detección de fraudes en tarjetas de crédito, emitidas por una empresa particular, usando métodos de clasificación binaria en un escenario altamente desbalanceado

Lina Victoria Parra Duque

Universidad Nacional de Colombia
Facultad de Ciencias, Escuela de Estadísticas
Medellín, Colombia 2023

Detección de fraudes en tarjetas de crédito, emitidas por una empresa particular, usando métodos de clasificación binaria en un escenario altamente desbalanceado

Lina Victoria Parra Duque

Tesis de grado presentada como requisito parcial para optar al título de:
Magíster en Ciencias - Estadística

Director:
Mauricio Alejandro Mazo Lopera
Doctor en Ciencias - Estadística

Linea de profundización:
Estadística Financiera

Universidad Nacional de Colombia
Facultad de Ciencias, Escuela de Estadística
Medellín, Colombia
2023

Agradecimientos

A Dios primeramente por darme la oportunidad de iniciar este camino de aprendizaje y a todas aquellas personas que hicieron parte de este proceso y que de una u otra forma hoy hacen posible este logro.

Resumen

Detección de fraudes en tarjetas de crédito, emitidas por una empresa particular, usando métodos de clasificación binaria en un escenario altamente desbalanceado

En los últimos años muchas empresas han incorporado en sus portafolios la tarjeta de crédito como estrategia de venta, lo que ha permitido que este medio de pago se vuelva cada vez más frecuente, convirtiéndose en un foco de atención para los casos de delitos. La empresa objeto de estudio no ha sido ajena a esta modalidad de fraude que en los últimos años se ha incrementado considerablemente generando pérdidas en la organización. Por esta razón, el presente trabajo busca implementar un modelo que identifique fraudes por medio de la aplicación de métodos de clasificación reduciendo pérdidas financieras en la organización.

En este trabajo se presenta un modelo para la detección de fraudes en tarjetas de crédito para una empresa en particular, aplicado a una base de datos altamente desbalanceada, es decir, que las observaciones de alguna de sus clases es mayor con respecto a las demás. Para contrarrestar este problema se utiliza la técnica de remuestreo, la cual se basa en dos aspectos fundamentales como agregar y quitar patrones; de este modo para el equilibrio de las clases se utilizan dos metodologías: la primera corresponde al método de submuestreo (undersampling) la cual consiste en quitar patrones, los cuales se eliminarían de la clase mayoritaria y de esta forma igualarla a la clase minoritaria; y la segunda corresponde al sobremuestreo (oversampling), que consiste básicamente en agregar patrones los cuales se anexarían a la clase minoritaria y de esta forma igualarla a la clase mayoritaria.

Para la medición del desempeño se utilizan métricas de sensibilidad y especificidad para los diferentes modelos aplicados entre los cuales se encuentra el modelo sin balanceo de datos y los modelos con datos balanceados. Se evidencia que con la metodología de balanceo oversampling hay una mejora considerable de estos indicadores con respecto a los otros modelos, ya que es más sensible al detectar en mayor proporción los clientes que cometen fraude.

Palabras clave: undersampling, oversampling, sensibilidad, especificidad, clasificación, estadística.

Abstract

Fraud Detection in Credit Cards, Issued by a Specific Company, Using Binary Classification Methods in a Highly Imbalanced Scenario

In recent years, many companies have incorporated credit cards into their portfolios as a sales strategy, which has made this payment method increasingly common and a focal point for criminal activity. The company under study has not been immune to this type of fraud, which has significantly increased in recent years, resulting in losses for the organization. Therefore, this study aims to implement a model that identifies fraud through the application of classification methods, reducing financial losses in the organization.

This study presents a model for fraud detection in credit cards for a specific company, applied to a highly imbalanced database, meaning that the observations for some classes are greater than others. To counteract this problem, the resampling technique is used, which is based on two fundamental aspects: adding and removing patterns. Thus, two methodologies are used to balance the classes. The first method is undersampling, which involves removing patterns from the majority class to equalize it with the minority class. The second method is oversampling, which involves adding patterns to the minority class to equalize it with the majority class.

To measure performance, sensitivity and specificity metrics are used for different applied models, including the model without data balancing and the models with balanced data. It is evident that the oversampling balancing methodology shows a considerable improvement in these indicators compared to the other models, as it is more sensitive in detecting fraudulent customers to a greater extent.

Keywords: undersampling, oversampling, sensitivity, specificity, classification, statistics.

Contenido

Lista de Figuras

Lista de Tablas

1 Introducción

Las transacciones fraudulentas implican perdidas considerables a las empresas que dentro de sus portafolios de servicios ofrecen la tarjeta de crédito. Estos fraudes se traducen en un riesgo inminente para el logro de los objetivos que tienen las organizaciones de maximizar sus utilidades. Lo anterior implica que las empresas orienten los esfuerzos hacia métodos que permitan detectar a tiempo este tipo de transacciones evitando perdidas financieras.
Este proyecto esta encaminado a encontrar un método estadístico que permita detectar a tiempo transacciones fraudulentas que le permitan a la empresa objeto de estudio minimizar el riesgo de fraude.

1.1. Planteamiento del problema

La empresa objeto del estudio tiene dentro de sus servicios la tarjeta de crédito como estrategia de reconocimiento para clientes de los servicios públicos. En los últimos años los fraudes por este medio se han incrementado considerablemente generando pérdidas al negocio. Este trabajo busca detectar este tipo de fraudes por medio de herramientas estadísticas proponiendo una metodología para detectar fraudes en una base de datos altamente desbalanceada y así reducir su incidencia.

En este capítulo haremos una presentación de los aspectos relacionados con el problema a ser abordado, con el fin de proveer una contextualización que abra paso al planteamiento de los métodos aplicados a la base de datos y así, proponer una solución aproximada al problema particular.

1.2. Pagos electrónicos con tarjeta de crédito

En los últimos años, el uso del dinero electrónico ha tenido un crecimiento acelerado y es que con la pandemia, las personas se vieron en la necesidad de usar medios de pago digitales en sus compras o adquisiciones.
Según Payments (2022), entre los medios de pago más frecuentes se encuentra la tarjeta de crédito y se tienen cifras de que 7 de cada 10 personas la utilizan en América Latina, alcanzando en el 2021 compras en internet por alrededor del 73.7 %.

Así, como crecen los medios de pago electrónico, también crece la preocupación por el aumento en los fraudes, llevando a las empresas que ofrecen servicios financieros a través de medios digitales, a tratar de generar sistemas de detección para prevenir y contrarrestar este tipo de riesgos.
Son preocupantes las cifras de fraude a nivel global entre el 2019 y 2021, ya que en contraste con años anteriores, aumentaron en un 52.2 % las sospechas de fraude digital y, peor aún, el robo de identidad aumentó en un 81.8 % en todas las industrias. Colombia no es ajena a estas cifras, con un 75 % en robo de identidad y un 73 % en fraude en tarjeta de crédito. (TransUnion, 2022).

1.3. Fraude en tarjeta de crédito

Según la Real Academia Española, el fraude es la acción contraria a la verdad y a la rectitud, que perjudica a la persona contra quien se comete. El fraude con tarjeta de crédito es el uso fraudulento de los datos de la tarjeta para comprar un producto o servicio. Estas transacciones pueden realizarse física o digitalmente (Alenzi y Aljehane, 2020a).
Jain, NamrataTiwari, ShripriyaDubey, y Jain (2019), clasifica los fraudes en las tarjetas de crédito de la siguiente manera:

- **Fraudes de aplicaciones:** Cuando el estafador al obtener datos confidenciales del usuario, como su contraseña y usuario, accede a todo el control de su cuenta en la aplicación para luego llevar a cabo las transacciones.
- **Impresiones electrónicas o manuales de las tarjetas de crédito:** El estafador extrae información confidencial por medio de la banda magnética que está presente en la tarjeta, permitiendole usar las credenciales y finalmente llevar a cabo las transacciones.
- **Tarjeta no presente:** Este tipo de fraude se lleva a cabo en las tarjetas de crédito, sin que la tarjeta física esté presente durante la transacción.
- **Tarjetas falsificadas:** El estafador hace una tarjeta con copia de todos los datos de la banda magnética de la tarjeta original. La tarjeta falsa es completamente funcional para realizar las transacciones.
- **Tarjeta perdida o robada:** Se lleva a cabo en los casos en que el titular de la tarjeta original extravía su tarjeta, llegando a manos de los estafadores, que luego la usan para realizar pagos. Es difícil hacer esto a través de la máquina, ya que se requiere un número de pin. Las transacciones en línea son bastante fáciles para el defraudador.
- **Robo de identificación de la tarjeta:** En el robo de identidad, el defraudador adquiere los datos de la tarjeta original para hacer uso de una tarjeta o para abrir una nueva cuenta. Este tipo de fraude es el más difícil de identificar.

- **Fraude de tarjeta no recibida por correo:** Cuando un cliente solicita una tarjeta, se necesita algún tiempo para todos los trámites procesales. Si el defraudador intercepta por medio del correo electrónico la entrega, puede registrar la tarjeta a su nombre y utilizarla para realizar compras. Este fraude también se conoce como fraude de emisión nunca recibida.

- **Adquisición de cuenta:** El estafador puede acceder a los detalles de la cuenta del titular original de la tarjeta y a varios documentos relevantes. Luego pueden comunicarse con la compañía de la tarjeta de crédito y hacerse pasar por el titular original de la tarjeta e incluso pedirles que cambien la dirección. La tarjeta duplicada se enviará a la dirección nueva o falsa y el delincuente podrá hacer uso de ella.

- **Sitios de comerciantes falsos:** El cliente titular de la tarjeta de crédito queda atrapado en una página web falsa, creada por el estafador, y una vez que se realiza la transacción, se recopila toda la información relacionada con esta y el estafador la utiliza para realizar intercambios fraudulentos.

- **Colusión de comerciantes:** En este tipo de fraude, los datos del titular de la tarjeta se comparten con un tercero, sin autorización del titular de la tarjeta.

En la actualidad se pueden encontrar diversas soluciones a esta problemática a través de modelos estadísticos que permiten identificar patrones y por ende predecir comportamientos fraudulentos, aportando herramientas importantes a las empresas para que puedan minimizar el riego de fraude (Alenzi. y Aljehane, 2020).

1.4. Objetivos de esta tesis propuestos en el anteproyecto

1.4.1. Objetivo general

Proponer una metodología para detectar fraudes en tarjetas de crédito, emitidas por una empresa particular, basada en métodos de clasificación binaria y considerando datos altamente desbalanceados.

1.4.2. Objetivos específicos

- Explorar distintos métodos de clasificación binaria, propuestos en la literatura, para abordar escenarios altamente desbalanceados.

- Analizar distintas técnicas de remuestreo que apunten a enfrentar el problema de datos binarios altamente desbalanceados.

- Comparar el desempeño de la combinación de distintas técnicas de remuestreo y clasificación binaria en un conjunto de datos particular, con el fin de obtener una metodología de detección de fraude en tarjetas de crédito.

1.4.3. Pregunta de investigación

Con base en los objetivos planteados anteriormente tenemos como pregunta de investigación: ¿Cómo seleccionar un método estadístico adecuado para cuantificar el riesgo de fraude en individuos que adquieren una tarjeta de crédito en una empresa de servicios públicos, teniendo en cuenta un conjunto de modelos de clasificación binaria y un alto desbalance entre las categorías fraude y no fraude?.
En el siguiente capítulo se presentarán algunos modelos de clasificación que fueron abordados a la hora de explorar posibles soluciones al problema de predecir si un cliente, asociado con la empresa proveedora de la base de datos, tiene o no alguna probabilidad significativa de cometer fraude.
Cabe resaltar que en la base de datos analizada se cuenta con un gran conjunto de variables que permiten describir un perfil específico a cada cliente, además de una variable dicotómica que clasifica a los clientes con 0 (no fraude) y 1 (sí fraude). Además, se observa un desbalanceo considerable entre las dos últimas categorías (mostrando muchos más unos que ceros) y por tanto, se realizó una búsqueda de alternativas de remuestreo que serán expuestas al final del siguiente capítulo.

Los capítulos de este trabajo están organizados de la siguiente forma. En el capítulo 2 se explorarán algunos modelos de clasificación utilizados en la literatura, también se presentarán alternativas basadas en remuestreo, para tratar el problema de desbalanceo entre casos de fraude y no fraude. En el capítulo 3 se tratarán temas relacionados con la extracción y transformación de los datos, análisis descriptivo, criterios de selección de variables, aplicación de metodologías para el tratamiento de datos desbalanceados y aplicación de diferentes técnicas estadísticas para la detección de fraudes en la base de datos objeto de estudio de este trabajo. Finalmente, en el último capítulo se presentarán algunas conclusiones y recomendaciones.

2 Algunos modelos de clasificación y alternativas al problema de clases desbalanceadas

A continuación exploraremos algunos modelos de clasificación utilizados en la literatura en el contexto de una variable respuesta dicotómica. También haremos una breve presentación de alternativas, basadas en remuestreo, al problema de desbalanceo entre casos de fraude y no fraude.

2.1. Regresión logística

Dentro de las técnicas de aprendizaje supervisado podemos encontrar los modelos de regresión cuya respuesta es numérica y modelos de clasificación en donde la respuesta puede ser de tipo cualitativa binaria, como por ejemplo, si hay fraude o no en una transacción con tarjeta de crédito.
La regresión logística es un método estadístico de clasificación que busca explicar y predecir características cualitativas por medio de variables explicativas. Es un método adecuado para resolver problemas de clasificación binaria, dado que el resultado es de tipo dicotómico. Matemáticamente, si definimos la variable aleatoria binaria Y como 0 (si no hay fraude) y 1 (si se presenta el fraude), entonces podemos modelar la probabilidad $p = P(Y = 1/x)$, a través de un modelo logístico como:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k, \tag{2-1}$$

donde $x_1, x_2, \ldots, x_k$ representan covariables que definen el perfil de un individuo a ser examinado y $\beta_0, \beta_1, \ldots, \beta_k$ son los parámetros del modelo a ser estimados. La función link, logit, considera el logaritmo de la razón de probabilidades de que un cliente incurra en un fraude en contraste con la probabilidad de que no lo cometa.
Haciendo un manejo algebraico de la expresión (2-1), se puede llegar a que la probabilidad de fraude es igual a:

$$p = \frac{1}{1 + \exp\{\beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k\}} \tag{2-2}$$

2.1.1. Método de estimación

Mediante la expresión (2-3) se puede estimar la probabilidad de que el i-ésimo cliente, para $i = 1, \ldots, n$ (donde n es el tamaño de la muestra) incurra en fraude, en función de un conjunto de covariables $x_{1i}, x_{2i}, \ldots, x_{ki}$ que describen su perfil. Los parámetros $\beta_0, \beta_1, \ldots, \beta_k$ se estiman con el método de máxima verosimilitud, teniendo en cuenta que:

$$p_i = P(Y_i = 1/x_i) = \frac{\exp\{\beta_0 + \beta_1 x_{1i} + \ldots + \beta_k x_{ki}\}}{1 + \exp\{\beta_0 + \beta_1 x_i + \ldots + \beta_k x_k\}} \quad (2\text{-}3)$$

y obteniendo la función de verosimilitud dada por:

$$L(\beta_0, \ldots, \beta_k/x) = \prod_{l=1}^{n} p_i{}^{y_i}(1 - p_i)^{1-y_i}, \quad (2\text{-}4)$$

donde y_i es igual a 0, si el i-ésimo cliente no cometió fraude y 1 en caso contrario.
De la expresión (2-4), podemos obtener la función de log-verosimilitud:

$$\ell(\beta_0, \ldots, \beta_k) = \ln[L(\beta_0, \ldots, \beta_k/x)]$$

Los estimadores de $\beta_0, \ldots, \beta_k/x$, se obtienen mediante la optimización de la función de log-verosimilitud y esta tarea se realizó en el software R mediante el uso de la función **glm** del paquete *stats*.

2.1.2. Supuestos de la regresión logística

Aunque la regresión logística no tiene supuestos tan restrictivos como la regresión lineal particularmente el de Linealidad, normalidad y homoscedasticidad, la regresión logística si hace algunas suposiciones que es importante acatar:

- **Observaciones independientes:** las observaciones no deben de estar relacionadas entre si y no deben haber mediciones repetidas del mismo individuo.
- **Variable de respuesta binaria:** se asume que la variable de respuesta solo toma dos valores posibles, en este caso fraude y no fraude.
- **el tamaño de la muestra es suficientemente grande:** con el fin de llegar a conclusiones validas, la regresión logística requiere de un conjunto de datos suficientemente grande que satisfaga este requisito.

En un escenario ideal donde se cuente con un número de clientes que comenten fraude igual de balanceado a los clientes que no lo cometen, el modelo de regresión logística debería funcionar muy bien. Sin embargo, la realidad es que el número de fraudes reportados y verificados es muy inferior a los casos que no tienen ninguna sospecha. Como veremos más

adelante, esta situación lleva a una tasa de error de clasificación muy baja que no corresponde con un modelo de clasificación bien construido.
Es por esto, que teniendo un desequilibrio tan alto entre las categorías fraude y no fraude en la base de datos que pretendemos analizar en este trabajo (fraude comprobado representa el 1 % del total de datos), se hace necesario explorar metodologías relacionadas con clases desbalanceadas.
Otro método de clasificación que permite asociar una variable categórica con un conjunto de covariables es conocida como *árboles de clasificación o decisión*. A continuación daremos una breve presentación de los mismos.

2.2. Árboles de clasificación

Partiendo de que la variable respuesta en el problema de fraudes es categórica, utilizaremos modelos de árboles de clasificación los cuales se basan en algoritmos de aprendizaje supervisado y cuya finalidad es realizar predicciones a partir de patrones aprendidos en una base de datos etiquetada. (Norman, 2019) Las variables predictoras generan un espacio el cual se divide en distintas regiones o ramas y en las observaciones que caen en cada rama se hace la misma predicción que equivale a la moda o clase que más se repite.
Una desventaja en los arboles de decisión es que son muy sensibles a cambios en los datos con los que fueron entrenados, este se puede mejorar utilizando un método conocido como **bagging**, que tienen como objetivo la reducción de la varianza a partir de muestras aleatorias (Springer, 1996).
Para calcular el error en árboles de clasificación se utiliza el **Índice de Gini** (Medina, 2001)

$$\mathrm{G}_m = \sum_{k=1}^{K} \hat{P}_{mk}(1 - \hat{P}_{mk}), \tag{2-5}$$

donde $\hat{P}_{mk}$ representa la proporción de las observaciones en la m-ésima region que son de la categoría k. El índice de Gini mide la varianza total sobre todas las K clases de la variable categórica Y, en la región m. El **Índice de Gini** se conoce como una medida de la pureza del nodo por que se puede ver que, el índice de Gini es pequeño cuando los $\hat{P}_{mk}$ son cercanos a cero o a uno (James, Witten, Hastie, y Tibshirani, 2013).
Otra medida del error para la clasificación es la **Entropía**(Rebollo Neira, Plastino, y Zyserman, 1994):

$$\mathrm{D}_m = \sum_{k=1}^{k} \hat{P}_{mk} \ln(1 - \hat{P}_{mk}) \tag{2-6}$$

También ocurre que la entropía es pequeña cuando los $\hat{P}_{mk}$ son cercanos a cero o a uno, lo cual lleva también a una interpretación de que cuando la entropía es cercana a cero, entonces el nodo es "puro" (James et al., 2013).

Otras alternativas al modelo logístico y a los árboles de clasificación tienen que ver con métodos basados en aprendizaje de máquinas *machine learning*. Ulises Jiménez Cardoso (2020), por ejemplo, expone la aplicación de algunas de estas metodologías para la detección de fraudes, entre las cuales encontramos las máquinas de soporte vectorial y en donde, bajo ciertos escenarios, no solo entrega el mejor ajuste, sino que hace un excelente uso de rendimiento de máquina al tener un tiempo de respuesta mas eficiente.
A continuación daremos una breve exposición del método conocido como *máquinas de soporte vectorial* con el fin de aplicarlo a nuestro conjunto de datos y así, poder contrastar los resultados con los demás métodos propuestos.

2.3. Máquinas de Soporte Vectorial

Este algoritmo de aprendizaje automático es utilizado para problemas de clasificación y regresión, y se basa en hiperplanos que permiten dividir las muestras linealmente (suponiendo que son separables) y buscar la línea óptima entre las infinitas soluciones que se presentan.

La notación matemática es tomada de (James et al., 2013)

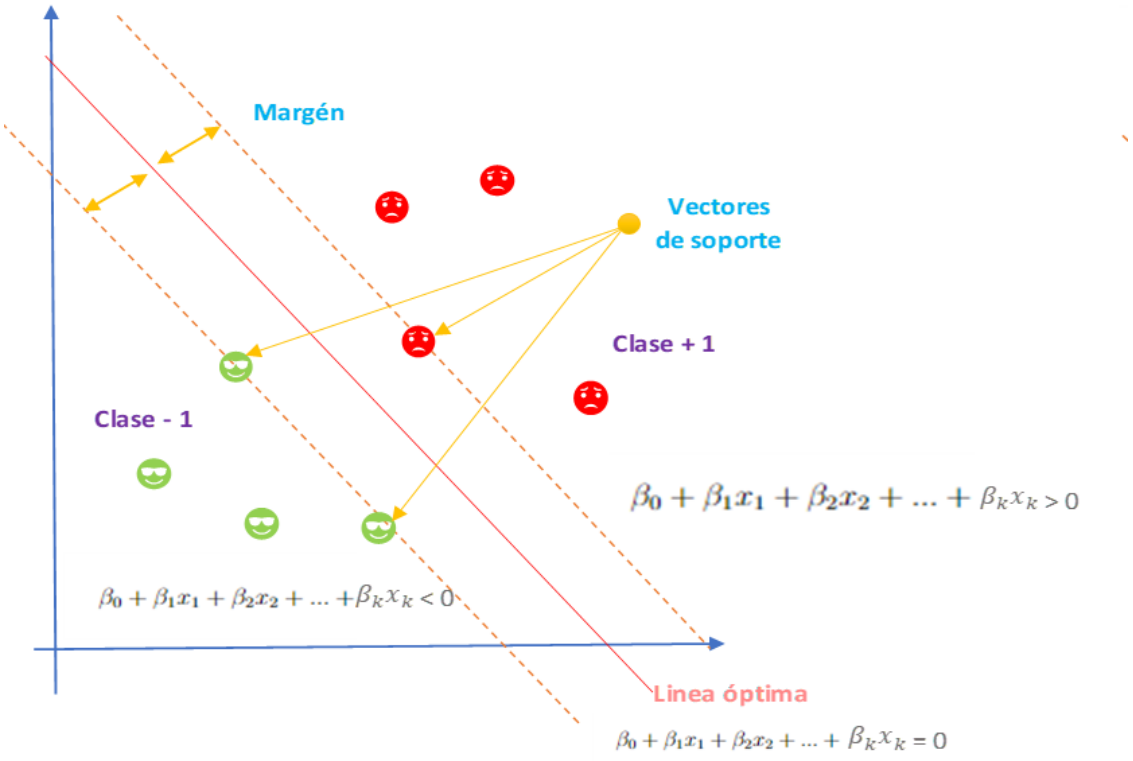


Figura 2-1: Frontera o hiperplano entre los datos a clasificar en Maquinas de soporte vectorial(Fuente: elaboración propia)

La Figura **2-1** ejemplifica un escenario simple de separación por medio de una recta. En este caso, el hiperplano está representado por la siguiente función:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0 \tag{2-7}$$

Donde β_0 , β_1 y β_2 son parámetros y (x_1, x_2) son los pares de valores para los que se cumple la igualdad. Son puntos del hiperplano.
Cuando $\mathbf{x}$ no satisface la ecuación:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 < 0, \tag{2-8}$$

o bien

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 > 0 \tag{2-9}$$

El hiperplano divide un espacio p-dimensional en dos mitades, asi el punto $\mathbf{x}$ cae a un lado o al otro en el hiperplano, para saber exactamente en que lado se encuentra un determinado punto $\mathbf{x}$, solo hay que calcular el signo de la ecuación.

Cuando los casos son separables linealmente ,entonces, un hiperplano de separación cumple que:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k > 0, \text{ si } y_i = 1, \tag{2-10}$$

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k < 0, \text{ si } y_i = -1 \tag{2-11}$$

las ecuaciones 2-10 y 2-11 pueden simplificarse en la siguiente ecuación:

$$y_i(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k) > 0, \sim \ i = 1, \vdots, n \tag{2-12}$$

Cuando no existe un hiperplano de separación y no existe un clasificador de margen máximo, se hace el uso del truco kernel para resolver problemas no lineales, en donde se aumenta de dimensión el espacio de interés para encontrar la separación entre clases (Ulises Jiménez Cardoso, 2020).

Los tipos de kernels mas usados en máquinas de soporte vectorial (SVM) se encuentran:

- **Kernel lineal:** cuantifica la similitud de un par de observaciones usando la correlación lineal de Pearson.

$$K(x_i, x_i') = \sum_{j=1}^{k} x_{ij}, x_{ij}' \tag{2-13}$$

- **Kernel polinómico:** permite un límite de decisión mucho más flexible.

$$K(x_i, x_i') = \left(1 + \sum_{j=1}^{p} x_{ij}, k x_{ij}'\right)^d \tag{2-14}$$

- **Kernel radial:** el valor de γ controla el comportamiento del kernel. Cuando es muy pequeño, el modelo final es equivalente al obtenido con un kernel lineal, a medida que aumenta su valor, también lo hace la flexibilidad del modelo

$$K(x_i, x_i') = \exp\left(-\gamma \sum_{j=1}^{k} x_{ij}, x_{ij}'\right)^2 \tag{2-15}$$

El método SVM se ha convertido en una de las técnicas más interesantes en el tema de aprendizaje automático, superando la precisión de otros modelos (Jaramillo Chaparro, 2015).

Recordando que el conjunto de datos que es objeto de estudio en este trabajo tiene una estructura altamente desbalanceada en la variable de interés (fraude o no fraude), presentamos a continuación algunos métodos de remuestreo que permiten aproximar una solución al desbalanceo.

2.3.1. Métodos de selección de variables

Estas metodologías permiten encontrar el mejor subconjunto de variables predictoras, con el fin de mejorar la interpretabilidad del modelo y reducir la varianza. En este trabajo se tendrán en cuenta las siguientes metodologías:

- *Selección hacia adelante*: Este método consiste en hacer una selección de variables paso a paso hacia adelante, partiendo del supuesto que no tiene variables explicativas; de esta manera evalúa cada variable y la agrega al modelo si está altamente correlacionada con la variable dependiente. Para la aplicación de esta metodología se utiliza la función regsubsets con el método forward.

- *Selección hacia atrás*: Este método consiste en hacer una selección de variables paso a paso hacia atrás, partiendo de todas las variables, se van excluyendo las menos influyentes. Para la aplicación de esta metodología se utiliza la función regsubsets con el método backward.

- *Stepwite*: Esta metodología es una combinación de las dos metodologías anteriores, consiste en ir introduciendo o eliminando variables de manera progresiva determinando si en cada etapa las variables deben de permanecer o no en el modelo..

La selección del mejor modelo se hace a través de los siguientes estadísticos de bondad de ajuste como: r-cuadrado ajustado (R^2 adj), C_p de Mallows, criterio de información de Schwartz (BIC), suma residual de cuadrados (RSS), el objetivo es escoger aquel modelo que tenga una mejor medida global, esta se puede observar graficamente para los estadísticos que menor valor presentan

Para un modelo con d variables regresoras:

$$C_p = \frac{1}{n}\left(SS_{res} + 2d\hat{\sigma^2}\right) \tag{2-16}$$

$$AIC = \frac{1}{n\hat{\sigma^2}}\left(SS_{RES} + 2d\hat{\sigma^2}\right) \tag{2-17}$$

$$R^2_{Ajustado} = 1 - \frac{\frac{SS_{RES}}{(n-d-1)}}{\frac{SS_T}{n-1}} \tag{2-18}$$

2.4. Alternativa para el tratamiento de clases desbalanceadas

Cuando en un conjunto de datos las clases no se representan igual (una clase predomina sobre otra en términos de cantidad de casos), se puede decir que es una situación de datos desbalanceados. La Figura **2-2** ilustra gráficamente esta situación de desbalanceo. Esto puede afectar los resultados del modelo de clasificación utilizado (siendo regresión logística uno de ellos), dado que el objetivo de dichos modelos es minimizar los errores de clasificación y, la clase con mayor número de observaciones, obtendrá la mayor probabilidad, de esta manera el resultado estará sesgado al clasificar nuevas observaciones (Lamblet, 2019).

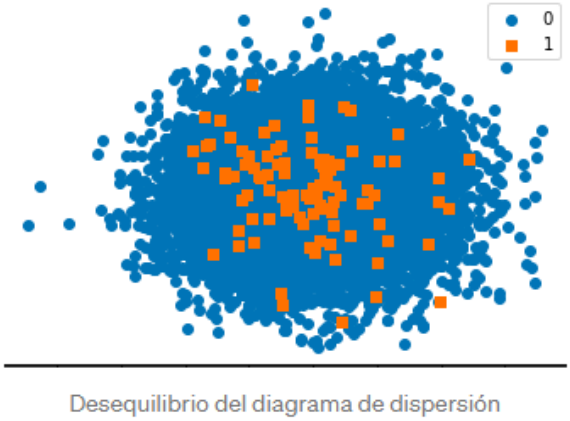


Figura 2-2: Conjunto de datos desbalanceados: Data Hackers

Existen muchas técnicas de remuestreo utilizadas para equilibrar datos desbalanceados y de esta forma aliviar el efecto de la distribución sesgada Osorio (2019).
Según Brownlee (2020), para solucionar este problema se pueden aplicar técnicas de sobremuestreo, submuestreo y combinación de técnicas:

- **Técnicas de sobremuestreo:** Las más relevantes son,
  1. Sobremuestreo aleatorio ROS (Random Oversampling): Es un método no heurístico que tiene como objetivo equilibrar la clase minoritaria con la mayoritaria. (Torres-Vásquez, Hernández-Torruco, Hernández-Ocaña, y Chávez-Bosquez, 2021)
  2. Técnica de sobremuestreo de minorías sintéticas (SMOTE): (Synthetic Minority Oversampling Technique) sobremuestra la clase minoritaria generando instancias sintéticas con el objetivo de equilibrarla con la mayoritaria. (Torres-Vásquez et al., 2021)

- **Técnicas de submuestreo:** Las más relevantes son,
  1. Submuestreo aleatorio RUS (Random Under Sampling): Elimina ejemplos de la clase mayoritaria para equilibrar el conjunto de datos. (Kraiem et al., 2020)
  2. Regla condensada del vecino más cercano (CNN).
  3. Near Miss Underampling.

- **Combinación de técnicas:** una de la más relevante es,
  1. SMOTE y submuestreo aleatorio:un método de sobremuestreo propuesto para abordar el problema delsobreajuste, haciendo más general el límite de decisión de la clase minoritaria. (Kraiem et al., 2020)

En este trabajo se aplicaran las técnicas de Sobremuestreo Random Oversampling y Sobremuestreo Random UnderSampling.

Otro aspecto de gran relevancia en el planteamiento de modelos de clasificación tiene que ver con la evaluación de los mismos en cuanto a la tasa de clasificación correcta o incorrecta, esta última más conocida como tasa de error de clasificación. A continuación presentamos algunos criterios de evaluación que pueden ser también utilizados para la selección cuando se cuenta con más de un modelo de clasificación binaria.

2.5. Evaluación y selección de modelos de clasificación dicotómica

Existen muchas posibilidades de comparar y evaluar los resultados de los diferentes modelos predictivos. Teniendo en cuenta que estamos frente a un problema de clasificación, se recurre a métricas apropiadas para este tipo de casos como: Exactitud (Accuracy), sensibilidad (sensitivity) y especificidad (specificity). La Figura **2-3** presenta la estructura de la matriz de confusión para el caso de la variable dicotómica fraude o no fraude.

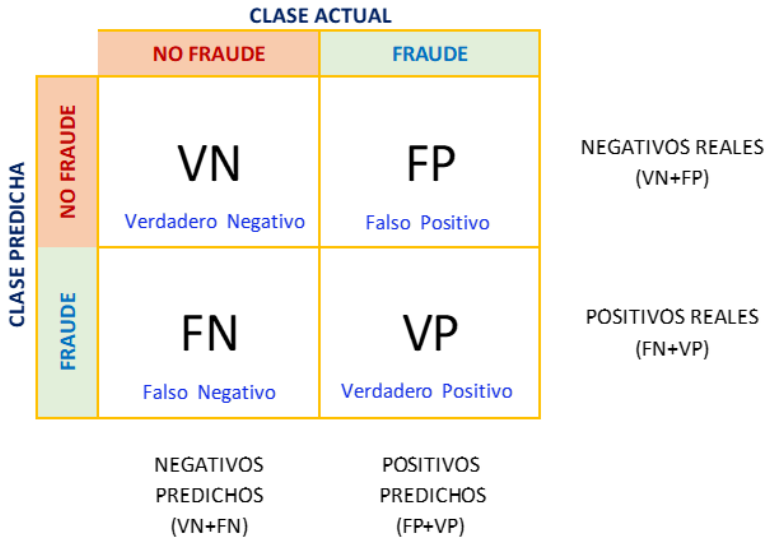


Figura 2-3: Matriz de Confusión en fraudes de tarjeta de crédito (Fuente: elaboración propia)

- **Exactitud (accuracy)** es la capacidad que tiene el modelo de clasificar los fraudes y no fraudes garantizando la calidad de las predicciones realizadas. Alenzi y Aljehane

(2020b), matemáticamente se define de la siguiente forma:

$$\text{Exactitud} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{VN} + \text{FP} + \text{FN}} \tag{2-19}$$

- **Sensibilidad (sensitivity):** se refiere a la tasa de verdaderos positivos por lo tanto indica la capacidad del estimador para detectar fraudes cuando realmente lo son. Alenzi y Aljehane (2020b) define sensibilidad como la capacidad para identificar resultados "verdaderos positivos", matemáticamente se define de la siguiente forma:

$$\text{Sensibilidad} = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{2-20}$$

- **Especificidad (specificity):** se refiere a la tasa de verdaderos negativos, es decir la capacidad que tiene el estimador para detectar no fraudes cuando realmente estos no lo son. Alenzi y Aljehane (2020b) define especificidad como la capacidad para identificar resultados "verdaderos negativos", matemáticamente se define de la siguiente forma:

$$\text{Especificidad} = \frac{\text{VN}}{\text{VN} + \text{FP}} \tag{2-21}$$

- **Curva ROC**: una forma de evaluar la capacidad discriminativa de un modelo es a través de la curva ROC (receiver operating characteristics), en esta se puede observar de forma gráfica la sensibilidad frente a la especificidad, así como comparar la capacidad del modelo para clasificar los fraudes y los que no son fraudes. El AUC area bajo la curva ROC, muestra el rendimiento de un clasificador binario por medio de un puntaje que indica que tan bien funciona el modelo, si este puntaje es cercano a 0 se puede afirmar que el modelo está prediciendo la clase incorrecta la mayoría de las veces, si este puntaje es cercano a 0,5 el modelo no tiene la capacidad de distinguir una clase negativa de una clase positiva y si este puntaje es cercano a 1 el modelo es capaz de distinguir perfectamente entre una clase positiva y una clase negativa.

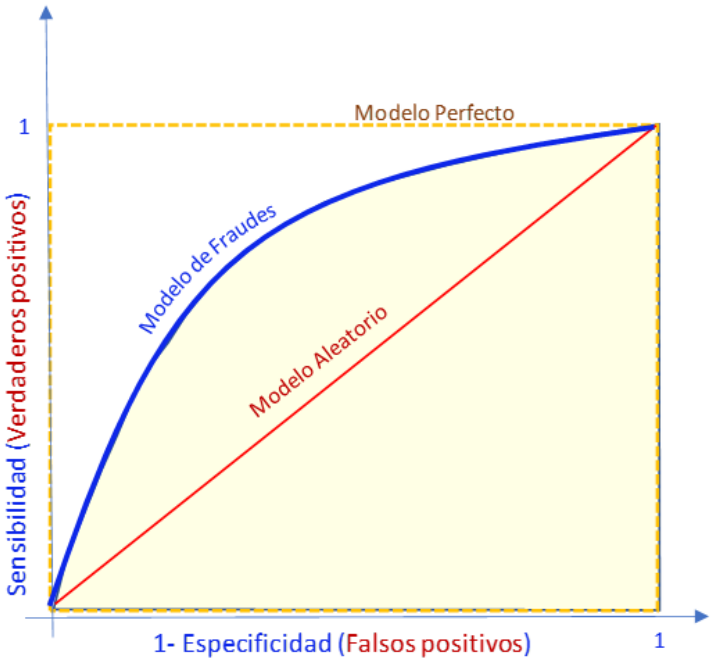


Figura 2-4: Curva Roc (Fuente: elaboración propia)

En la Figura **2-4**, se puede observar la proporción de falsos positivos en el eje X (1– Especificidad) y la proporción de verdaderos positivos (Sensibilidad) en el eje Y.

En el siguiente Capítulo aplicaremos los distintos modelos mencionados anteriormente, además de exponer cómo se abordó el problema de desbalanceo de los datos con varias técnicas y el contraste que se genera antes y después de aplicar las mismas.

3 Aplicación de modelos de clasificación para la detección de fraudes

En este capítulo se tratarán temas relacionados con la extracción y transformación de los datos, análisis descriptivo, criterios de selección de variables, aplicación de metodologías para el tratamiento de datos desbalanceados y aplicación de diferentes técnicas estadísticas para la detección de fraudes en la base de datos objeto de estudio de este trabajo.

3.1. Origen y pre-procesamiento de los datos

Los datos utilizados para la elaboración de este modelo provienen de una empresa particular cuya información corresponde a las solicitudes de tarjeta de crédito realizadas durante los años 2016 al 2019 con periodicidad diaria. Cabe aclarar que no es posible revelar el nombre de la empresa por motivos de confidencialidad. La base de datos cuenta con 165082 registros y 149 variables, de las cuales hay 53 variables con datos nulos del 100 % y 20 variables con datos nulos que superan el 90 %. Por lo anterior, es importante hacer un riguroso tratamiento de datos con el fin de eliminar tanto variables con alto porcentaje de nulos como variables que no dan indicio de que exista fraude y así, poder contar con una "sábana" de datos limpia con la cual se puedan crear las estimaciones correspondientes.

3.1.1. Eliminación de variables no representativas

Se realiza una exploración minuciosa de los datos con el fin de seleccionar una base de datos idónea que garantice un buen ajuste del modelo, en donde se identifica que aproximadamente el 49 % de los datos corresponden a valores nulos y el 41 % corresponde a variables no concluyentes para el resultado del modelo. Para dar cumplimiento a lo anterior, inicialmente se eliminan 73 variables como: descripción, varPrendaAFavor, varLimitacion1, varLimitacion3, varMesUtlFactura, intNumHijos, intNroVehiculos, floArrendamiento; con un alto porcentaje de valores nulos y 61 variables que no son importantes a la hora de detectar fraude como los contactos y las referencias del solicitante entre las que se encuentran: varNomConyugue, varCelConyugue, varCelConyugue, varTelefono, varCelular, varNombreRef1, varNombreRef2, varTelefonoRef2, varCelularRefProv1. También se eliminan las variables intEdad y Estado, ya que se encuentran duplicadas.

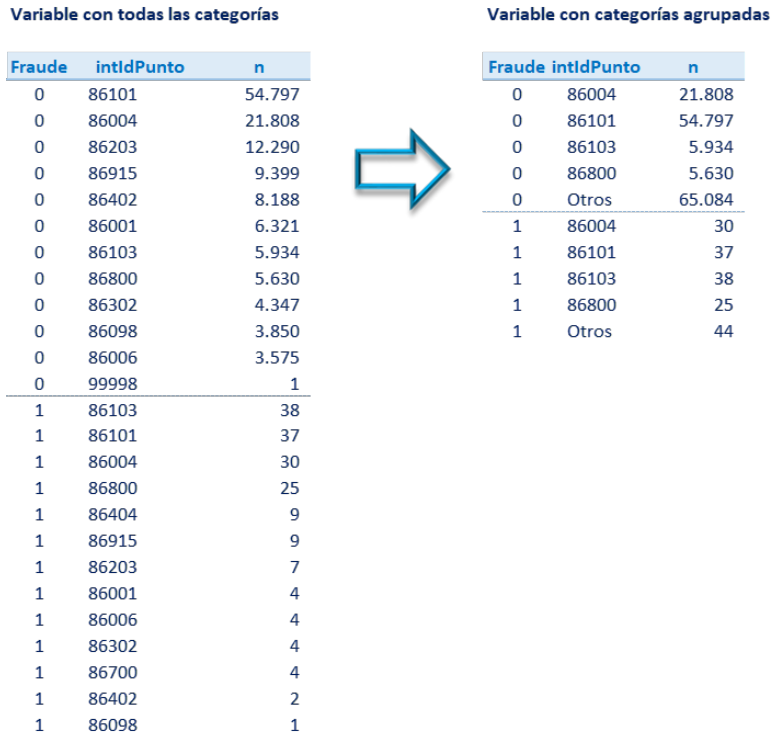
3.1.2. Limpieza de datos

Con una base de datos reducida en variables, se garantiza la buena calidad de los datos que faciliten una correcta manipulación, visualización y modelación de la información. La limpieza de datos consiste en eliminar o corregir registros anómalos, datos ausentes, valores NA, identificar y sustituir datos o registros incompletos y conversión de tipos de variables, entre otros. Es importante para este proceso hacer uso de una aplicación especializada tanto a nivel estadístico como en el procesamiento de datos, en este caso se utilizó el software RStudio ya que se trata de una interfaz muy cómoda, actualmente muy utilizada por estudiantes y personas dedicadas a la parte estadística. (CRC, s.f.)

3.1.3. Recategorización de la información

Realizando un análisis bivariado de los datos entre la variable dependiente fraude y las demás covariables, se puede observar que variables como: varMedio, intIdPunto, intIdCanal, intMesFecSolicitud, varEstCivil, varEstrato, varCodOcupacion, varNivEstudio, EstadoReal, Edad y PersonasACargo; no cuentan con información suficiente en algunas categorías. La descripción de estas variables se presentará en la siguiente sección. A continuación, se relaciona la variable varMedio con su respectiva recategorización (las demás variables se pueden visualizar en el anexo I.

- **varMedio:**



Variable con todas las categorías

Fraude	intIdPunto	n
0	86101	54.797
0	86004	21.808
0	86203	12.290
0	86915	9.399
0	86402	8.188
0	86001	6.321
0	86103	5.934
0	86800	5.630
0	86302	4.347
0	86098	3.850
0	86006	3.575
0	99998	1
1	86103	38
1	86101	37
1	86004	30
1	86800	25
1	86404	9
1	86915	9
1	86203	7
1	86001	4
1	86006	4
1	86302	4
1	86700	4
1	86402	2
1	86098	1

Variable con categorías agrupadas

Fraude	intIdPunto	n
0	86004	21.808
0	86101	54.797
0	86103	5.934
0	86800	5.630
0	Otros	65.084
1	86004	30
1	86101	37
1	86103	38
1	86800	25
1	Otros	44

Figura 3-1: Recategorización de la variable varMedio

Como se muestra en la Figura **3-1**, para la variable varMedio las categorías Asesor y Referido son las más representativas y las categorías restantes se agrupan en la categoría Otros.

3.2. Análisis descriptivo de los datos

Con el fin de identificar características particulares en los datos y partiendo de que en este paso se cuenta con una base de datos de óptima calidad, es importante realizar un análisis descriptivo con las variables relevantes.
A continuación se hará una descripción de las variables relevantes que hacen parte de la base de datos de solicitudes de tarjeta de crédito en una empresa particular:

- **Fraude:** indica el tipo de transacción realizada por un cliente:
  - 0: Indica una transacción de tipo normal
  - 1: Indica una transacción de tipo fraudulenta

Categoría	n	%
No Fraude	153.253	0,999
Fraude	174	0,001

Tabla 3-1: Frecuencia observada para la variable fraude

En la Tabla **3-1** se puede observar que esta variable presenta un desbalance muy marcado en la categoría 0 correspondiente a transacciones de tipo no fraudulenta, frente a la categoría 1 que corresponde a transacciones fraudulentas.

- **varMedio:** corresponde al medio a través del cual el cliente adquirió el producto.

varMedio	n	%
Asesor	126.549	0,825
Otros	11.168	0,073
Referido	15.710	0,102

Tabla 3-2: Frecuencia observada para la variable varMedio

En la Tabla **3-2** se puede observar que el medio por el cual mas clientes adquirieron el producto fue por el Asesor.

- **intIdPunto:** La variable intIdPunto corresponde al lugar donde se generó la solicitud, la empresa cuenta con 138 puntos de venta aliados disponibles.

intIdPunto	n	%
86004	21.838	0,142
86101	54.834	0,357
86103	5.972	0,039
86800	5.655	0,037
Otros	65.128	0,424

Tabla 3-3: Frecuencia observada para la variable intIdPunto

En la Tabla **3-3** se muestran los puntos de venta más significativos como 86004 que corresponde al ÉXITO SAN ANTONIO, 86101 corresponde a la OFICINA EDIFICIO EPM MULTI, 86103 corresponde al ÉXITO COLOMBIA, 86800 corresponde a HOMECENTER SAN JUAN y Otros agrupa al resto de los puntos.

- **intIdCanal:** esta variable indica el medio a través del cual se solicitó el producto, en un punto puede haber varios canales. En la actualidad la empresa cuenta con 149 canales de ventas.

intIdCanal	n	%
1043	30.048	0,196
1045	100.024	0,652
Otros	23.355	0,152

Tabla 3-4: Frecuencia observada para la variable intIdCanal

En la Tabla **3-4** se muestran los canales de venta más utilizados por los clientes, en este caso se tiene el canal 1045 correspondiente a la fuerza de ventas directa - EFICACIA y el 1043 correspondiente a CONTACT CENTER y por ultimo el canal de venta Otros.

- **intMesFecSolicitud:** la variable intMesFecSolicitud hace referencia al trimestre en el cual se realizó la solitud.

intMesFecSolicitud	n	%
Trim_1	36.721	0,239
Trim_2	38.783	0,253
Trim_3	39.088	0,255
Trim_4	38.835	0,253

Tabla 3-5: Frecuencia observada para la variable intMesFecSolicitud

- **varGenero:** define el tipo de persona que realizó la solicitud en este caso hombre o mujer.

varGenero	n	%
Hombre	62.304	0,406
Mujer	91.123	0,594

Tabla 3-6: Frecuencia observada para la variable varGenero

En la Tabla **3-6** se puede observar que las mujeres registran más solicitudes que los hombres.

- **varEstCivil:** indica la situación marital del solicitante al momento de requerir la tarjeta de crédito, en la categoría Otros se agrupan las categorías Divorciado y Viudo.

varEstCivil	n	%
Casado	43.590	0,284
Otros	16.726	0,109
Soltero	57.867	0,377
Unionlibre	35.244	0,230

Tabla 3-7: Frecuencia observada para la variable varEstCivil

En la Tabla **3-7** se puede observar que un alto número de solicitantes son solteros, seguido de solicitantes casados.

- **varTipoVivienda:** se refiere a si la vivienda del solicitante es propia, arrendada o familiar.

varTipoVivienda	n	%
Arrendadda	30.255	0,197
Familiar	59.935	0,391
Propia	63.237	0,412

Tabla 3-8: Frecuencia observada para la variable varTipoVivienda

En la Tabla **3-8** se puede resaltar que la vivienda que predomina entre los solicitantes es de carácter propia.

- **varEstrato:** esta variable define la estratificación socio económica que tiene la empresa para identificar si la población es residencial categoría entre 1 y 6 o no residencial diferente a 1 o 6.

varEstrato	n	%
1	22.020	0,144
2	65.681	0,428
3	50.580	0,330
4	10.237	0,067
5-6	4.909	0,032

Tabla 3-9: Frecuencia observada para la variable varEstrato

En la Tabla **3-9** se muestra que un alto número de solicitantes se encuentran en el estrato 2, también se puede resaltar que todos los solicitantes se encuentran en la categoría residencial que hace referencia a los estratos entre 1 y 6.

- **varTipoInmueble:** esta variable identifica si un inmueble es urbano: centros poblados con 2500 o más habitantes definida en cualquier nivel de la División Político Territorial del país o rural: personas que viven fuera de las áreas definidas como urbanas, en lo que se denomina periferia urbana.

varTipoInmueble	n	%
Rural	9.644	0,063
Urbano	143.783	0,937

Tabla 3-10: Frecuencia observada para la variable varTipoInmueble

En la Tabla **3-10** se resalta que la mayor parte de los solicitantes son de tipo de inmueble urbano.

- **varCodOcupacion:** identifica la clase o tipo de trabajo desarrollado por la persona al momento de hacer la solicitud, en la categoría Otros se agrupan: Ama de Casa y Pensionado.

varCodOcupacion	n	%
Empleado	88.650	0,578
Independiente	25.452	0,166
Otros	39.325	0,256

Tabla 3-11: Frecuencia observada para la variable varTipoInmueble

En la Tabla **3-11** se identifica que un alto número de solicitantes tienen empleo.

- **varNivEstudio:** esta variable identifica el grado de aprendizaje que adquirió el solicitante a lo largo de su formación en una Institución educativa, en la categoría Otros se agrupan: Primaria y Tecnólogo.

varNivEstudio	n	%
Ninguno	1.100	0,007
Otros	42.108	0,274
Secundaria	65.510	0,427
Técnico	26.612	0,173
Universitario	18.097	0,118

Tabla 3-12: Frecuencia observada para la variable varNivEstudio

En la Tabla **3-12** se identifica que un alto número de solicitantes cuentan con un nivel de estudio de secundaria.

- **EstadoReal:** identifica las diferentes etapas de aprobación o rechazo de la solicitud, en la categoría otros se encuentran agrupadas: Pendiente Activar Redeban, Pendiente Confronta, Pendiente Creación Cliente y Tarjeta Reservada.

EstadoReal	n	%
Tarjeta Activada	88.633	0.578
Habilitado Para Nuevo Estudio	64.598	0.421
Otros	196	0.001

Tabla 3-13: Frecuencia observada para la variable EstadoReal

En la Tabla **3-13** se puede identificar que tarjeta activada es el estado más alto.

- **Edad:** tiempo que ha vivido el solicitante desde su nacimiento.

Edad	n	%
> 63	13.939	0,091
18-22	11.660	0,076
23-27	18.531	0,121
28-32	18.747	0,122
33-37	18.464	0,120
38-42	16.507	0,108
43-47	14.415	0,094
48-52	15.782	0,103
53-57	14.519	0,095
58-62	10.863	0,071

Tabla 3-14: Frecuencia observada para la variable Edad

- **PersonasACargo:** identifica el número de personas dependientes a cargo del solicitante.

PersonasACargo	n	%
>=2	29.012	0,189
0	80.656	0,526
1	43.759	0,285

Tabla 3-15: Frecuencia observada para la variable PersonasACargo

En la Tabla **3-15** muestra que la mayoría de los solicitantes no tienen personas dependientes a cargo.

3.3. Aplicación de la metodología de selección de variables

La base de datos depurada cuenta con las siguientes variables: varMedio, intIdPunto, intIdCanal, intMesFecSolicitud, varGenero, varEstCivil, varTipoVivienda, varEstrato, varTipoInmueble, varCodOcupacion, varNivEstudio, EstadoReal, Edad, PersonasACargo y Fraude, pero se desconoce cuáles variables son importantes o no para el modelo, por esta razón se aplican las metodologías de selección de variables para definir el mejor subconjunto de variables predictoras que expliquen si existe fraude o no, las metodologías aplicadas son:

- **Selección hacia adelante (forward)**: para la aplicación de esta metodología se utiliza la función regsubsets con el método forward. El resultado se puede observar en la Figura **3-2** en cada gráfica los diferentes estadísticos, los cuales muestran mejores resultados a medida que se agregan variables predictivas de tal forma que se puede deducir que el mejor modelo incluye las 15 variables.

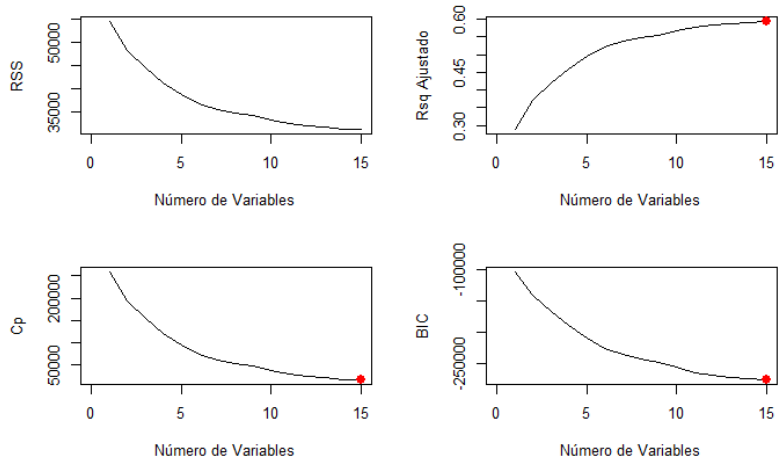


Figura 3-2: Selección hacia adelante (forward)

- **Selección hacia atrás (backward)**: para la aplicación de esta metodología se utiliza la función regsubsets con el método backward. En la Figura **3-3** se puede observar en cada gráfica los diferentes estadísticos, al igual que en la metodología de selección forward se puede deducir que el mejor modelo incluye las 15 variables.

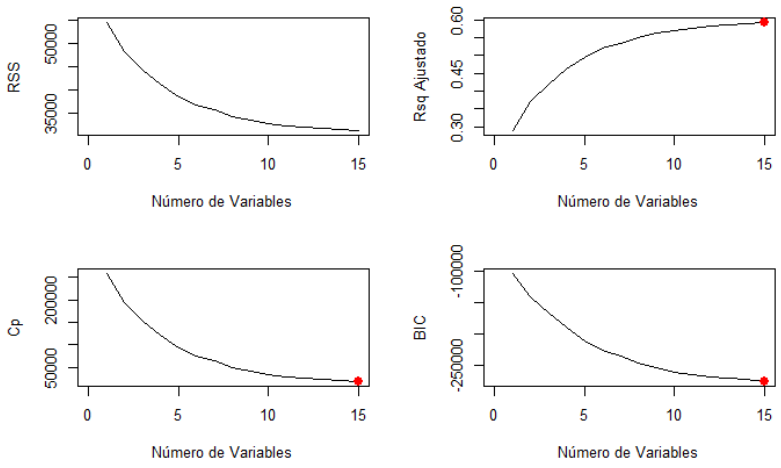


Figura 3-3: Selección hacia atrás (backward)

- **Stepwite**: para la aplicación de esta metodología se generan dos modelos: uno sin incluir las variables independientes y el otro incluyendo todas las variables independientes, una vez se cuenta con los dos modelos, se utiliza la función step() de paquete stats, el proceso comienza con el modelo sin variables y después toma una a una cada variable del modelo completo, agregándola a un nuevo modelo, si estadísticamente es significativa. Los resultados se puede observar en la Figura **3-4** todas las variables son significativas, ya que presentan un $p - valor \leq 0,05$, excepto la variable varNivEstudio que presentan un $p - valor \geq 0,05$. Para este trabajo se tendrán en cuenta las 15 variables disponibles, teniendo en cuenta que las metodologías de selección hacia adelante (forward) y selección hacia atrás (backward), muestran que las 15 variables son importantes para el modelo.

	Estimate	Std.Error	z value	value	Pr(>|z|)	exp(Estimate)
(Intercept)	-19,460	495,515	-0,039	0,969		3,5E-09
varMedioOtros	-0,623	0,319	-1,956	0,050		0,536
varMedioReferido	-0,420	0,300	-1,400	0,162		0,657
intIdPunto86101	-3,048	0,349	-8,738	< 2E-16	***	0,047
intIdPunto86103	0,883	0,316	2,795	0,005	**	2,418
intIdPunto86800	0,459	0,361	1,270	0,204		1,583
intIdPuntoOtros	-0,890	0,319	-2,788	0,005	**	0,410
intIdCanal1045	-2,362	0,297	-7,957	2E-15	***	0,094
intIdCanalOtros	-3,298	0,460	-7,171	7E-13	***	0,037
ntMesFecSolicitudTrim_2	1,126	0,393	2,867	0,004	**	3,082
ntMesFecSolicitudTrim_3	0,841	0,399	2,108	0,035	*	2,318
ntMesFecSolicitudTrim_4	1,285	0,373	3,449	0,001	***	3,616
varGeneroMujer	-1,513	0,219	-6,912	5E-12	***	0,220
varEstCivilOtros	1,275	0,460	2,771	0,006	**	3,578
varEstCivilSoltero	1,620	0,362	4,479	7E-06	***	5,052
varEstCivilUnionLibre	0,609	0,437	1,393	0,164		1,839
varTipoViviendaFamiliar	1,567	0,377	4,150	3E-05	***	4,790
varTipoViviendaPropia	0,431	0,442	0,975	0,330		1,539
varEstrato2	-0,625	0,360	-1,736	0,082		0,535
varEstrato3	-0,105	0,348	-0,303	0,762		0,900
varEstrato4	0,366	0,392	0,935	0,350		1,442
varEstrato5_6	-0,596	0,516	-1,155	0,248		0,551
varTipoInmuebleUrbano	0,615	0,603	1,021	0,307		1,850
OcupacionIndependiente	2,120	0,227	9,325	< 2E-16	***	8,333
varCodOcupacionOtros	0,285	0,439	0,650	0,516		1,330
varNivEstudioOtros	12,283	495,514	0,025	0,980		2,2E+05
varNivEstudioSecundaria	12,498	495,514	0,025	0,980		2,7E+05
varNivEstudioTecnico	12,486	495,514	0,025	0,980		2,6E+05
arNivEstudioUniversitario	13,696	495,514	0,028	0,978		8,9E+05
EstadoRealOtros	-13,733	1216,367	-0,011	0,991		1,1E-06
tadoRealTarjetaActivada	-2,410	0,251	-9,598	< 2E-16	***	0,090
Edad18-22	-1,542	0,716	-2,152	0,031	*	0,214
Edad23-27	-0,955	0,604	-1,583	0,113		0,385
Edad28-32	0,129	0,539	0,239	0,811		1,138
Edad33-37	0,409	0,535	0,764	0,445		1,505
Edad38-42	0,523	0,531	0,985	0,325		1,687
Edad43-47	0,519	0,539	0,964	0,335		1,681
Edad48-52	-0,557	0,606	-0,920	0,358		0,573
Edad53-57	-0,028	0,551	-0,051	0,959		0,972
Edad58-62	-0,517	0,632	-0,819	0,413		0,596
PersonasACargo0	0,617	0,323	1,910	0,056		1,853
PersonasACargo1	-0,584	0,396	-1,473	0,141		0,558

Figura 3-4: Nivel de significancia para el método de selección de variables (Stepwite)

3.4. Aplicación de la metodología para datos desbalanceados

En los casos de fraude es muy común que se encuentren datos desequilibrados, en donde la cantidad de observaciones de una clase es significativamente mayor que las demás clases. La base de datos actualmente presenta un problema de desbalanceo de información; como se puede observar en la Tabla **3-16**, la clase minoritaria (Fraudes) representan tan solo el 0.10 % del total de los casos, frente a la clase mayoritaria (No Fraude) que representa el 99,90 %. Lo anterior puede ocasionar que el algoritmo tenga problemas en la predicción al tener en cuenta la clase mayoritaria y de esta manera se tengan conclusiones erróneas en la medición del rendimiento de las métricas al proporcionar una precisión de clasificación engañosa.

No Fraude	Fraude
153.253	174
99,9 %	0,10 %

Tabla 3-16: Desbalance de la variable Fraude

Teniendo en cuenta lo anterior es importante aplicar metodologías que ayuden a corregir el tema de desbalanceo de datos para evitar el sobre costo que representa un problema de clasificación como este.

3.4.1. Undersampling

Esta metodología busca mantener la clase minoritaria disminuyendo la muestra de la clase mayoritaria, es decir, iguala las clases teniendo en cuenta la clase minoritaria, tal como se muestra en la Tabla **3-17**, para la aplicación de esta metodología se utilizo la función downSample del paquete Caret.

No Fraude	Fraude	Total
174	174	348

Tabla 3-17: Metodología Undersampling

3.4.2. Oversampling

Esta metodología busca mantener la clase mayoritaria, aumentando la muestra de la clase minoritaria, es decir iguala las clases teniendo en cuenta la clase mayoritaria, tal como se muestra en la Tabla **3-18**, para la aplicación de esta metodología se utilizo la función upSample del paquete Caret.

No Fraude	Fraude	Total
153.253	153.253	306.506

Tabla 3-18: Metodología Oversampling

BASE COMPLETA	TAMAÑOS		
	0	1	Total
Base de datos total	153.253	174	153.427
Balanceo Oversampling	153.253	153.253	306.506
Balanceo Undersmpling	174	174	348

BASE DE ENTRENAMIENTO	TAMAÑOS		
	0	1	Total
Base de datos total	122.602	139	122.742
Balanceo Oversampling	122.602	122.602	245.205
Balanceo Undersmpling	139	139	278

BASE DE PRUEBA	TAMAÑOS		
	0	1	Total
Base de datos total	30.651	35	30.685
Balanceo Oversampling	30.651	30.651	61.301
Balanceo Undersmpling	35	35	70

Figura 3-5: Registros por clases con las metodologías aplicadas (Fuente: elaboración propia)

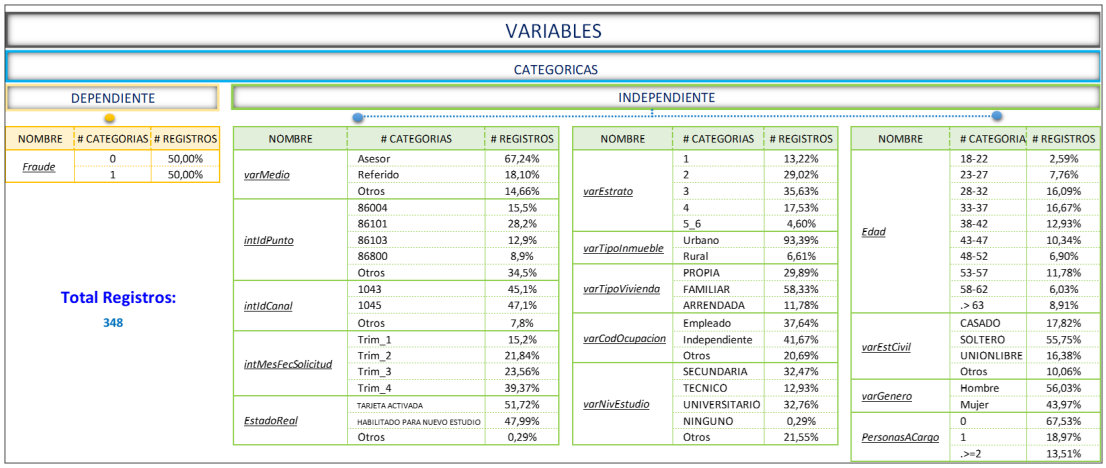
En la Figura **3-5** se puede ver el resumen de los datos con las metodologías aplicadas anteriormente.

3.5. Partición de la base de datos

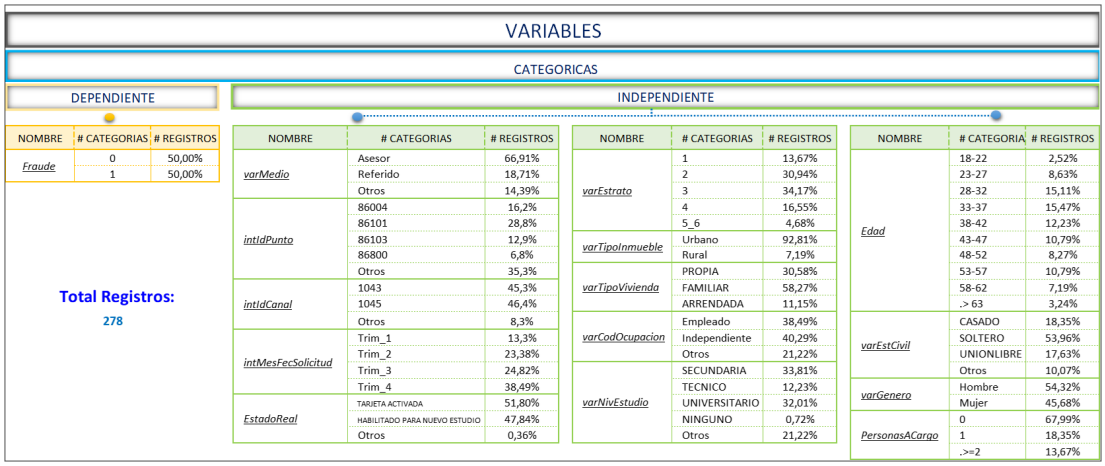
Figura 3-6: : Partición de la base de datos (Fuente: elaboración propia)

Para comprobar la capacidad predictiva de un modelo, es importante poner a prueba un conjunto que no esté comprometido en el ajuste, pero del que se pueda conocer la variable respuesta para identificar qué tan exactas son sus predicciones. En este sentido, una posibilidad consiste en aplicar validación cruzada dividiendo los datos en un conjunto de entrenamiento equivalente al 80 % y un conjunto de prueba equivalente al 20 %, tal como se ilustra en la Figura **3-6**. Dicha partición debe garantizar una estratificación que sea capaz de obtener una estructura muestral similar al total de los datos, tanto en el conjunto de entrenamiento como en el conjunto de prueba. Según Yadav (2021), el muestreo estratificado se utiliza a menudo cuando uno o más de los estratos de la población tienen una incidencia baja en relación con los otros estratos, este caso se presenta en la base de datos utilizada al estar altamente desbalanceada alguna de sus categorías, no solo de la variable de interés, sino de las demás variables categóricas que serán consideradas como variables explicativas o covariables en los modelos a ser aplicados.

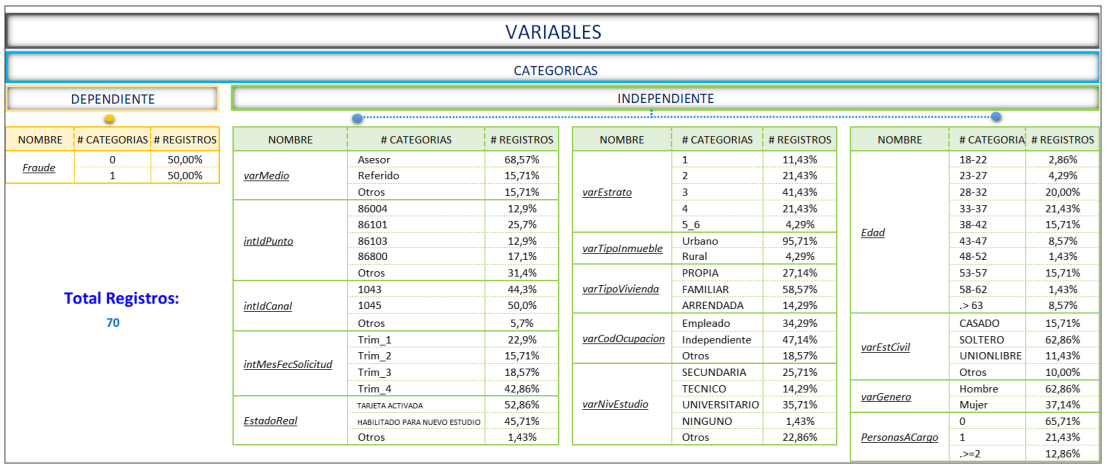
A continuación, se muestra la distribución de las variables con cada una de sus categorías para las bases de datos balanceadas bajo las metodologías Undersampling y Oversampling. Para cada una de estas se aplica la partición de los datos en un conjunto de entrenamiento equivalente al 80 % y un conjunto de prueba equivalente al 20 %.



(a) Base de datos completa.

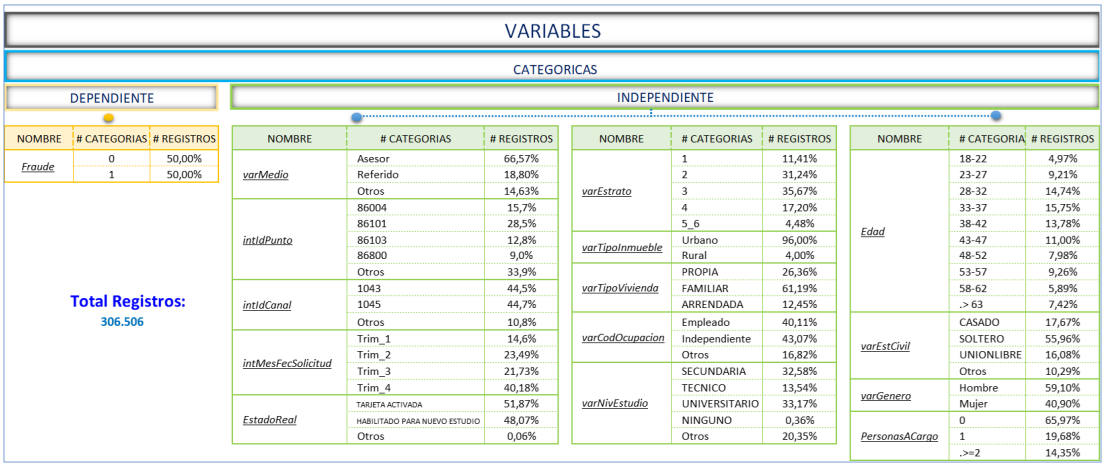


(b) Datos de entrenamiento 80 %.



(c) Datos del testing 20 %.

Figura 3-7: Partición datos balanceados Undersampling. Fuente: elaboración propia.



VARIABLES

CATEGORICAS

DEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
Fraude	0	50,00%
	1	50,00%

Total Registros: 306.506

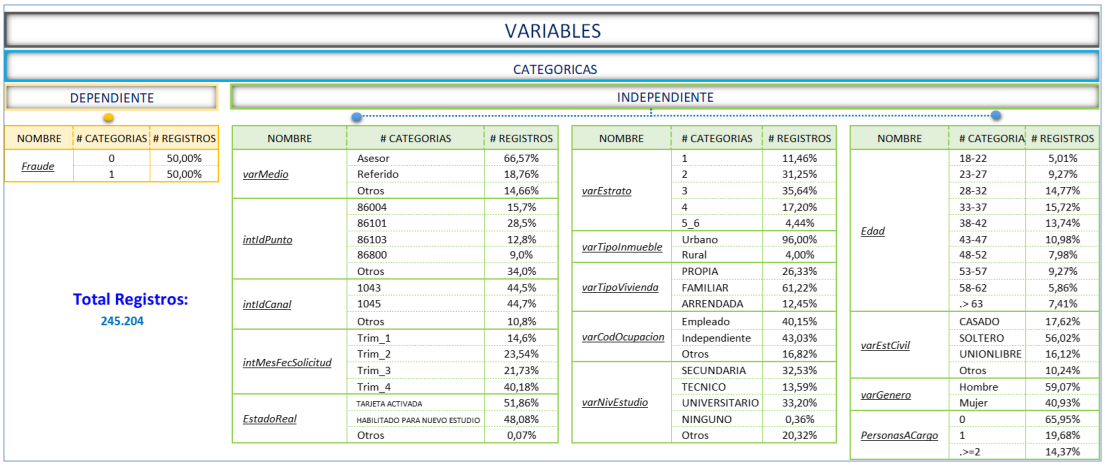
INDEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
varMedio	Asesor	66,57%
	Referido	18,80%
	Otros	14,63%
intIdPunto	86004	15,7%
	86101	28,5%
	86103	12,8%
	86800	9,0%
	Otros	33,9%
intIdCanal	1043	44,5%
	1045	44,7%
	Otros	10,8%
intMesFecSolicitud	Trim_1	14,6%
	Trim_2	23,49%
	Trim_3	21,73%
	Trim_4	40,18%
EstadoReal	TARJETA ACTIVADA	51,87%
	HABILITADO PARA NUEVO ESTUDIO	48,07%
	Otros	0,06%

NOMBRE	# CATEGORIAS	# REGISTROS
varEstrato	1	11,41%
	2	31,24%
	3	35,67%
	4	17,20%
	5_6	4,48%
varTipoInmueble	Urbano	96,00%
	Rural	4,00%
varTipoVivienda	PROPIA	26,36%
	FAMILIAR	61,19%
	ARRENDADA	12,45%
varCodOcupacion	Empleado	40,11%
	Independiente	43,07%
	Otros	16,82%
varNivEstudio	SECUNDARIA	32,58%
	TECNICO	13,54%
	UNIVERSITARIO	33,17%
	NINGUNO	0,36%
	Otros	20,35%

NOMBRE	# CATEGORIA	# REGISTROS
Edad	18-22	4,97%
	23-27	9,21%
	28-32	14,74%
	33-37	15,75%
	38-42	13,78%
	43-47	11,00%
	48-52	7,98%
	53-57	9,26%
	58-62	5,89%
	.> 63	7,42%
varEstCivil	CASADO	17,67%
	SOLTERO	55,96%
	UNIONLIBRE	16,08%
	Otros	10,29%
varGenero	Hombre	59,10%
	Mujer	40,90%
PersonasACargo	0	65,97%
	1	19,68%
	.>=2	14,35%

(a) Base de datos completa.



VARIABLES

CATEGORICAS

DEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
Fraude	0	50,00%
	1	50,00%

Total Registros: 245.204

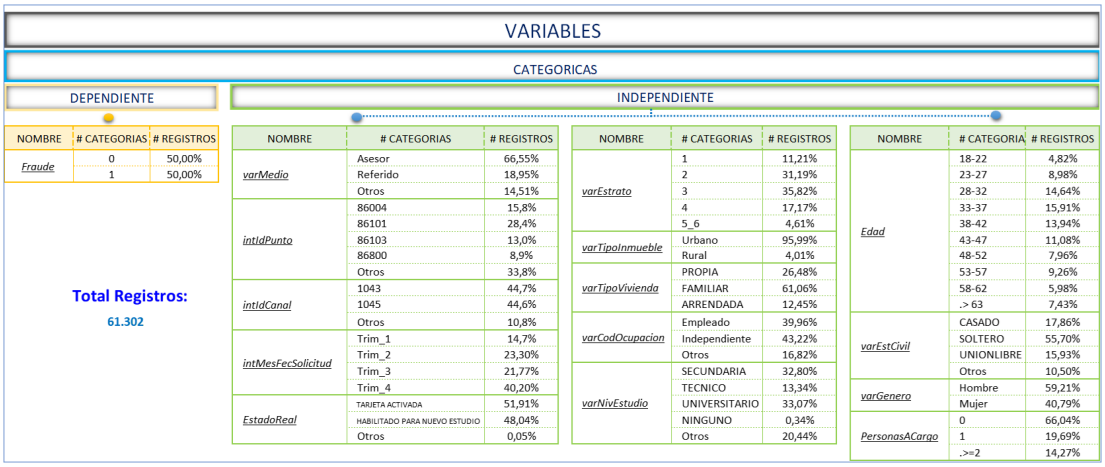
INDEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
varMedio	Asesor	66,57%
	Referido	18,76%
	Otros	14,66%
intIdPunto	86004	15,7%
	86101	28,5%
	86103	12,8%
	86800	9,0%
	Otros	34,0%
intIdCanal	1043	44,5%
	1045	44,7%
	Otros	10,8%
intMesFecSolicitud	Trim_1	14,6%
	Trim_2	23,54%
	Trim_3	21,73%
	Trim_4	40,18%
EstadoReal	TARJETA ACTIVADA	51,86%
	HABILITADO PARA NUEVO ESTUDIO	48,08%
	Otros	0,07%

NOMBRE	# CATEGORIAS	# REGISTROS
varEstrato	1	11,46%
	2	31,25%
	3	35,64%
	4	17,20%
	5_6	4,44%
varTipoInmueble	Urbano	96,00%
	Rural	4,00%
varTipoVivienda	PROPIA	26,33%
	FAMILIAR	61,22%
	ARRENDADA	12,45%
varCodOcupacion	Empleado	40,15%
	Independiente	43,03%
	Otros	16,82%
varNivEstudio	SECUNDARIA	32,53%
	TECNICO	13,59%
	UNIVERSITARIO	33,20%
	NINGUNO	0,36%
	Otros	20,32%

NOMBRE	# CATEGORIA	# REGISTROS
Edad	18-22	5,01%
	23-27	9,27%
	28-32	14,77%
	33-37	15,72%
	38-42	13,74%
	43-47	10,98%
	48-52	7,98%
	53-57	9,27%
	58-62	5,86%
	.> 63	7,41%
varEstCivil	CASADO	17,62%
	SOLTERO	56,02%
	UNIONLIBRE	16,12%
	Otros	10,24%
varGenero	Hombre	59,07%
	Mujer	40,93%
PersonasACargo	0	65,95%
	1	19,68%
	.>=2	14,37%

(b) Datos de entrenamiento 80 %.



VARIABLES

CATEGORICAS

DEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
Fraude	0	50,00%
	1	50,00%

Total Registros: 61.302

INDEPENDIENTE

NOMBRE	# CATEGORIAS	# REGISTROS
varMedio	Asesor	66,55%
	Referido	18,95%
	Otros	14,51%
intIdPunto	86004	15,8%
	86101	28,4%
	86103	13,0%
	86800	8,9%
	Otros	33,8%
intIdCanal	1043	44,7%
	1045	44,6%
	Otros	10,8%
intMesFecSolicitud	Trim_1	14,7%
	Trim_2	23,30%
	Trim_3	21,77%
	Trim_4	40,20%
EstadoReal	TARJETA ACTIVADA	51,91%
	HABILITADO PARA NUEVO ESTUDIO	48,04%
	Otros	0,05%

NOMBRE	# CATEGORIAS	# REGISTROS
varEstrato	1	11,21%
	2	31,19%
	3	35,82%
	4	17,17%
	5_6	4,61%
varTipoInmueble	Urbano	95,99%
	Rural	4,01%
varTipoVivienda	PROPIA	26,48%
	FAMILIAR	61,06%
	ARRENDADA	12,45%
varCodOcupacion	Empleado	39,96%
	Independiente	43,22%
	Otros	16,82%
varNivEstudio	SECUNDARIA	32,80%
	TECNICO	13,34%
	UNIVERSITARIO	33,07%
	NINGUNO	0,34%
	Otros	20,44%

NOMBRE	# CATEGORIA	# REGISTROS
Edad	18-22	4,82%
	23-27	8,98%
	28-32	14,64%
	33-37	15,91%
	38-42	13,94%
	43-47	11,08%
	48-52	7,96%
	53-57	9,26%
	58-62	5,98%
	.> 63	7,43%
varEstCivil	CASADO	17,86%
	SOLTERO	55,70%
	UNIONLIBRE	15,93%
	Otros	10,50%
varGenero	Hombre	59,21%
	Mujer	40,79%
PersonasACargo	0	66,04%
	1	19,69%
	.>=2	14,27%

(c) Datos del testing 20 %.

Figura 3-8: Partición datos balanceados Oversampling. Fuente: elaboración propia.

Como se puede observar en las Figuras **3-7** y **3-8**, el porcentaje de distribución en cada una de las clases de las variables es similar entre las particiones y la base de datos completa, lo que garantiza una adecuada estratificación en las bases de datos balanceadas tanto con undersampling como por oversampling.

3.6. Aplicación de modelos de clasificación para la detección de fraude

A continuación se presentan los resultados obtenidos luego de aplicar los modelos que fueron expuestos en el Capítulo 2:

3.6.1. Modelo logístico con datos desbalanceados

Para exponer la desventaja que lleva el hecho de tener una base de datos cuya variable respuesta es una variable dicotómica con un alto desbalance entre las categorías, aplicaremos el modelo logístico a la base de datos completa. Una vez se cuenta con la base de datos depurada, organizada y con las variables predictoras que salieron significativas con los métodos de selección, se ejecuta un primer modelo logístico, haciendo uso de la función glm() con el parámetro family, teniendo en cuenta que se trata de un modelo binomial. El modelo se realiza con los datos de entrenamiento correspondiente al 80 % y se valida los resultados con los datos de prueba equivalentes al 20 %.

	Estimate	Std.Error	z value	value	Pr(>|z|)	exp(Estimate)
(Intercept)	-19,460	495,515	-0,039	0,969		3,5E-09
varMedioOtros	-0,623	0,319	-1,956	0,050		0,536
varMedioReferido	-0,420	0,300	-1,400	0,162		0,657
intIdPunto86101	-3,048	0,349	-8,738	< 2E-16	***	0,047
intIdPunto86103	0,883	0,316	2,795	0,005	**	2,418
intIdPunto86800	0,459	0,361	1,270	0,204		1,583
intIdPuntoOtros	-0,890	0,319	-2,788	0,005	**	0,410
intIdCanal1045	-2,362	0,297	-7,957	2E-15	***	0,094
intIdCanalOtros	-3,298	0,460	-7,171	7E-13	***	0,037
ntMesFecSolicitudTrim_2	1,126	0,393	2,867	0,004	**	3,082
ntMesFecSolicitudTrim_3	0,841	0,399	2,108	0,035	*	2,318
ntMesFecSolicitudTrim_4	1,285	0,373	3,449	0,001	***	3,616
varGeneroMujer	-1,513	0,219	-6,912	5E-12	***	0,220
varEstCivilOtros	1,275	0,460	2,771	0,006	**	3,578
varEstCivilSoltero	1,620	0,362	4,479	7E-06	***	5,052
varEstCivilUnionLibre	0,609	0,437	1,393	0,164		1,839
varTipoViviendaFamiliar	1,567	0,377	4,150	3E-05	***	4,790
varTipoViviendaPropia	0,431	0,442	0,975	0,330		1,539
varEstrato2	-0,625	0,360	-1,736	0,082		0,535
varEstrato3	-0,105	0,348	-0,303	0,762		0,900
varEstrato4	0,366	0,392	0,935	0,350		1,442
varEstrato5_6	-0,596	0,516	-1,155	0,248		0,551
varTipoInmuebleUrbano	0,615	0,603	1,021	0,307		1,850
OcupacionIndependiente	2,120	0,227	9,325	< 2E-16	***	8,333
varCodOcupacionOtros	0,285	0,439	0,650	0,516		1,330
varNivEstudioOtros	12,283	495,514	0,025	0,980		2,2E+05
varNivEstudioSecundaria	12,498	495,514	0,025	0,980		2,7E+05
varNivEstudioTecnico	12,486	495,514	0,025	0,980		2,6E+05
arNivEstudioUniversitario	13,696	495,514	0,028	0,978		8,9E+05
EstadoRealOtros	-13,733	1216,367	-0,011	0,991		1,1E-06
tadoRealTarjetaActivada	-2,410	0,251	-9,598	< 2E-16	***	0,090
Edad18-22	-1,542	0,716	-2,152	0,031	*	0,214
Edad23-27	-0,955	0,604	-1,583	0,113		0,385
Edad28-32	0,129	0,539	0,239	0,811		1,138
Edad33-37	0,409	0,535	0,764	0,445		1,505
Edad38-42	0,523	0,531	0,985	0,325		1,687
Edad43-47	0,519	0,539	0,964	0,335		1,681
Edad48-52	-0,557	0,606	-0,920	0,358		0,573
Edad53-57	-0,028	0,551	-0,051	0,959		0,972
Edad58-62	-0,517	0,632	-0,819	0,413		0,596
PersonasACargo0	0,617	0,323	1,910	0,056		1,853
PersonasACargo1	-0,584	0,396	-1,473	0,141		0,558

Figura 3-9: Nivel de significancia para el modelo logístico sin balanceo de datos

En la Figura **3-9** se pueden observar las variables significativas con sus respectivos coeficientes, en donde se puede constatar según la prueba de significancia un p-valor menor o igual a 0,05, lo que garantiza que existen algunas variables como: **varGenero**, **varTipoVivienda**, **varCodOcupacion**, **intIdPunto**,**intIdCanal**,**intMesFecSolicitud**,y **EstadoReal**, que son altamente significativas. Con respecto al signo se debe tener en cuenta que si este es negativo quiere decir que es una variable que premia y si es positivo es una variable que castiga.

Determinación del punto de corte óptimo y curva ROC del modelo logístico sin balanceo de datos

Como el objetivo es maximizar tanto la sensibilidad como la especificidad, (Hosmer, Le-

meshow, y Sturdivant, 2013) sugiere la importancia de encontrar un punto de corte óptimo. En la Figura **3-10** se puede observar que el punto óptimo para el modelo logistico sin balanceo de datos es de 15,56 % con un área bajo la curva (AUC) de 94,96 %.

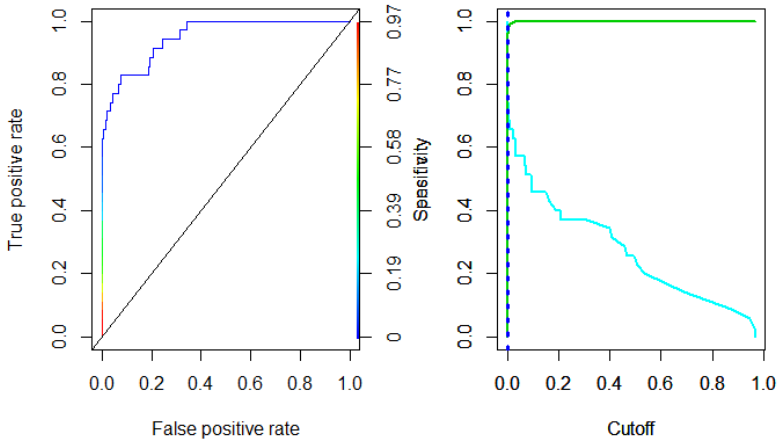


Figura 3-10: Punto de corte óptimo y curva ROC del modelo logístico sin balanceo de datos

Al aplicar el punto de corte óptimo de 15,56 %, se puede observar en la Tabla **3-19** el resultado de una la matriz de confusión más ajustada y con mejores medidas de desempeño. El modelo presenta un "accuracy" o precisión muy alta de 92,48 %; sin embargo, se debe de tener cuidado, ya que las clases están tan desbalanceadas que se puede llegar a una falsa conclusión de que el modelo es bueno. Dicho fenómeno se conoce como *la Paradoja del Accuracy* y es mencionado por algunos autores como Martínez (2019). Este indicador resulta provechoso solo cuando el costo de un falso positivo es igual que el de un falso negativo, por lo tanto, este no es un buen elemento de medición en clases desbalanceadas y se puede apoyar en indicadores más realistas como la **Sensibilidad** que es la probabilidad de que el resultado de fraude sea positivo si realmente es un fraude y **Especificidad** que es la probabilidad de que el resultado de fraude sea negativo si realmente no es un fraude, estos indicadores muestran la capacidad del estimador para diferenciar los casos positivos de los negativos. Tal como se muestra en la Tabla **3-20**, para este modelo la sensibilidad es del 80,00 % y la especificidad del 92,49 %.

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	28.352	7
	fraude	2.299	28

Tabla 3-19: Matriz de confusión con aplicación de corte óptimo para el modelo logıstico sin balanceo de datos

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Sin balanceo	0,8000	0,9249	0,9248

Tabla 3-20: Métricas asociadas al modelo logıstico con aplicación de corte óptimo sin balanceo de datos

3.6.2. Aplicación del modelo logístico con datos balanceados undersampling

Teniendo en cuenta que el objetivo es clasificar los fraudes y no fraudes, la variable de respuesta se definió como 1 (fraude) y 0 (no fraude).

Se aplica a la base de datos la metodología de balanceo undersamplig y se ejecuta el modelo logístico con los datos de entrenamiento correspondiente al 80 % y se valida los resultados con los datos de prueba equivalentes al 20 %.

	Estimate	Std.Error	z value	value	Pr(>|z|)	exp(Estimate)
(Intercept)	-3,585	3,130	-1,146	0,252		2,8E-02
varMedioOtros	1,861	1,591	1,170	0,242		6,431
varMedioReferido	0,583	1,457	0,400	0,689		1,792
intIdPunto86101	-5,357	1,456	-3,679	2E-04	***	0,005
intIdPunto86103	1,895	1,652	1,147	0,251		6,650
intIdPunto86800	2,707	1,777	1,524	0,128		14,984
intIdPuntoOtros	-1,492	1,174	-1,271	0,204		0,225
intIdCanal1045	-1,094	1,232	-0,888	4E-01		0,335
intIdCanalOtros	-2,556	1,799	-1,421	2E-01		0,078
ntMesFecSolicitudTrim_2	3,589	1,279	2,806	0,005	**	36,205
ntMesFecSolicitudTrim_3	0,437	1,164	0,375	0,708		1,547
ntMesFecSolicitudTrim_4	1,529	1,295	1,181	0,238		4,614
varGeneroMujer	-2,413	0,830	-2,906	4E-03	**	0,090
varEstCivilOtros	3,903	1,338	2,917	0,004	**	49,565
varEstCivilSoltero	2,361	1,017	2,321	2E-02	*	10,607
varEstCivilUnionLibre	0,988	1,194	0,827	0,408		2,686
varTipoViviendaFamiliar	3,699	1,212	3,052	2E-03	**	40,426
varTipoViviendaPropia	1,661	1,195	1,390	0,164		5,263
varEstrato2	-0,489	1,003	-0,488	0,626		0,613
varEstrato3	0,361	0,966	0,373	0,709		1,435
varEstrato4	2,350	1,589	1,479	0,139		10,487
varEstrato5_6	-3,006	1,994	-1,507	0,132		0,050
varTipoInmuebleUrbano	2,557	1,461	1,751	0,080	,	12,898
OcupacionIndependiente	3,344	1,000	3,345	8E-04	***	28,325
varCodOcupacionOtros	0,098	1,190	0,082	0,934		1,103
varNivEstudioSecundaria	-0,627	0,952	-0,658	0,510		0,534
varNivEstudioTecnico	-0,091	1,111	-0,082	0,934		0,913
arNivEstudioUniversitario	1,320	1,039	1,270	0,204		3,742
tadoRealTarjetaActivada	-6,328	1,298	-4,875	0,000	***	0,002
Edad18-22	2,652	2,368	1,120	0,263		1,4E+01
Edad23-27	0,608	1,994	0,305	8E-01		1,836
Edad28-32	0,044	1,742	0,025	0,980		1,045
Edad33-37	3,586	2,120	1,692	0,091	,	36,092
Edad38-42	4,071	1,986	2,049	0,040	*	58,613
Edad43-47	1,351	2,004	0,674	0,500		3,861
Edad48-52	1,129	1,853	0,609	0,542		3,092
Edad53-57	-0,238	1,889	-0,126	0,900		0,788
Edad58-62	1,597	1,739	0,918	0,358		4,937
PersonasACargo0	-1,040	1,028	-1,012	0,311		0,353
PersonasACargo1	-2,555	1,148	-2,226	0,026	*	0,078

Figura 3-11: Nivel de significancia para el modelo logístico con balanceo de datos undersampling

En la Figura **3-11** se pueden observar las variables significativas con sus respectivos coeficientes, en donde se puede constatar según la prueba de significancia un p-valor menor o igual a 0,05, lo que garantiza que existen algunas variables como **intIdPunto**, **varCodOcupacion** y **EstadoReal** que son altamente significativas. Con respecto al signo se debe tener en cuenta que si este es negativo quiere decir que es una variable que premia y si es positivo es una variable que castiga.

Determinación del punto de corte óptimo y curva ROC del modelo logístico con balanceo de datos undersampling

Como el objetivo es maximizar tanto la sensibilidad como la especificidad, (Hosmer et al., 2013) sugiere la importancia de encontrar un punto de corte óptimo. En la Figura **3-12** se puede observar que el punto óptimo es de 72,00 % con un área bajo la curva (AUC) de

92,73 %.

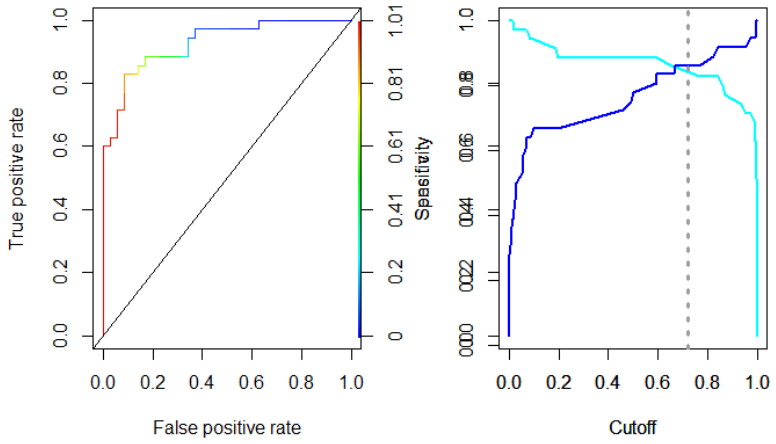


Figura 3-12: Punto de corte óptimo y curva ROC del modelo logístico con balanceo de datos undersampling.

Al aplicar el punto de corte óptimo de 72,00 %, se puede observar en la Tabla **3-21** el resultado de una la matriz de confusión más ajustada y con mejores medidas de desempeño, sin embargo métricas como la precisión y la especificidad estuvieron por debajo de las obtenidas en el modelo con datos desbalanceados, sin embargo, la sensibilidad presenta un mejor ajuste, pasando del 80,00 % en el modelo sin balanceo al 82,85 % en el modelo con balanceo (Undersampling), según lo muestra la Tabla **3-22**.

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	30	6
	fraude	5	29

Tabla 3-21: Matriz de confusión con aplicación de corte óptimo para el modelo logıstico con balanceo de datos undersampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Undersampling	0,8285	0,8571	0,8428

Tabla 3-22: Métricas asociadas al modelo logıstico con aplicación de corte óptimo con balanceo de datos undersampling

3.6.3. Aplicación del modelo logístico con datos balanceados oversampling

Se aplica a la base de datos la metodología de balanceo oversampling y se ejecuta el modelo logístico con los datos de prueba equivalentes al 20,00 % de la base de datos completa.

	Estimate	Std.Error	z value	value	Pr(>|z|)	exp(Estimate)
(Intercept)	-16,593	37,774	-0,439	0,660		6,2E-08
varMedioOtros	-0,222	0,036	-6,110	0,000	***	0,801
varMedioReferido	-0,452	0,036	-12,616	< 0,000	***	0,636
intIdPunto86101	-3,545	0,030	-119,341	< 2E-16	***	0,029
intIdPunto86103	0,488	0,038	12,916	< 0,000	***	1,629
intIdPunto86800	1,175	0,039	29,849	< 0,000	***	3,238
intIdPuntoOtros	-0,476	0,028	-17,308	< 0,000	***	0,621
intIdCanal1045	-1,590	0,032	-50,049	< 2E-16	***	0,204
intIdCanalOtros	-2,927	0,041	-70,736	< 2E-16	***	0,054
ntMesFecSolicitudTrim_2	2,223	0,028	80,734	< 0,000	***	9,234
ntMesFecSolicitudTrim_3	0,949	0,028	33,388	< 0,000	***	2,583
ntMesFecSolicitudTrim_4	1,676	0,028	60,164	< 0,000	***	5,347
varGeneroMujer	-1,728	0,017	-99,757	< 2E-16	***	0,178
varEstCivilOtros	2,404	0,033	72,863	< 0,000	***	11,069
varEstCivilSoltero	1,309	0,023	55,812	< 2E-16	***	3,702
varEstCivilUnionLibre	0,631	0,026	24,211	< 0,000	***	1,880
varTipoViviendaFamiliar	2,082	0,025	82,288	< 2E-16	***	8,018
varTipoViviendaPropia	0,895	0,028	32,339	< 0,000	***	2,447
varEstrato2	-0,810	0,025	-32,329	< 0,000	***	0,445
varEstrato3	-0,443	0,026	-17,148	< 0,000	***	0,642
varEstrato4	0,343	0,033	10,467	< 0,000	***	1,409
varEstrato5_6	-1,239	0,048	-25,998	< 0,000	***	0,290
varTipoInmuebleUrbano	0,991	0,039	25,088	< 0,000	***	2,693
OcupacionIndependiente	2,207	0,018	119,486	< 2E-16	***	9,093
varCodOcupacionOtros	-0,236	0,030	-7,959	0,000	***	0,790
varNivEstudioOtros	14,694	37,774	0,389	0,697		2,4E+06
varNivEstudioSecundaria	14,722	37,774	0,390	0,697		2,5E+06
varNivEstudioTecnico	14,561	37,774	0,385	0,700		2,1E+06
arNivEstudioUniversitario	16,463	37,774	0,436	0,663		1,4E+07
EstadoRealOtros	-16,995	89,764	-0,189	0,850		4,2E-08
tadoRealTarjetaActivada	-4,259	0,027	-155,132	< 2E-16	***	0,014
Edad18-22	0,521	0,054	9,579	< 0,000	***	1,683
Edad23-27	0,363	0,048	7,606	0,000	***	1,437
Edad28-32	1,977	0,045	44,344	< 0,000	***	7,222
Edad33-37	2,115	0,045	46,937	< 0,000	***	8,285
Edad38-42	2,068	0,045	45,568	< 0,000	***	7,913
Edad43-47	0,924	0,047	19,547	< 0,000	***	2,520
Edad48-52	0,757	0,047	15,963	< 0,000	***	2,132
Edad53-57	0,606	0,047	13,008	< 0,000	***	1,833
Edad58-62	1,157	0,049	23,710	< 0,000	***	3,182
PersonasACargo0	0,205	0,022	9,286	< 0,000	***	1,228
PersonasACargo1	-0,979	0,025	-39,808	< 0,000	***	0,376

Figura 3-13: Nivel de significancia para el modelo logístico con balanceo de datos oversampling

En la Figura **3-13** se pueden observar las variables con sus respectivos coeficientes, en donde se puede constatar según la prueba de significancia un p-valor menor o igual a $0,05$, por lo anterior se puede decir que todas las variables son altamente significativas, excepto la variable **varNivEstudio**.

Determinación del punto de corte óptimo y curva ROC del modelo logıstico con

balanceo de datos oversampling

En la Figura **3-14** se puede observar que el punto óptimo es de 49,35 % con un área bajo la curva (AUC) de 96,45 %.

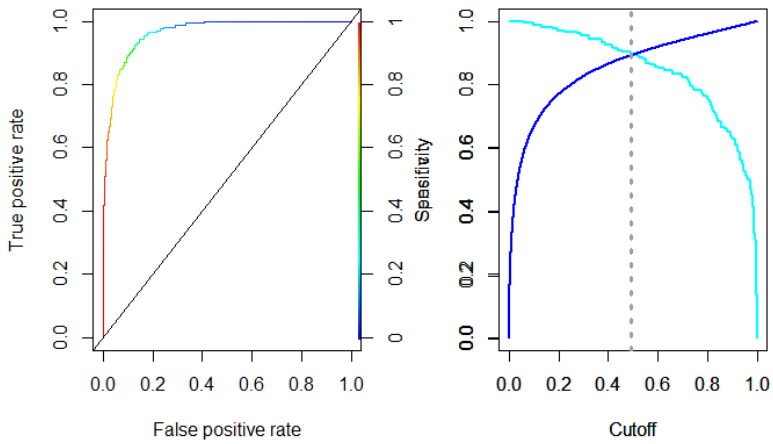


Figura 3-14: Punto de corte óptimo y curva ROC del modelo logístico con balanceo de datos oversampling.

Al aplicar el punto de corte óptimo de 49,35 %, se puede observar en la Tabla **3-23** el resultado de una la matriz de confusión más ajustada y con mejores medidas de desempeño. Con esta metodología se obtiene un mejor ajuste en todas las métricas, entre ellas la sensibilidad que pasa de 80,00 % con el modelo (undersampling) a 89,72 % con el modelo (oversampling), valor que garantiza una excelente clasificación de los fraudes. En la Tabla **3-24** se observa las metricas las cuales son superiores a los obtenidos con la metodología de balanceo undersampling.

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	27.350	3.149
	fraude	3.301	27.502

Tabla 3-23: Matriz de confusión con aplicación de corte óptimo para el modelo logıstico con balanceo de datos oversampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	0,8972	0,8923	0,8947

Tabla 3-24: Métricas asociadas al modelo logıstico con aplicación de corte óptimo con balanceo de datos oversampling

3.6.4. Aplicación del modelo árbol de clasificación con datos desbalanceados

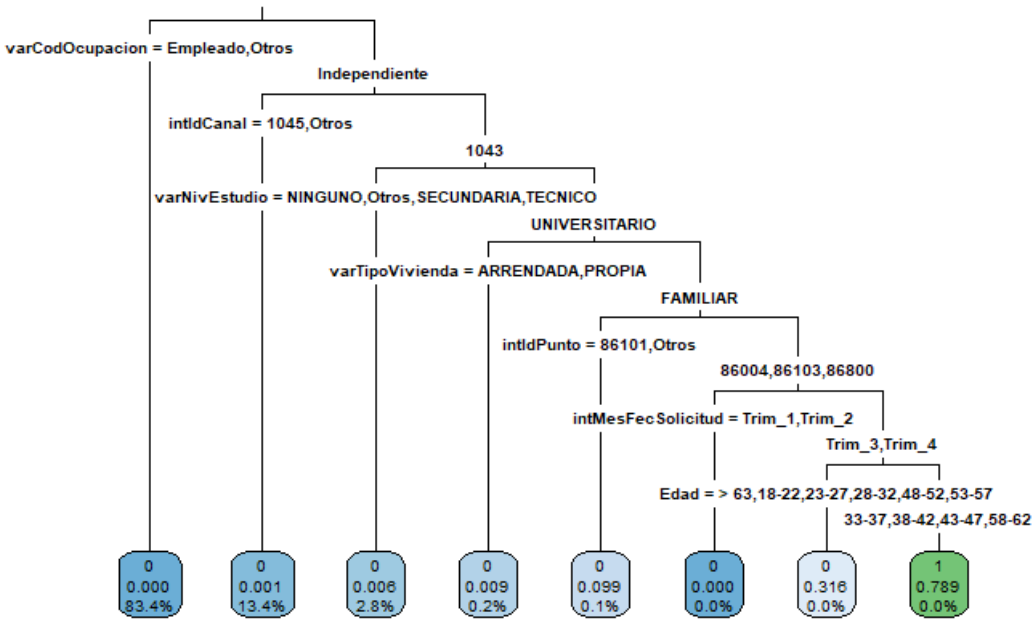


Figura 3-15: Árbol de clasificación sin balanceo de datos.

Para la aplicación de esta metodología se hace uso de la función rpart, ejecutando varios modelos con los datos de entrenamiento correspondiente al 80,00 % y validando los resultados con los datos de prueba equivalentes al 20,00 %, para su ejecución se tiene en cuenta las variables seleccionadas y la aplicación de un criterio de complejidad bajo (cp = 0,00001) el cual ira modificándose hasta encontrar un error de validación cruzada que minimice este criterio y permita encontrar un árbol con los nodos óptimos (podado) de manera tal que pueda interpretarse más fácilmente.

Con un error de validación cruzada (xerror = 0,97842) la cual minimiza el criterio de complejidad a 0,0086331, se ejecuta el modelo final del árbol sin balanceo de datos, este consta de 8 nodos terminales en los cuales se puede ver que una de las variables con mayor capacidad discriminativa es varCodOcupacion específicamente cuando esta es igual a empleado

y/o otros (el 83,40 % de los clientes se encuentra en este nodo).

Determinación del punto de corte óptimo y curva ROC del modelo árbol de clasificación sin balanceo de datos

Como el objetivo es maximizar la relación entre la sensibilidad y la especificidad se analizan diferentes umbrales sobre los indicadores de la matriz de confusión, con un umbral óptimo de 0.10, tenemos un modelo con las métricas descritas en la Tabla **3-25** en donde se puede ver que al igual que en el modelo logístico con datos desbalanceados, el modelo presenta un "accuracy" o "precisión"muy alta de 99,89 %, y se debe tener cuidado de llegar a una falsa conclusión de que el modelo es bueno, cuando realmente no lo es,

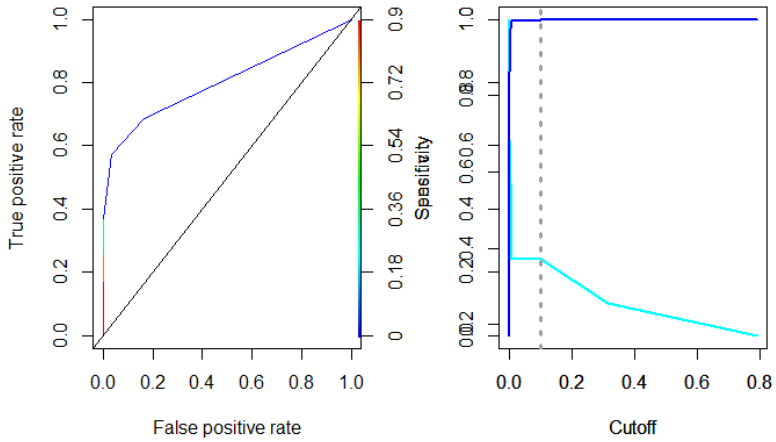


Figura 3-16: Punto de corte óptimo y curva ROC del modelo árbol de clasificación sin balanceo de datos

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	30.646	5
	fraude	26	9

Tabla 3-25: Matriz de confusión con aplicación de corte óptimo para el modelo árbol de clasificación sin balanceo de datos

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Sin balanceo	0,6429	0,9992	0,9990

Tabla 3-26: Métricas asociadas al modelo árbol de clasificación con aplicación de corte óptimo sin balanceo de datos

3.6.5. Aplicación del modelo árbol de clasificación con datos balanceados Undersampling

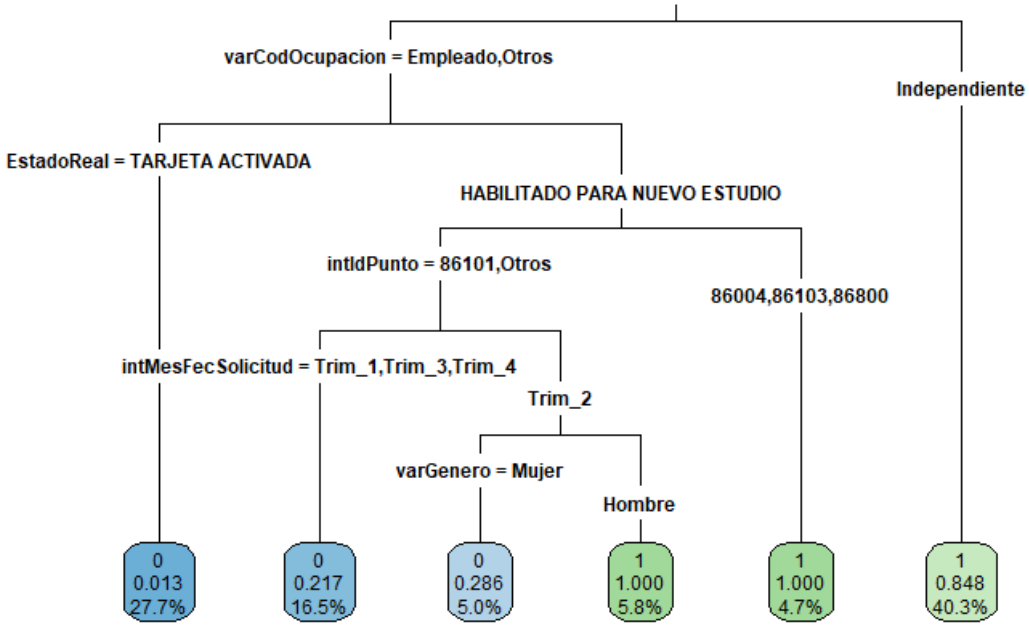


Figura 3-17: Árbol de clasificación base de datos undersampling.

Se aplica a la base de datos la metodología de balanceo undersampling y se ejecuta varios modelos con los datos de entrenamiento correspondiente al 80,00 % y se valida los resultados con los datos de prueba equivalentes al 20,00 %, para su ejecución se tiene en cuenta las variables seleccionadas y la aplicación de un criterio de complejidad bajo (cp = 0,00001) el cual ira modificándose hasta encontrar un error de validación cruzada que minimice este criterio y permita encontrar un árbol con los nodos óptimos (podado) de manera tal que pueda interpretarse más fácilmente.
Con un error de validación cruzada (xerror = 0,29496) la cual minimiza el criterio de complejidad a 0,021583, se ejecuta el modelo final del árbol con datos balanceados bajo la metodologia de undersampling, este puede observase en la Figura **3-17** consta de 8 nodos terminales en los cuales se puede ver que una de las variables con mayor capacidad discriminativa es varCodOcupacion cuando esta es igual a independiente (el 43,30 % de los clientes se encuentra en este nodo).

Punto de corte optimo y curva ROC del modelo arbol de clasificacion con balanceo de datos undersampling

Como el objetivo es maximizar la relación entre la sensibilidad y la especificidad se analizan diferentes umbrales sobre los indicadores de la matriz de confusión, con un umbral óptimo de 0,30, en la Tabla **3-28** se puede observar que se tiene un modelo que mejora en sensibilidad con respecto al modelo de árbol con datos desbalanceados pasando de 64,29 % a 73,81 % , sin embargo en especificidad y precisión las métricas del modelo de árbol sin balanceo de datos es mejor.

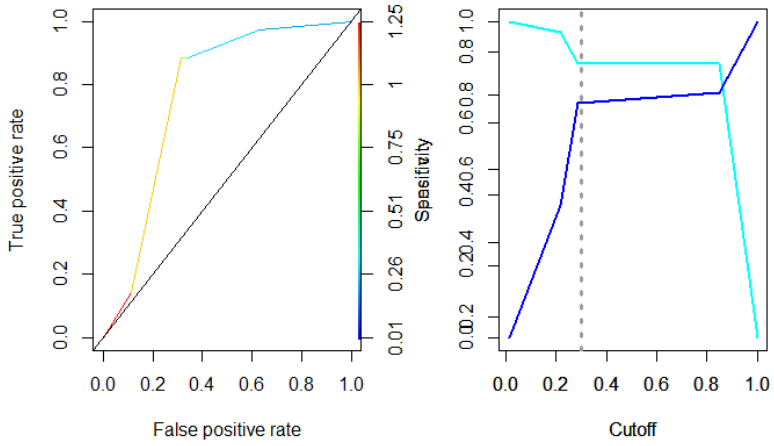


Figura 3-18: Punto de corte óptimo y curva ROC del modelo árbol de clasificación con balanceo de datos undersampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	24	11
	fraude	4	31

Tabla 3-27: Matriz de confusión con aplicación de corte óptimo para el modelo árbol de clasificación con balanceo de datos undersampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Undersampling	0,7381	0,8571	0,7857

Tabla 3-28: Métricas asociadas al modelo árbol de clasificación con aplicación de corte óptimo con balanceo de datos undersampling

3.6.6. Aplicación del modelo árbol de clasificación con datos balanceados oversampling

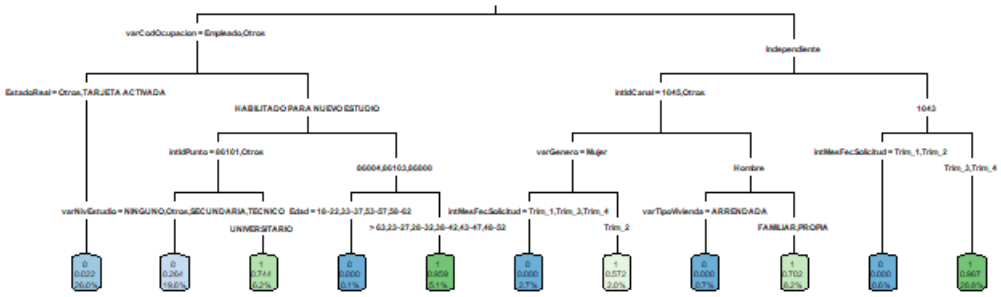


Figura 3-19: Resultado del árbol de clasificación con la base de datos y el método oversampling

Se aplica a la base de datos la metodología de balanceo oversampling y se ejecuta varios modelos con los datos de entrenamiento correspondiente al 80,00 % y se valida los resultados con los datos de prueba equivalentes al 20,00 %, para su ejecución se tiene en cuenta las variables seleccionadas y la aplicación de un criterio de complejidad bajo (cp = 0,00001) el cual ira modificándose hasta encontrar un error de validación cruzada que minimice este criterio y permita encontrar un árbol con los nodos óptimos (podado) de manera tal que pueda interpretarse más fácilmente.
Con un error de validación cruzada (xerror = 0,0037846) la cual minimiza el criterio de complejidad a 0,000010, se ejecuta el modelo final del árbol, este se puede observase en la Figura **3-19** consta de 11 nodos terminales en los cuales se puede ver que una de las variables con mayor capacidad discriminativa es varCodOcupacion cuando esta es igual a independiente (el 28,00 % de los clientes se encuentra en este nodo).

Determinación del punto de corte óptimo y curva ROC del modelo árbol de clasificación con balanceo de datos oversampling

Como el objetivo es maximizar la relación entre la sensibilidad y la especificidad se analizan diferentes umbrales sobre los indicadores de la matriz de confusión, con un umbral óptimo de 0,5, tenemos un modelo más estables, que mejora notablemente la métrica de la sensibilidad con respecto a los modelos de árbol aplicados anteriormente y la especificidad y la precisión con respecto al modelo con balanceo undersampling como se puede observar en la Tabla. **3-30**

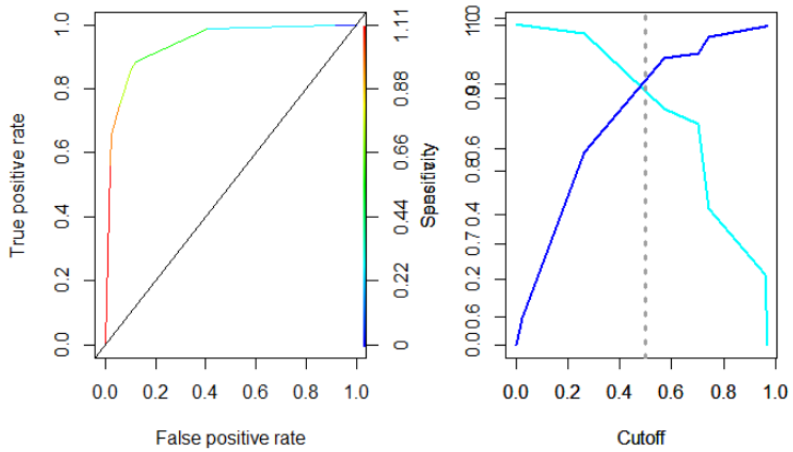


Figura 3-20: Punto de corte óptimo y curva ROC del modelo árbol de clasificación con balanceo de datos oversampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	26.897	3,754
	fraude	3.509	27.142

Tabla 3-29: Matriz de confusión con aplicación de corte óptimo para el modelo árbol de clasificación con balanceo de datos oversampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	0,8785	0,8846	0,8815

Tabla 3-30: Métricas asociadas al modelo árbol de clasificación con aplicación de corte óptimo con balanceo de datos oversampling

3.6.7. Aplicación modelo máquinas de soporte vectorial (SVM)

Para la aplicación de estas metodologías se tienen en cuenta tanto los datos desbalanceados como los balanceados con undersampling y oversampling en el entrenamiento (para el ajuste de los modelos) y prueba (para la evaluación o datos de validación de los mismos). La división es del 80,00 %-20,00 % y se establece una semilla de inicio igual a 123.

Optimización de hiperparámetros mediante validación cruzada 10-fold

Como se desconoce si los datos son separables linealmente, se aplicarán modelos con tipo de kernel : lineal, polinomial y radial. Y se elegirá el que mejor ajuste que presenten los datos. Con el fin de garantizar la capacidad predictiva del modelo, es necesario fijar un margen de separación mediante la configuración del kernel lineal, este se realiza mediante validación cruzada para escoger el valor óptimo y el cual será aplicado a los modelos con kernel polinómico y radial.

Para la aplicación de los modelos con la base de datos balanceada con oversampling se tomará una muestra de1 5,00 % de la base de datos completa, equivalente a 15.325 registros la cual se particionará a su vez en 80,00 % equivalente a 12.260 para entrenamiento y 20,00 % equivalente a 3.066 registros para la prueba, lo anterior con el fin de agilizar el procesamiento de máquina, ya que el proceso con los datos completos es muy demorado.

Costo	Error	Dispersión
0,001	0,00116333	0,0016449
0,010	0,00116333	0,0016449
0,100	0,00116333	0,0016449
1,000	0,0011406	0,0005697
5,000	0,00186155	0,0014712
10,000	0,00256031	0,0023148
15,000	0,00325798	0,0022470
20,000	0,00372364	0,0025005

Figura 3-21: Valor del costo óptimo sin balanceo. Fuente: elaboración propia

Costo	Error	Dispersión
0,001	0,5218254	0,1238644
0,010	0,1760582	0,0699780
0,100	0,157672	0,0711280
1,000	0,1503968	0,0816520
5,000	0,1431217	0,0887346
10,000	0,1361111	0,0834451
15,000	0,1580688	0,0673413
20,000	0,1506614	0,0724352

Figura 3-22: Valor del costo óptimo undersampling. Fuente: elaboración propia

Costo	Error	Dispersión
0,001	0,1681442	0,0109463
0,010	0,1134925	0,0118858
0,100	0,1026551	0,0118865
1,000	0,1013731	0,0093293
5,000	0,1019563	0,0105266
10,000	0,1024226	0,0105178
15,000	0,1031219	0,0111574
20,000	0,1024226	0,0107870

Figura 3-23: Valor del costo óptimo oversampling. Fuente: elaboración propia

Se puede observar en la Figura **3-21** que el valor de costo óptimo que resulta en el menor error de validación (0,0011) es 1, para los datos desbalanceados, en la Figura **3-22** el valor de costo óptimo que resulta en el menor error de validación (0,136) es 10, para los datos balanceados con undersampling y en la Figura **3-23** se puede ver que el costo que resulta en el menor error de validación (0,101) es 1, para los datos balanceados con oversampling.

3.6.8. Aplicación del modelo SVM kernel lineal con datos desbalanceados

		Clase Actuaol	
		No Fraude	Fraude
Clase Predicha	No Fraude	30.065	3
	fraude	0	0

Tabla 3-31: Matriz de confusión con aplicación de corte óptimo para el modelo SVM lineal sin balanceo de datos

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Sin balanceo	0,0000	1,0000	0,9990

Tabla 3-32: Métricas asociadas al modelo SVM lineal con aplicación de corte óptimo sin balanceo de datos

Al aplicar el modelo SVM con kernel lineal y base de datos sin balanceo,se puede observar en la Tabla **3-32** que las métricas no son muy ajustadas ya que la sensibilidad tiene el 0,00 %, y la especificidad se ajusta al 1,00 %, también presenta una precisión muy alto del 99,95 % debido al desbalanceo tan alto de los datos.

3.6.9. Aplicación del modelo SVM kernel lineal con datos balanceados undersampling

		Clase Actual	
		No Predicha	Fraude
Clase Actual	No Fraude	27	3
	fraude	8	32

Tabla 3-33: Matriz de confusión con aplicación de corte óptimo para el modelo SVM lineal con balanceo de datos undersampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Undersampling	0,9143	0,7714	0,8429

Tabla 3-34: Métricas asociadas al modelo SVM lineal con aplicación de corte óptimo con balanceo de datos undersampling

Al aplicar el modelo SVM con kernel lineal y base de datos balanceada con undersampling, se puede observar en la Tabla **3-34** que presenta un buen ajuste de la sensibilidad con un 91,00 %, de la especificidad con un 77,00 % y la precisión con un 84,29 %.

3.6.10. Aplicación del modelo SVM kernel lineal con datos balanceados oversampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	1.359	156
	fraude	174	1.377

Tabla 3-35: Matriz de confusión con aplicación de corte óptimo para el modelo SVM lineal con balanceo de datos oversampling

En la Tabla **3-35** se puede observar la matriz de confusión aplicado el punto de corte óptimo, es importante resaltar que esta metodología se aplica a una muestra de1 5,00 % de la base de datos completa, equivalente a 15.325 registros la cual se particionará a su vez en 80,00 % equivalente a 12.260 para entrenamiento y 20,00 % equivalente a 3.066 registros para la prueba, lo anterior con el fin de agilizar el procesamiento de máquina, ya que al utilizar la

base de datos completa, se requiere de un procesamiento de maquina considerable que se traduce en tiempos lentos de respuesta.

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	0,8982	0,8864	0,8923

Tabla 3-36: Métricas asociadas al modelo SVM lineal con aplicación de corte óptimo con balanceo de datos oversampling

Al aplicar el modelo SVM con kernel lineal y base de datos balanceada con oversampling, se puede observar en la Tabla **3-36** que presenta un mejoramiento en las métricas comparado con los dos modelos anteriores aplicados con este mismo kernel.

3.6.11. Aplicación del modelo SVM kernel polinomial con datos de prueba sin balanceo de datos.

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	30.651	35
	fraude	0	0

Tabla 3-37: Matriz de confusión con aplicación de corte óptimo para el modelo SVM polinomial sin balanceo de datos

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Sin balanceo	0,0000	1,0000	0,9988

Tabla 3-38: Métricas asociadas al modelo SVM polinomial con aplicación de corte óptimo sin balanceo de datos

Al aplicar el modelo SVM con kernel polinomial y base de datos sin balanceo, se puede observar en la Tabla **3-38** que las métricas son muy parecidas al obtenido en el modelo con kernel lineal sin balanceo de datos, estas se deben en gran parte al desbalanceo de la base de datos.

3.6.12. Aplicación del modelo SVM kernel polinomial con datos de prueba y metodología de balanceo undersampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	33	10
	fraude	2	25

Tabla 3-39: Matriz de confusión con aplicación de corte óptimo para el modelo SVM polinomial con balanceo de datos undersampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Undersampling	0,7143	0,9429	0,8286

Tabla 3-40: Métricas asociadas al modelo SVM polinomial con aplicación de corte óptimo con balanceo de datos undersampling

Como se puede observar en la Tabla **3-40** aplicando el modelo SVM con kernel polinomial y base de datos undersampling, vemos que la sensibilidad baja considerablemente a un 71,00 %, pero hay un mejoramiento de la especificidad del 94,00 % las métricas de accuracy se mantienen estables en 83,00 %, pero menores a los valores de la tabla anterior.

3.6.13. Aplicación del modelo SVM kernel polinomial con datos balanceados oversampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	1.458	301
	fraude	75	1.232

Tabla 3-41: Matriz de confusión con aplicación de corte óptimo para el modelo SVM polinomial con balanceo de datos oversampling

En la Tabla **3-41** se puede observar la matriz de confusión aplicado el punto de corte óptimo, es importante resaltar que esta metodología se aplica a una muestra de1 5,00 % de la base de datos completa, equivalente a 15.325 registros la cual se particionará a su vez en 80,00 % equivalente a 12.260 para entrenamiento y 20,00 % equivalente a 3.066 registros para la

prueba, lo anterior con el fin de agilizar el procesamiento de máquina, ya que al utilizar la base de datos completa, se requiere de un procesamiento de maquina considerable que se traduce en tiempos lentos de respuesta.

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	0,8036	0,9510	0,8773

Tabla 3-42: Métricas asociadas al modelo SVM polinomial con aplicación de corte óptimo con balanceo de datos oversampling

En la Tabla **3-42** se puede observar que con la aplicación del modelo SVM con kernel polinomial y base de datos oversampling, vemos un mejoramiento en todas las métricas con respecto a los dos modelos con kernel polinomial aplicados anteriormente, sin embargo estas métricas están un poco mas bajas comparadas con las del modelo con kernel polinomial con balanceo de datos oversampling.

3.6.14. Aplicación del modelo SVM kernel radial con datos desbalanceados

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	30.651	35
	fraude	0	0

Tabla 3-43: Matriz de confusión con aplicación de corte óptimo para el modelo SVM radial sin balanceo de datos

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Sin balanceo	0,000	1,0000	0,9985

Tabla 3-44: Métricas asociadas al modelo SVM radial con aplicación de corte óptimo sin balanceo de datos

En la Tabla **3-44**. se puede observar que con el modelo SVM con kernel radial y sin balanceo de datos , muestra unas métricas similares a los modelos con kernel lineal y radial sin balanceo de datos, confirmado que este es un problema que se debe corregir con una metodología de balanceo.

3.6.15. Aplicación del modelo SVM kernel radial con datos balanceados undersampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	29	3
	fraude	6	32

Tabla 3-45: Matriz de confusión con aplicación de corte óptimo para el modelo SVM radial con balanceo de datos undersampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Undersampling	0,9143	0,8286	0,8714

Tabla 3-46: Métricas asociadas al modelo SVM radial con aplicación de corte óptimo con balanceo de datos undersampling

Con el modelo SVM con kernel radial y base de datos undersampling, se ve un mejoramiento de todas las métricas con respecto al modelo sin balanceo de datos como se puede observar en la Tabla **3-46**.

3.6.16. Aplicación del modelo SVM kernel radial con datos balanceados oversampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	1.419	47
	fraude	114	1.486

Tabla 3-47: Matriz de confusión con aplicación de corte óptimo para el modelo SVM radial con balanceo de datos oversampling

En la Tabla **3-47** se puede observar la matriz de confusión aplicado el punto de corte óptimo, es importante resaltar que esta metodología se aplica a una muestra de1 5,00 % de la base de datos completa, equivalente a 15.325 registros la cual se particionará a su vez en 80,00 % equivalente a 12.260 para entrenamiento y 20,00 % equivalente a 3.066 registros para la prueba, lo anterior con el fin de agilizar el procesamiento de máquina, ya que al utilizar la

base de datos completa, se requiere de un procesamiento de maquina considerable que se traduce en tiempos lentos de respuesta.

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	0,9693	0,9256	0,9474

Tabla 3-48: Métricas asociadas al modelo SVM radial con aplicación de corte óptimo con balanceo de datos oversampling

Con el modelo SVM con kernel radial y base de datos oversampling, se pueden observar muy buen ajuste en las métricas con respecto a los dos modelos con kernel radial aplicados anteriormente, valores muy buenos en la sensibilidad de 96,25 %, la especificidad con valores de 91,95 % y una precisión del 94,10 % como se puede observar en la Tabla **3-48**.

3.6.17. Comparación modelos SVM con mejor kernel

En la Tabla **3-52** se puede ver un resumen de los diferentes modelos aplicados en máquina de soporte vectorial (SVM) tanto para los datos desbalanceados como para datos balanceados undersampling y oversampling. La metodología con la cual se obtiene un mejor ajuste en todas las métricas es para el modelos con kernel radial para datos balanceados con la metodologia oversampling, este alcanza un ajuste muy alto en la **sensibilidad, accuracy y especificidad** superando a los modelos en donde se aplicó máquinas de soporte con kernel lineal y polinomial. Ahora se tomarán estos modelos y se procesarán con la base de datos completa.

	Tipo Balanceo	Sensibilidad	Especificidad	Precisión
1	SVM Lineal sin balanceo	0,0000	1,0000	0,9990
2	SVM Lineal balanceo under	0,9143	0,7714	0,8429
3	SVM Lineal balanceo over	0,8982	0,8864	0,8923
4	SVM Polinómico sin balanceo	0,0000	1,0000	0,9988
5	SVM Polinómico balanceo unde	0,7143	0,9429	0,8286
6	SVM Polinómico balanceo over	0,8036	0,9510	0,8773
7	SVM Radial sin balanceo	0,0000	1,0000	0,9985
8	SVM Radial balanceo under	0,9143	0,8286	0,8714
9	SVM Radial balanceo over	0,9693	0,9256	0,9474

Tabla 3-49: Comparación modelos SVM sin balanceo y con metodología de balanceo undersampling y oversampling.

3.6.18. Aplicación modelo mejor kernel con la base de datos completa de oversampling

		Clase Actual	
		No Fraude	Fraude
Clase Predicha	No Fraude	29.990	661
	fraude	0	30.651

Tabla 3-50: Matriz de confusión modelo SVM mejor kernel con datos de prueba y metodología de balanceo oversampling

Tipo Balanceo	Sensibilidad	Especificidad	Precisión
Oversampling	1,0000	0,9784	0,9892

Tabla 3-51: Métricas modelo SVM mejor kernel con datos de prueba y metodología de balanceo oversampling.

Se puede observar en la tabla **3-51** que en el modelo de máquinas de soporte vectorial con kernel radial y base de datos completa y balanceada con metodología oversampling, presenta un ajuste casi perfecto de las métricas, lo que demuestra que el modelo es muy bueno detectando tanto fraudes como no fraudes.
Una vez seleccionamos nuestro mejor modelo de máquinas de soporte vectorial, ahora se comparará este con los modelos logísticos y árboles de clasificación tanto para la base de datos balanceada con undersampling como con oversampling.

3.6.19. Comparación modelos: Logísticos, árboles de clasificación y máquinas de soporte vectorial

Se aplicaron las metodologías de balanceo overtraing y undertraing a la base de datos original, estas a su vez se particionaron en dataTrain 80 % y dataTest 20 % para cada metodología de balanceo se aplicaron modelos logísticos, árboles de clasificación y máquinas de soporte vectorial, con el fin de verificar las métricas y seleccionar el modelo con el mejor ajuste.
En la Tabla **3-48** vemos como la aplicación de una metodología de balanceo de datos mejora considerablemente las métricas, permitiendo la aplicación de un modelo y alcanzar un excelente ajuste, en este caso se puede observar que las máquinas de soporte vectorial clasifican muy bien los fraudes y no fraudes, sin embargo, el compromiso de máquina es muy alto. Con un número de registros bajo, los árboles de clasificación también alcanzan ajustes muy buenos en la métrica de sensibilidad, pero en la especificidad y accuracy no. El modelo logístico

con metodología para datos desbalanceados oversampling, alcanza ajustes muy buenos en todas las métricas sin comprometer mucho la máquina y así se convierte en la metodología más interesante a la hora de clasificar fraudes y no fraudes, por esta razón se recomienda esta metodología ya que logra un mejor poblamiento de la base de datos y una buena estimación de la sensibilidad que es la capacidad del modelo para detectar los fraudes.

	Tipo Balanceo	Sensibilidad	Especificidad	Precisión
1	Logístico sin balanceo	0,8000	0,9249	0,9248
2	Logístico balanceo under	0,8285	0,8571	0,8425
3	Logístico balanceo over	0,8972	0,8923	0,8947
4	Árbol sin balanceo	0,6429	0,9992	0,9990
5	Árbol balanceo under	0,7381	0,8571	0,7857
6	Árbol balanceo over	0,8785	0,8846	0,8815
7	SVM Lineal sin balanceo	0,0000	1,0000	0,9990
8	SVM Lineal balanceo under	0,9143	0,7714	0,8429
9	SVM Lineal balanceo over	0,8982	0,8864	0,8923
10	SVM Polinómico sin balanceo	0,0000	1,0000	0,9988
11	SVM Polinómico balanceo unde	0,7143	0,9429	0,8286
12	SVM Polinómico balanceo over	0,8036	0,9510	0,8773
13	SVM Radial sin balanceo	0,0000	1,0000	0,9985
14	SVM Radial balanceo under	0,9143	0,8286	0,8714
15	SVM Radial balanceo over	0,9693	0,9256	0,9474
16	SVM Radial mejor kernel base completa	1,0000	0,9784	0,9892

Tabla 3-52: Comparación modelos logistico, arboles de clasificación y maquinas de soporte vectorial (SVM))

4 Conclusiones y recomendaciones

El resultado de un modelo se basa principalmente en los datos de entrada, es aquí donde radica la importancia de realizar un trabajo consiente de limpieza, extracción y transformación de los datos de tal modo que se puedan garantizar predicciones de alta calidad.

Otro reto fue trabajar con datos desbalanceados, ya que el algoritmo presentó problemas en la generalización de la información, debido a que las clases minoritarias se veían totalmente perjudicadas. Para corregir tal problema fue indispensable la aplicación de metodologías para el tratamiento de datos desbalanceados como *undersampling* y *oversampling*, las cuales resultaron eficientes al mejorar notablemente las medidas de desempeño en todos los modelos.

Los modelos aplicados en este proyecto tienen en común que son utilizados para la clasificación binaria entre categorías, cuyo objetivo es delimitar las categorías de manera correcta. En este sentido se aplicaron técnicas de regresión logística, árboles de clasificación y máquinas de soporte vectorial. Se ejecutan los modelos con los datos de entrenamiento correspondiente al 80 % y se validan los resultados con los datos de prueba equivalentes al 20 %, los tres modelos obtuvieron excelentes resultados, sin embargo, algunos requirieron más esfuerzo de máquina, lo que implicó mucho tiempo de procesamiento.

Para las máquinas de soporte vectorial se aplicaron los siguientes modelos: kernel lineal, polinómica y radial tanto para la base de datos desbalanceada como para las bases de datos balanceadas con las metodologías undersampling y oversampling, el ajuste de los modelos con kernel radial y polinómico superaron notablemente al kernel lineal en ambas bases de datos.
Al hacer una comparación de los diferentes modelos aplicados, podemos decir que las máquinas de soporte vectorial obtuvieron mejores resultados con una sensibilidad del 100 %, especificidad del 97,84 % y precisión del 98,92 %, seguido de la regresión logística que obtuvo una sensibilidad del 89,72 %, especificidad del 89,23 % y precisión del 89,47 %, los árboles de clasificación obtuvieron mejores resultados en la especificidad de 88,46 %, pero la sensibilidad y accuracy estuvieron por debajo de los otros modelos.

Teniendo en cuenta que los resultados de los tres modelos son muy buenos, hacemos uso del principio de parsimonia, en donde resulta beneficioso escoger el modelo más simple y que presente buenos resultados, como es el caso de la regresión logística, que en este caso es más

eficiente en el procesamiento de los datos y alcanza un buen ajuste a la hora de predecir los fraudes, además es un modelo lineal simple con respecto a otros modelos como las máquinas de soporte vectorial o árboles de clasificación.

El modelo logístico con datos balanceados bajo la metodología de oversampling claramente presenta un mejor desempeño frente al logístico con datos balanceados con undersampling, garantizando una mejor clasificación en todas las métricas, en especial la métrica de sensibilidad que es la capacidad del modelo de detectar los verdaderos positivos.

Anexo I: Recategorización de variables

A continuación se relacionan las variables que no cuentan con información suficiente en algunas categorías y que debido a esta condición son objeto de recategorización de sus clases.

- **intIdPunto:**

Variable con todas las categorias

Fraude	intIdPunto	n
0	86101	54.797
0	86004	21.808
0	86203	12.290
0	86915	9.399
0	86402	8.188
0	86001	6.321
0	86103	5.934
0	86800	5.630
0	86302	4.347
0	86098	3.850
0	86006	3.575
0	99998	1
1	86103	38
1	86101	37
1	86004	30
1	86800	25
1	86404	9
1	86915	9
1	86203	7
1	86001	4
1	86006	4
1	86302	4
1	86700	4
1	86402	2
1	86098	1

Variable con categorias agrupadas

Fraude	intIdPunto	n
0	86004	21.808
0	86101	54.797
0	86103	5.934
0	86800	5.630
0	Otros	65.084
1	86004	30
1	86101	37
1	86103	38
1	86800	25
1	Otros	44

Figura 4-1: Recategorización de la variable intIdPunto

En la Figura **4-1** se puede observar para la variable intIdPunto categorías con muy pocos registros como 86007, 86012, 86206, 86098, 86700 entre otras, estas categorías son agrupadas en la categoría otros.

- **intIdCanal:**

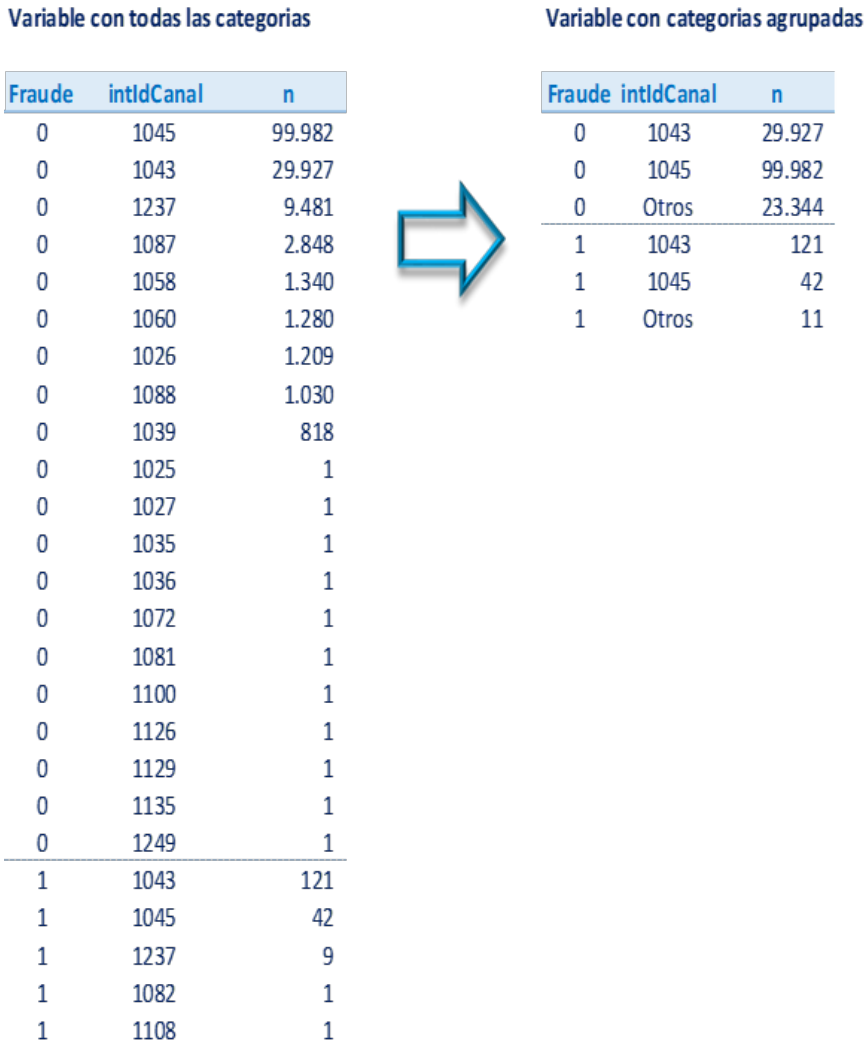


Figura 4-2: Recategorización de la variable intIdCanal

En la Figura **4-2** se observan para la variable intIdCanal las categorías 1025,1027,1033,1108,1082 cuentan con pocos registros, estos son agrupados en la categoría otros.

- **intMesFecSolicitud:**

Variable con todas las categorias

Fraude	intMesFecSolicitud	n
0	1	11.647
0	2	11.246
0	3	13.819
0	4	12.687
0	5	13.591
0	6	12.467
0	7	13.949
0	8	13.225
0	9	11.883
0	10	13.234
0	11	14.389
0	12	11.116
1	1	1
1	3	8
1	4	21
1	5	13
1	6	4
1	7	2
1	8	8
1	9	21
1	10	68
1	11	24
1	12	4

Variable con categorias agrupadas

Fraude	intMesFecSolicitud	n
0	Trim_1	36.712
0	Trim_2	38.745
0	Trim_3	39.057
0	Trim_4	38.739
1	Trim_1	9
1	Trim_2	38
1	Trim_3	31
1	Trim_4	96

Figura 4-3: Recategorización de la variable intMesFecSolicitud

En la Figura **4-3** se encuentran los registros para la variable intMesFecSolicitud que corresponde al mes en que se generó la solicitud, debido a que se encuentran meses con pocos registros en la categoría fraude, estos se agruparan en trimestres.

- **varEstCivil:**

Variable con todas las categorias

Fraude	varEstCivil	n
0	CASADO	43.578
0	DIVORCIADO	9.993
0	SOLTERO	57.738
0	UNIONLIBRE	35.228
0	VIUDO	6.716
1	CASADO	12
1	DIVORCIADO	9
1	SOLTERO	129
1	UNIONLIBRE	16
1	VIUDO	8

Variable con categorias agrupadas

Fraude	varEstCivil	n
0	CASADO	43.578
0	OTROS	16.709
0	SOLTERO	57.738
0	UNIONLIBRE	35.228
1	CASADO	12
1	OTROS	17
1	SOLTERO	129
1	UNIONLIBRE	16

Figura 4-4: Recategorización de la variable varEstCivil

En la Figura **4-4** se encuentran las categorías de la variable estado civil, las categorías soltero, divorciado y unión libre se agrupan en la categoría OTROS.

- **varEstrato:**

Variable con todas las categorias

Fraude	varEstrato	n
0	1	22.005
0	2	65.647
0	3	50.513
0	4	10.189
0	5	3.991
0	6	908
1	1	15
1	2	34
1	3	67
1	4	48
1	5	9
1	6	1

Variable con categorias agrupadas

Fraude	varEstrato	n
0	1	22.005
0	2	65.647
0	3	50.513
0	4	10.189
0	5_6	4.899
1	1	15
1	2	34
1	3	67
1	4	48
1	5_6	10

Figura 4-5: Recategorización de la variable varEstrato

En la Figura **4-5** se encuentran todas las categorías para la variable estrato, se agrupan los estratos 5 y 6, ya que cuentan con pocos registros.

- **varCodOcupacion:**

Variable con todas las categorias

Fraude	varCodOcupacion	n
0	AmaDeCasa	23.826
0	Empleado	88.611
0	Independiente	25.331
0	Pensionado	15.485
1	AmaDeCasa	4
1	Empleado	39
1	Independiente	121
1	Pensionado	10

Variable con categorias agrupadas

Fraude	varCodOcupacion	n
0	Empleado	88.611
0	Independiente	25.331
0	Otros	39.311
1	Empleado	39
1	Independiente	121
1	Otros	14

Figura 4-6: Recategorización de la variable varCodOcupacion

En la Figura **4-6** se encuentran todas las categorías para la variable ocupación, se agrupan los las categorías AmaDeCasa y Pensionado, ya que cuentan con pocos registros.

- **varNivEstudio:**

Variable con todas las categorias

Fraude	varNivEstudio	n
0	NINGUNO	1.100
0	POSGRADO	3.725
0	PRIMARIA	23.149
0	SECUNDARIA	65.471
0	SIN ESPECIFICAR	2
0	TECNICO	26.595
0	TECNOLOGICO	15.209
0	UNIVERSITARIO	18.002
1	POSGRADO	2
1	PRIMARIA	7
1	SECUNDARIA	39
1	TECNICO	17
1	TECNOLOGICO	14
1	UNIVERSITARIO	95

Variable con categorias agrupadas

Fraude	varNivEstudio	n
0	NINGUNO	1.100
0	OTROS	42.085
0	SECUNDARIA	65.471
0	TECNICO	26.595
0	UNIVERSITARIO	18.002
1	OTROS	23
1	SECUNDARIA	39
1	TECNICO	17
1	UNIVERSITARIO	95

Figura 4-7: Recategorización de la variable varNivEstudio

En la Figura **4-7** se encuentran categorías con pocos registros para la variable varNivEstudio, estos se agrupan en la categoría OTROS.

- **EstadoReal:**

Variable con todas las categorias

Fraude	EstadoReal	n
0	HABILITADO PARA NUEVO ESTUDIO	64.504
0	PENDIENTE ACTIVAR REDEBAN	101
0	PENDIENTE CONFRONTA	1
0	PENDIENTE CREACION CLIENTE	12
0	TARJETA ACTIVADA	88.553
0	TARJETA RESERVADA	82
1	HABILITADO PARA NUEVO ESTUDIO	94
1	TARJETA ACTIVADA	80

Variable con categorias agrupadas

Fraude	EstadoReal	n
0	HABILITADO PARA NUEVO ESTUDIO	64.504
0	OTROS	196
0	TARJETA ACTIVADA	88.553
1	HABILITADO PARA NUEVO ESTUDIO	94
1	TARJETA ACTIVADA	80

Figura 4-8: Recategorización de la variable EstadoReal

En la Figura **4-8** para la variable EstadoReal se encuentran categorías con pocos registros, estos se agrupan en la categoría OTROS.

- **Edad:**

Variable con todas las categorias

Fraude	Edad	n
0	18	1.082
0	19	1.899
0	20	2.498
0	21	2.973
0	22	3.204
0	23	3.444
0	24	3.654
0	25	3.794
0	26	3.843
0	27	3.785
0	28	3.839
0	29	3.728
0	30	3.904
0	31	3.709
0	32	3.537
0	33	3.722
0	34	3.605
0	35	3.666
0	36	3.735
0	37	3.702
0	70	972
0	71	814
0	74	542
0	75	392
1	20	2
1	21	2
1	23	1
1	24	1
1	25	2
1	26	4
1	27	3
1	28	3
1	29	8
1	30	3
1	35	7
1	36	4
1	37	12
1	38	7
1	39	7
1	45	2
1	46	5
1	47	6
1	48	3
1	49	1
1	50	2
1	51	2
1	52	2
1	53	2
1	54	3
1	59	3
1	60	1
1	63	1
1	64	1
1	72	1
1	75	1

Variable con categorias agrupadas

Fraude	Edad	n
0	> 63	13.929
0	18-22	11.656
0	23-27	18.520
0	28-32	18.717
0	33-37	18.430
0	38-42	16.478
0	43-47	14.393
0	48-52	15.772
0	53-57	14.503
0	58-62	10.855
1	> 63	10
1	18-22	4
1	23-27	11
1	28-32	30
1	33-37	34
1	38-42	29
1	43-47	22
1	48-52	10
1	53-57	16
1	58-62	8

Figura 4-9: Recategorización de la variable Edad

En la variable Edad se evidenciaron muchas categorías con poca información, debido a esto, se agruparon en rangos de edades, como se puede observar en la Figura **4-9** .

- **PersonasACargo:**

Variable con todas las categorias

Fraude	PersonasACargo	n
0	0	80.518
0	1	43.740
0	2	22.513
0	3	5.214
0	4	954
0	5	187
0	MAS DE 5	127
1	0	138
1	1	19
1	2	13
1	3	3
1	4	1

Variable con categorias agrupadas

Fraude	PersonasACargo	n
0	>=2	28.995
0	0	80.518
0	1	43.740
1	>=2	17
1	0	138
1	1	19

Figura 4-10: Recategorización de la variable PersonasACargo

En la Figura **4-10** para la variable PersonasACargo se encuentran categorías con pocos registros estos se agrupan en la categoría mayor a 2

Anexo II: Código desarrollado en RStudio

```
#title: "Tesis"
#author: "Lina"
#date: "31/01/2023"

# **Objetivo**: Proponer una metodología para detectar fraudes en tarjetas de credito,
#emitidas por una empresa particular, basada en metodos de clasificacion binaria y
#considerando datos altamente desbalanceados.

#***************** Librerias necesarias ******************

library(ggplot2)
library(knitr) # Para ver tablas mas amigables en formato html markdown
library(dplyr) # Varias operaciones
library(stats)
library(gamlss)
require(caret) #Modelos de Clasificación
library(ROCit)
library(pROC)
library(cutpointr)
library(OptimalCutpoints)
library(kableExtra)
library(randomForest)
library(kernlab)
library(magrittr)
require(GoFKernel)
library(PDQutils)
library(tidyverse)
library(grid)
library(ggpubr)        # Varios graficos en un panel
library(tree)          #Arboles de Regresion
library(rpart)         #Arboles de Regresión
library(rpart.plot)    #Grafica Arboles de Regresión
library("e1071")       #Liberías Vectores Maquinas. svm {e1071}
library("rgl")         #Visualizar en 3D
library("neuralnet")   #Redes Neuronales
library(adabag)        #Para aplicar la función Bagging en un árbol de clasificación
library(ipred)         #Bagging en árboles
library(randomForest)  #Paquete Bosques aletorios
library(ROCR) #Libreria para usar funciones de desempeño de los modelos
library(tidyr)

# ***************** Lectura de los datos ******************

load("datos")

dim(datos)
```

```
#***************** Seleccion de variables *****************


#Se seleccionan las variables que tienen un alto porcentaje de datos poblados y se
#eliminan 98 variables de las caules 73  con mas del 0.90 sin datos y 58 no cuentan con
#información relevante que puedan explicar si existe fraude

#Con las variables unificadas finalmente seleccionamos las variables que pueden explicar
#si existe fraude o no teniendo en que la base de datos cuenta con muchas variables que
#no son concluyentes para el resultado del modelo, se seleccionan las siguientes
#variables que podrian incidir en el resultado de la variable de respuesta

datos_trans <- datos[,c("varMedio","intIdPunto","intIdCanal","intMesFecSolicitud",
                        "varGenero","varEstCivil","varTipoVivienda","varEstrato",
                        "varTipoInmueble","varCodOcupacion","varNivEstudio",
                        "EstadoReal","Edad","PersonasACargo","Fraude")]

names(datos_trans)

# *************** Preparación de los datos ***************

# Detección si hay alguna fila incompleta
any(!complete.cases(datos_trans))
datos_trans <- na.omit(datos_trans)
sum(datos_trans$Edad == "NULL")
# Número de datos ausentes por variable
map_dbl(datos_trans, .f = function(x){sum(is.na(x))})2

#Se puede observar que en la variable edad hay datos ausentes, se debe validar si se
#imputan con el valor de la media de la edad o que tratamiento se hará, por el
#momento no se tendran en cuenta

# Eliminar datos ausentes
df_trans <- datos_trans[complete.cases(datos_trans), ]
df_trans <-df_trans[complete.cases(df_trans),]
# Verificar si existen campos vacios
sapply(df_trans, function(x) sum(is.na(x)))
# Convertir el campo edad en númerico
df_trans$Edad  <- as.numeric(df_trans$Edad)
df_trans  <- df_trans[!is.na(df_trans$Edad),]

#********** Verificar información de cada variable*********


df_trans %>%
  group_by(varGenero) %>%
  summarise(n = n())
```

```
df_trans$varGenero[df_trans$varGenero=="HOMBRE"]<-"Hombre"

df_trans %>%
  group_by(varGenero) %>%
  summarise(n = n())

#Se modifica la observación NOMBRE por Hombre, con el fin de homologar las categorias

df_trans %>%
  group_by(varEstCivil) %>%
  summarise(n = n())

#Se reemplaza el valor de 0 por Sin Campaña en la variable intIdCampana

df_trans %>%
  group_by(varMedio) %>%
  summarise(n = n())

df_trans %>%
  group_by(varEstrato) %>%
  summarise(n = n())

df_trans <- df_trans[df_trans$varEstrato != "-", ]
df_trans$varEstrato[df_trans$varEstrato==11]<-1

df_trans %>%
  group_by(varEstrato) %>%
  summarise(n = n())

#Se reemplaza el valor 11 por 1 ya que estrato 11 no existe, suponiendo que pretendian
#digitar 1 y no 11, asi como tambien se eliminan los estratos con valor "-"


df_trans %>%
  group_by(intIdCanal) %>%
  summarise(n = n())

df_trans %>%
  group_by(Edad) %>%
  summarise(n = n())

df_trans %>%
  group_by(Fraude) %>%
  summarise(n = n())

#En algunas variables se debe Agrupar las clases por nivel de significancia
```

```
#****************** Variable:varMedio *********************

#Las categorias Asesor, referido son las más representativas las demas categorias
#las Agrupamos en otros

df_trans$varMedio[df_trans$varMedio=="Impresos"]<-"Otros"
df_trans$varMedio[df_trans$varMedio=="Otro"]<-"Otros"
df_trans$varMedio[df_trans$varMedio=="Prensa"]<-"Otros"
df_trans$varMedio[df_trans$varMedio=="Radio"]<-"Otros"
df_trans$varMedio[df_trans$varMedio=="Sin Especificar"]<-"Otros"
df_trans$varMedio[df_trans$varMedio=="TV"]<-"Otros"

df_trans %>%
  group_by(Fraude,varMedio) %>%
  summarise(n = n())

#******************Variable:intIdPunto*********************

df_trans %>%
  group_by(Fraude,intIdPunto) %>%
  summarise(n = n())

#Las categorias 86004, 86101,86103 y 86800  son las más representativas las demas
#categorias las Agrupamos en otros

df_trans %>%
  group_by(Fraude,intIdPunto) %>%
  summarise(n = n())

#******************Variable:intIdCanal*******************

#Las categorias  1043,1045,  son las más representativas las demas categorias las
#Agrupamos en otros
df_trans %>%
  group_by(Fraude,intIdCanal) %>%
  summarise(n = n())

#**************Variable: intMesFecSolicitud***************

df_trans %>%
  group_by(Fraude,intMesFecSolicitud) %>%
  summarise(n = n())

#Las categorias Asesor, referido son las más representativas las demas categorias las
#Agrupamos en otros
```

```
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="1"]<-"Trim_1"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="2"]<-"Trim_1"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="3"]<-"Trim_1"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="4"]<-"Trim_2"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="5"]<-"Trim_2"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="6"]<-"Trim_2"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="7"]<-"Trim_3"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="8"]<-"Trim_3"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="9"]<-"Trim_3"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="10"]<-"Trim_4"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="11"]<-"Trim_4"
df_trans$intMesFecSolicitud[df_trans$intMesFecSolicitud=="12"]<-"Trim_4"

df_trans %>%
  group_by(Fraude,intMesFecSolicitud) %>%
  summarise(n = n())

#********************Variable:varEstCivil******************

df_trans %>%
  group_by(Fraude,varEstCivil) %>%
  summarise(n = n())

#Las categorias Asesor, referido son las más representativas las demas categorias las
#Agrupamos en otros

df_trans$varEstCivil[df_trans$varEstCivil=="DIVORCIADO"]<-"Otros"
df_trans$varEstCivil[df_trans$varEstCivil=="VIUDO"]<-"Otros"

df_trans %>%
  group_by(Fraude,varEstCivil) %>%
  summarise(n = n())

#********************Variable:varEstrato******************

df_trans %>%
  group_by(Fraude,varEstrato) %>%
  summarise(n = n())

df_trans$varEstrato[df_trans$varEstrato=="5"]<-"5_6"
df_trans$varEstrato[df_trans$varEstrato=="6"]<-"5_6"

df_trans %>%
  group_by(Fraude,varEstrato) %>%
  summarise(n = n())

#*****************Variable: varCodOcupacion***************
```

```
df_trans %>%
  group_by(Fraude,varCodOcupacion) %>%
  summarise(n = n())

#Las categorias Empleado, Independiente son las más representativas las demas categorias
#las Agrupamos en otros

df_trans$varCodOcupacion[df_trans$varCodOcupacion=="AmaDeCasa"]<-"Otros"
df_trans$varCodOcupacion[df_trans$varCodOcupacion=="Pensionado"]<-"Otros"

df_trans %>%
  group_by(Fraude,varCodOcupacion) %>%
  summarise(n = n())

#*****************Variable: varNivEstudio*****************


df_trans %>%
  group_by(Fraude,varNivEstudio) %>%
  summarise(n = n())

df_trans$varNivEstudio[df_trans$varNivEstudio=="POSGRADO"]<-"Otros"
df_trans$varNivEstudio[df_trans$varNivEstudio=="PRIMARIA"]<-"Otros"
df_trans$varNivEstudio[df_trans$varNivEstudio=="SIN ESPECIFICAR"]<-"Otros"
df_trans$varNivEstudio[df_trans$varNivEstudio=="TECNOLOGICO"]<-"Otros"

df_trans %>%
  group_by(Fraude,varNivEstudio) %>%
  summarise(n = n())

#*************** Variable: EstadoReal ********************

df_trans %>%
  group_by(Fraude,EstadoReal) %>%
  summarise(n = n())

#Las categorias  HABILITADO PARA NUEVO ESTUDIO, TARJETA ACTIVADA  son las más
#representativas las demas categorias las Agrupamos en otros

df_trans$EstadoReal[df_trans$EstadoReal=="PENDIENTE ACTIVAR REDEBAN"]<-"Otros"
df_trans$EstadoReal[df_trans$EstadoReal=="PENDIENTE CONFRONTA"]<-"Otros"
df_trans$EstadoReal[df_trans$EstadoReal=="PENDIENTE CREACION CLIENTE"]<-"Otros"
df_trans$EstadoReal[df_trans$EstadoReal=="TARJETA RESERVADA"]<-"Otros"

df_trans %>%
  group_by(Fraude,EstadoReal) %>%
  summarise(n = n())
```

```
#********************* Variable:Edad *********************

df_trans %>%
  group_by(Fraude,Edad) %>%
  summarise(n = n())

#Las categorias Asesor, referido son las más representativas las demas categorias las
#Agrupamos en otros

df_trans %>%
  group_by(Fraude,Edad) %>%
  summarise(n = n())

#**************Variable: PersonasACargo*******************

df_trans %>%
  group_by(Fraude,PersonasACargo) %>%
  summarise(n = n())

df_trans$PersonasACargo[df_trans$PersonasACargo=="2"]<-">=2"
df_trans$PersonasACargo[df_trans$PersonasACargo=="3"]<-">=2"
df_trans$PersonasACargo[df_trans$PersonasACargo=="4"]<-">=2"
df_trans$PersonasACargo[df_trans$PersonasACargo=="5"]<-">=2"
df_trans$PersonasACargo[df_trans$PersonasACargo=="MAS DE 5"]<-">=2"

df_trans %>%
  group_by(Fraude,PersonasACargo) %>%
  summarise(n = n())

#Ahora todos los registros estan completos para todas las variables, no se cuenta
#con #NAs y campos vacios.

#**************Variable: intMesFecSolicitud***************
# **********Comnvertir a variables categoricas*************


str(df_trans)
df_trans$Fraude<-as.factor((df_trans$Fraude))
df_trans$varMedio<-as.factor((df_trans$varMedio))
df_trans$intIdCanal<-as.factor((df_trans$intIdCanal))
df_trans$varGenero<-as.factor((df_trans$varGenero))
df_trans$varEstCivil<-as.factor((df_trans$varEstCivil))
df_trans$varEstrato<-as.factor((df_trans$varEstrato))
df_trans$intIdPunto<-as.factor((df_trans$intIdPunto))
df_trans$intMesFecSolicitud<-as.factor((df_trans$intMesFecSolicitud))
df_trans$varTipoVivienda<-as.factor((df_trans$varTipoVivienda))
df_trans$varTipoInmueble<-as.factor((df_trans$varTipoInmueble))
```

```
df_trans$varCodOcupacion<-as.factor((df_trans$varCodOcupacion))
df_trans$varNivEstudio<-as.factor((df_trans$varNivEstudio))
df_trans$varNivEstudio<-as.factor((df_trans$varNivEstudio))
df_trans$EstadoReal<-as.factor((df_trans$EstadoReal))

tbla <- df_trans %>%
  group_by(varMedio) %>%
  summarise(n = n())

df_trans %>%
  group_by(varMedio) %>%
  summarise(n = n())

#Aplicar metodologia para Selection de variables: definir el mejor subconjunto de
#variables predictoras que expliquen si existe fraude o no


#*****************Todos los conjuntos*********************


require(leaps)
Data <- as.data.frame(df_trans)

regfit.full<-regsubsets(Fraude~., data=Data, nvmax=15)
reg.summary<-summary(regfit.full)

reg.summary$rss

par(mfrow =c(1,2))
plot(reg.summary$rss ,xlab="Número de variables",ylab="SS_Res",type="l")
plot(reg.summary$adjr2 ,xlab ="Número de variables",ylab="R2 Ajustado",
     type="l")
a1<-which.max(reg.summary$adjr2)
points(a1, reg.summary$adjr2[a1], col ="red",cex =2, pch =20)


#plot(reg.summary$cp, xlab = "Numero de variables", ylab = "R2 ajustado", type = "b")
par(mfrow =c(1,2))
plot(reg.summary$cp ,xlab ="Número de variables", ylab="Cp", type="l")
a2<-which.min(reg.summary$cp)
points(a2, reg.summary$cp[a2], col ="red",cex =2, pch =20)
a3<-which.min(reg.summary$bic)
plot(reg.summary$bic ,xlab="Número de variables",ylab=" BIC",type="l")
points(a3, reg.summary$bic[a3], col =" red",cex =2, pch =20)

plot(regfit.full, scale ="r2")
plot(regfit.full, scale ="adjr2")
plot(regfit.full, scale ="Cp")
```

```
plot(regfit.full, scale ="bic")
coef(regfit.full,15)


#**************Selección hacia adelante********************


regfit.fwd<-regsubsets(Fraude~.,data=Data,
                       nvmax=15, method = "forward")
summary(regfit.fwd)
plot(regfit.fwd, scale ="r2")
plot(regfit.fwd, scale ="adjr2")
plot(regfit.fwd, scale ="Cp")
plot(regfit.fwd, scale ="bic")


#**************Selección hacia atrás************************


regfit.bwd<-regsubsets(Fraude~.,data=Data,
                       nvmax=15, method = "backward")
summary(regfit.bwd)

plot(regfit.bwd, scale ="r2")
plot(regfit.bwd, scale ="adjr2")
plot(regfit.bwd, scale ="Cp")
plot(regfit.bwd, scale ="bic")


#**************Selección Stepwise************************

#*#Modelo sin Variables independientes
Mod_vacio <- glm(Fraude ~1, data = over_train, family = binomial(link = "logit"))
summary(Mod_vacio)

#Modelo con todas las Variables independientes
Mod_compl <- glm(Fraude ~., data = over_train, family = binomial(link = "logit"))
summary(Mod_compl)


#Particionamos la base de datos con todas las variables en train y testing, utilizamos la
#función stratified para garantizar el eqquilibrio de las clases en cada una de las
#variables.

library(splitstackshape)
set.seed(123)

datapart   <- stratified(Data, c("Fraude"),.8,bothSets = TRUE)
```

```
data_train_tv    <- datapart$SAMP1
data_testing_tv  <- datapart$SAMP2

addmargins(table(Data$Fraude))
round(prop.table(table(Data$Fraude))*100,2)

addmargins(table(data_train_tv$Fraude))
round(prop.table(table(data_train_tv$Fraude))*100,2)

addmargins(table(data_testing_tv$Fraude))
round(prop.table(table(data_testing_tv$Fraude))*100,2)2



#******Aplicar metodologia para datos desbalanceadaos*********************

contrasts(Data$Fraude)
table(Data$Fraude)
round(prop.table(table(Data$Fraude)),3)*100


#**************Balanceo de Datos Undersampling*****************************

# Undersampling (under_train)

set.seed(123)
under_train <- downSample(x = Data[,c(1:14)],
                          y = Data$Fraude,
                          yname="Fraude")

addmargins(table(under_train$Fraude))

#**************Partición de datos training y testing*****************************

datapartic_under   <- stratified(under_train, c("Fraude"),.8,bothSets = TRUE)
data_train_under    <- datapartic_under$SAMP1
data_testing_under  <- datapartic_under$SAMP2

round(prop.table(table(data_train_under$Fraude))*100,2)

round(prop.table(table(data_testing_under$Fraude))*100,2)


#**************Balanceo de Datos oversampling**********************************


#OverSampling(over_train)
```

```
set.seed(123)
over_train <- upSample(x = Data[, c(1:14)],
                       y = Data$Fraude,
                       yname="Fraude")

addmargins(table(over_train$Fraude))

#**************Partición de datos training y testing*************************


datapartic_over   <- stratified(over_train, c("Fraude"),.8,bothSets = TRUE)
data_train_over    <- datapartic_over$SAMP1
data_testing_over  <- datapartic_over$SAMP2


addmargins(table(data_train_over$Fraude))
round(prop.table(table(data_train_over$Fraude))*100,2)

addmargins(table(data_testing_over$Fraude))
round(prop.table(table(data_testing_over$Fraude))*100,2)

#********aplicacion del modelo Logistico sin balanceo de datos************

library(caret)
library(glmnet)

mod_grl<- glm(data_train_tv$Fraude ~ ., family = binomial, data= data_train_tv)
summary(mod_grl)


cioef <- exp(coefficients(mod_grl))

#probabilidades estimadas por individuo

pre_grl <- predict(mod_grl, newdata = data_testing_tv, type = "response")
head(pre_grl)

#********Entrenamiento con datos de prueba y testing ************

ctrl <- trainControl(method="cv", number=10)
set.seed(123)
modelo_orig_tv   <- train(Fraude ~ .,
                          data = data_train_tv,
                          method="glm", family="binomial",
                          trControl = ctrl,
                          metric="Accuracy")

clase.modelo_orig_tv <- predict(modelo_orig_tv,newdata = data_testing_tv )
```

```
#***************Determinación del umbral ()***************


#Determinación del umbral
library(ROCR)
RocPred_gral <-  ROCR::prediction(pre_grl,data_testing_tv$Fraude)
ROCpref_gral <-  performance(RocPred_gral,"tpr","fpr")

par(mfrow = c(1,2))

plot(ROCpref_gral,colorize=TRUE,type="l")
abline(a=0,b=1)

#punto de corte

opt.cut = function(perf, pred){
  cut.ind = mapply(FUN=function(x, y, p){
    d = (x - 0)^2 + (y-1)^2
    ind = which(d == min(d))
    c(sensitivity = y[[ind]], specificity = 1-x[[ind]],
      cutoff = p[[ind]])
  }, perf@x.values, perf@y.values, pred@cutoffs)
}
pc<-opt.cut(ROCpref_gral, RocPred_gral)

ROCpref_gral_sen <-  performance(RocPred_gral,"sens")
ROCpref_gral_esp <-  performance(RocPred_gral,"spec")

plot(ROCpref_gral_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_gral_esp,lwd=2,col="green3")
abline(v= pc[3],lwd=3,col="blue",lty="dotted")

par(mfrow = c(1, 1))

#*********Matriz de confusion a partir del punto de corte optimo**********

clas_grl <- ifelse(pre_grl >  pc[3],1,0)
mean(clas_grl == data_train_tv$Fraude)
table(pre_grl = clas_grl, actual = data_testing_tv$Fraude)

performance_data<-data.frame(observed=data_testing_tv$Fraude,
                             predicted= clas_grl)
total <- nrow(performance_data)
fraude_ <- sum(performance_data$observed=="1")
nofraude_ <- sum(performance_data$observed=="0")
predicted_fraude <- sum(performance_data$predicted=="1")
```

```
predicted_nofraude <- sum(performance_data$predicted=="0")
tp<-sum(performance_data$observed=="1" & performance_data$predicted=="1")
tn<-sum(performance_data$observed=="0" & performance_data$predicted=="0")
fp<-sum(performance_data$observed=="0" & performance_data$predicted=="1")
fn<-sum(performance_data$observed=="1" & performance_data$predicted=="0")

balanceo_tv <- "Sin balanceo"
accuracy_tv <- (tp+tn)/total
error_rate <- (fp+fn)/total
sensibilidad_tv <- tp/fraude_
especificidad_tv <- tn/nofraude_
precision <- tp/predicted_fraude
npv <- tn / predicted_nofraude


#********aplicacion del modelo Logistico con balanceo undersampling***********
library(glmnet)
set.seed(123)

mod_bal_under <- glm(Fraude ~., data = data_train_under, family = binomial(link = "logit"))

summary(mod_bal_under)

cioef <- exp(coefficients(mod_bal_under))

#probabilidades estimadas por individuo

pre_bal_under <- predict(mod_bal_under, newdata = data_testing_under, type = "response")
head(pre_bal_under)

#********Entrenamiento con datos de prueba y testing ************

ctrl <- trainControl(method="cv", number=10)
set.seed(123)
modelo_bal_under   <- train(Fraude ~ .,
                            data = data_train_under,
                            method="glm", family="binomial",
                            trControl = ctrl,
                            metric="Accuracy")

#****************Determinación del umbral ()***************

#Determinación del umbral
library(ROCR)
set.seed(123)

RocPred_under <-  ROCR::prediction(pre_bal_under,data_testing_under$Fraude)
ROCpref_under <-  performance(RocPred_under,"tpr","fpr")
```

```
par(mfrow =c(1,2))

plot(ROCpref_under,colorize=TRUE,type="l")
abline(a=0,b=1)
# Punto de corte óptimo para grafica #1
cost.perf_under <- performance(RocPred_under, measure ="cost")
opt.cut_under  <- RocPred_under@cutoffs[[1]][which.min(cost.perf_under@y.values[[1]])]
#coordenadas del punto de corte óptimo
x<-ROCpref_under@x.values[[1]][which.min(cost.perf_under@y.values[[1]])]
y<-ROCpref_under@y.values[[1]][which.min(cost.perf_under@y.values[[1]])]
#points(x,y, pch=20, col="red")
cat("AUC:", AUCaltura[[1]])
cat("Punto de corte óptimo:",opt.cut_under)


#punto de corte optimo para grafica 2

opt.cut = function(perf, pred){
  cut.ind = mapply(FUN=function(x, y, p){
    d = (x - 0)^2 + (y-1)^2
    ind = which(d == min(d))
    c(sensitivity = y[[ind]], specificity = 1-x[[ind]],
      cutoff = p[[ind]])
  }, perf@x.values, perf@y.values, pred@cutoffs)
}
pc_u<-opt.cut(ROCpref_under, RocPred_under)

ROCpref_und_sen <-  performance(RocPred_under,"sens")
ROCpref_und_esp <-  performance(RocPred_under,"spec")

plot(ROCpref_und_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_und_esp,lwd=2,col="blue")
abline(v= 0.72,lwd=3,col="gray62",lty="dotted")

par(mfrow =c(1,1))


#*********Matriz de confusion a partir del punto de corte optimo**********

# matriz de confusión

clas_bal_under <- ifelse(pre_bal_under >0.72,1,0)
mean(clas_bal_under == data_train_under$Fraude)
table(pre_bal_under = clas_bal_under, actual = data_testing_under$Fraude)
```

```
performance_data<-data.frame(observed=data_testing_under$Fraude,
                             predicted= clas_bal_under)
total <- nrow(performance_data)
fraude_ <- sum(performance_data$observed=="1")
nofraude_ <- sum(performance_data$observed=="0")
predicted_fraude <- sum(performance_data$predicted=="1")
predicted_nofraude <- sum(performance_data$predicted=="0")
tp<-sum(performance_data$observed=="1" & performance_data$predicted=="1")
tn<-sum(performance_data$observed=="0" & performance_data$predicted=="0")
fp<-sum(performance_data$observed=="0" & performance_data$predicted=="1")
fn<-sum(performance_data$observed=="1" & performance_data$predicted=="0")

balanceo_und <- "Undersampling"
accuracy_und <- (tp+tn)/total
error_rate <- (fp+fn)/total
sensibilidad_und <- tp/fraude_
especificidad_und <- tn/nofraude_
precision <- tp/predicted_fraude
npv <- tn / predicted_nofraude
data.frame(sensibilidad_und,especificidad_und,accuracy_und)

comparacion_und <- data.frame(balanceo_und,sensibilidad_und,
                              especificidad_und,accuracy_und)
comparacion_und

#********aplicacion del modelo Logistico con balanceo Oversampling************
#*
#*library(caret)
library(glmnet)

set.seed(123)

mod_bal_over <- glm(Fraude ~., data = data_train_over, family = binomial(link = "logit"))

summary(mod_bal_over)

cioef <- exp(coefficients(mod_bal_over))

#probabilidades estimadas por individuo

pre_bal_over <- predict(mod_bal_over, newdata = data_testing_over, type = "response")
head(pre_bal_over)


#********Entrenamiento con datos de prueba y testing ************

ctrl <- trainControl(method="cv", number=10)
set.seed(123)
```

```
modelo_bal_over   <- train(Fraude ~ .,
                           data = data_train_over,
                           method="glm", family="binomial",
                           trControl = ctrl,
                           metric="Accuracy")


#****************Determinación del umbral ()****************

#Determinación del umbral
library(ROCR)

RocPred_over <-  ROCR::prediction(pre_bal_over,data_testing_over$Fraude)
ROCpref_over <-  performance(RocPred_over,"tpr","fpr")

par(mfrow =c(1,2))

plot(ROCpref_over,colorize=TRUE,type="l")
abline(a=0,b=1)
# Punto de corte óptimo para grafica #1

cost.perf_over <- performance(RocPred_over, measure ="cost")
opt.cut_over   <- RocPred_over@cutoffs[[1]][which.min(cost.perf_over@y.values[[1]])]
#coordenadas del punto de corte óptimo
x<-ROCpref_over@x.values[[1]][which.min(cost.perf_over@y.values[[1]])]
y<-ROCpref_over@y.values[[1]][which.min(cost.perf_over@y.values[[1]])]
#points(x,y, pch=20, col="red")
cat("AUC:", AUCaltura[[1]])
cat("Punto de corte óptimo:",opt.cut_over)

#punto de corte optimo para grafica 2

opt.cut = function(perf, pred){
  cut.ind = mapply(FUN=function(x, y, p){
    d = (x - 0)^2 + (y-1)^2
    ind = which(d == min(d))
    c(sensitivity = y[[ind]], specificity = 1-x[[ind]],
      cutoff = p[[ind]])
  }, perf@x.values, perf@y.values, pred@cutoffs)
}
pc_o<-opt.cut(ROCpref_over, RocPred_over)

ROCpref_ove_sen <-  performance(RocPred_over,"sens")
ROCpref_ove_esp <-  performance(RocPred_over,"spec")

plot(ROCpref_ove_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_ove_esp,lwd=2,col="blue")
```

```
abline(v= pc_o[3],lwd=3,col="gray62",lty="dotted")

par(mfrow =c(1,1))

#*********Matriz de confusion a partir del punto de corte optimo**********

# matriz de confusión

clas_bal_over <- ifelse(pre_bal_over > pc_o[3],1,0)
mean(clas_bal_over == data_train_over$Fraude)
table(pre_bal_over = clas_bal_over, actual = data_testing_over$Fraude)

confusionMatrix(as.factor(clas_bal_over),data_testing_over$Fraude)

performance_data<-data.frame(observed=data_testing_over$Fraude,
                             predicted= clas_bal_over)
total <- nrow(performance_data)
fraude_ <- sum(performance_data$observed=="1")
nofraude_ <- sum(performance_data$observed=="0")
predicted_fraude <- sum(performance_data$predicted=="1")
predicted_nofraude <- sum(performance_data$predicted=="0")
tp<-sum(performance_data$observed=="1" & performance_data$predicted=="1")
tn<-sum(performance_data$observed=="0" & performance_data$predicted=="0")
fp<-sum(performance_data$observed=="0" & performance_data$predicted=="1")
fn<-sum(performance_data$observed=="1" & performance_data$predicted=="0")

balanceo_over <- "Oversampling"
accuracy_over <- (tp+tn)/total
error_rate <- (fp+fn)/total
sensibilidad_over <- tp/fraude_
especificidad_over <- tn/nofraude_
precision <- tp/predicted_fraude
npv <- tn / predicted_nofraude
data.frame(sensibilidad_over,especificidad_over,precision)

comparacion_over <- data.frame(balanceo_over,sensibilidad_over,
                               especificidad_over,accuracy_over)
comparacion_over



#********aplicacion del modelo äbol de clasificación sin balanceo ************

# Funcion para la matriz de confusion
confusion<-function(real,scoring,umbral){
  conf<-table(real,scoring>=umbral)
  if(ncol(conf)==2) return(conf) else return(NULL)
}
```

```
## Funcion para métricas de los modelos

  metricas<-function(matriz_conf){
  acierto <- (matriz_conf[1,1] + matriz_conf[2,2]) / sum(matriz_conf) *100
  precision <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[1,2]) *100
  cobertura <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[2,1]) *100
  sensibilidad <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[2,1]) *100
  especificidad <- matriz_conf[2,1] / (matriz_conf[2,1] + matriz_conf[2,2]) *100
  F1 <- 2*precision*cobertura/(precision+cobertura)
  salida<-c(acierto,precision,cobertura,F1)
  return(salida)}

  ## Función para probar distintos umbrales

  umbrales<-function(real,scoring){
    umbrales<-data.frame(umbral=rep(0,times=19),acierto=rep(0,times=19),
    precision=rep(0,times=19),cobertura=rep(0,times=19),F1=rep(0,times=19))
    cont <- 1
    for (cada in seq(0.05,0.95,by = 0.05)){
      datos<-metricas(confusion(real,scoring,cada))
      registro<-c(cada,datos)
      umbrales[cont,]<-registro
      cont <- cont + 1
    }
    return(umbrales)
  }

  # Paso 1: Primer modelo árbol de clasificación sin balanceo de datos

  require(rpart)
  require(rpart.plot)

  set.seed(123)
  mod.rpart_tv_1 <- rpart(Fraude~.,data=data_train_tv, method="class",
  parms = list(split = "information"),control = rpart.control(cp = 0.00001))


  rpart.plot(mod.rpart_tv_1, type=3, digits = 3,fallen.leaves = TRUE)

  printcp(mod.rpart_tv_1)
  plotcp(mod.rpart_tv_1)

  ## Paso 2: Segundo modelo árbol de clasificación sin balanceo de datos


  set.seed(123)
  mod.rpart_tv_2 <- rpart(Fraude~.,data=data_train_tv, method="class",
```

```
parms = list(split = "information"),control = rpart.control(cp = 0.0086331,maxdepth = 7))

rpart.plot(mod.rpart_tv_2, type=3, digits = 3,fallen.leaves = TRUE)

printcp(mod.rpart_tv_2)
plotcp(mod.rpart_tv_2)

rpart.pred_tv <- predict(mod.rpart_tv_2, data_testing_tv,type="class")
result_rpart_tv<- caret::confusionMatrix(rpart.pred_tv,
                                         data_testing_tv$Fraude,
                                         positive="1")
result_rpart_tv

summary(mod.rpart_tv_2)

#********************Prediccion del modelo************************

dt_score<-predict(mod.rpart_tv_2, data_testing_tv,type = 'prob')[,2]
head(dt_score)

plot(dt_score ~ data_testing_tv$Fraude)


#****************Determinación del umbral ()***************

# Con la función umbrales probamos diferentes cortes y escogemos el mejor

umb_bt<-umbrales(data_testing_tv$Fraude,dt_score)
umb_bt
umbral_final_rf2<-umb_bt[which.max(umb_bt$F1),1]
umbral_final_rf2

#*********Matriz de confusion a partir del punto de corte optimo**********

confusion(data_testing_tv$Fraude,dt_score,umbral_final_rf2)
rf2_metricas<-filter(umb_bt,umbral==umbral_final_rf2)
rf2_metricas

#creamos el objeto prediction
rf2_prediction<-prediction(dt_score,data_testing_tv$Fraude)
#visualizamos la ROC
#roc(rf2_prediction)

RocPred_arb_tv <-  ROCR::prediction(dt_score,data_testing_tv$Fraude)
ROCpref_arb_tv <-  performance(RocPred_arb_tv,"tpr","fpr")

par(mfrow =c(1,2))
```

```
plot(ROCpref_arb_tv,colorize=TRUE,type="l")
abline(a=0,b=1)


ROCpref_arb_tv_sen <-  performance(RocPred_arb_tv,"sens")
ROCpref_arb_tv_esp <-  performance(RocPred_arb_tv,"spec")

plot(ROCpref_arb_tv_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_arb_tv_esp,lwd=2,col="blue")
abline(v= umbral_final_rf2,lwd=3,col="gray62",lty="dotted")

par(mfrow =c(1,1))


#***aplicacion del modelo äbol de clasificación con balanceo undersampling****7

# Paso 1: Primer modelo árbol de clasificación sin balanceo de datos

require(rpart)
require(rpart.plot)

set.seed(123)
mod.rpart_under_1 <- rpart(Fraude~.,data=data_train_under,
method="class",parms = list(split = "information"),
control = rpart.control(cp = 0.00001))

rpart.plot(mod.rpart_under_1, type=3, digits = 3,fallen.leaves = TRUE)

printcp(mod.rpart_under_1)
plotcp(mod.rpart_under_1)

## Paso 2: Segundo modelo árbol de clasificación sin balanceo de datos

set.seed(123)
mod.rpart_under_2 <- rpart(Fraude~.,data=data_train_under, method="class",
                           parms = list(split = "information"),
                           control = rpart.control(cp = 0.021583,maxdepth = 7))

rpart.plot(mod.rpart_under_2, type=3, digits = 3,fallen.leaves = TRUE)

printcp(mod.rpart_under_2)
plotcp(mod.rpart_under_2)




#********************Prediccion del modelo************************
```

```
dt_score<-predict(mod.rpart_under_2, data_testing_under,type = 'prob')[,2]
head(dt_score)

# Con la función umbrales probamos diferentes cortes y escogemos el mejor
umb_bt<-umbrales(data_testing_under$Fraude,dt_score)
umb_bt

umbral_final_rf2<-umb_bt[which.max(umb_bt$F1),1]
umbral_final_rf2

#*********Matriz de confusion a partir del punto de corte optimo**********

confusion(data_testing_under$Fraude,dt_score,umbral_final_rf2)
rf2_metricas<-filter(umb_bt,umbral==umbral_final_rf2)
rf2_metricas
#creamos el objeto prediction
rf2_prediction<-prediction(dt_score,data_testing_under$Fraude)
#visualizamos la ROC
#roc(rf2_prediction)

RocPred_arb_und <-  ROCR::prediction(dt_score,data_testing_under$Fraude)
ROCpref_arb_und <-  performance(RocPred_arb_und,"tpr","fpr")

par(mfrow =c(1,2))

plot(ROCpref_arb_und,colorize=TRUE,type="l")
abline(a=0,b=1)


ROCpref_arb_und_sen <-  performance(RocPred_arb_und,"sens")
ROCpref_arb_und_esp <-  performance(RocPred_arb_und,"spec")

plot(ROCpref_arb_und_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_arb_und_esp,lwd=2,col="blue")
abline(v= umbral_final_rf2,lwd=3,col="gray62",lty="dotted")

par(mfrow =c(1,1))



#***aplicacion del modelo äbol de clasificación con balanceo oversampling****

# Paso 1: Primer modelo árbol de clasificación sin balanceo de datos
require(rpart)
require(rpart.plot)

set.seed(123)
```

```
mod.rpart_over_1 <- rpart(Fraude~.,data=data_train_over,
method="class",parms = list(split = "information"),
control = rpart.control(cp = 0.00001))

rpart.plot(mod.rpart_over_1, type=3, digits = 3,fallen.leaves = TRUE)

printcp(mod.rpart_over_1)
plotcp(mod.rpart_over_1)

## Paso 2: Segundo modelo árbol de clasificación sin balanceo de datos

set.seed(123)
mod.rpart_over_2 <- rpart(Fraude~.,data=data_train_over,
method="class",parms = list(split = "information"),
control = rpart.control(cp = 0.000010000  ,maxdepth = 4))

rpart.plot(mod.rpart_over_2, type=3, digits = 3,fallen.leaves = TRUE)

printcp(mod.rpart_over_2)
plotcp(mod.rpart_over_2)

#********************Prediccion del modelo************************


dt_score<-predict(mod.rpart_over_2, data_testing_over,type = 'prob')[,2]
head(dt_score)
plot(dt_score ~ data_testing_over$Fraude)

# Con la función umbrales probamos diferentes cortes y escogemos el mejor****

umb_bt<-umbrales(data_testing_over$Fraude,dt_score)
umb_bt
umbral_final_rf2<-umb_bt[which.max(umb_bt$F1),1]
umbral_final_rf2

#*********Matriz de confusion a partir del punto de corte optimo**********

confusion(data_testing_over$Fraude,dt_score,umbral_final_rf2)
rf2_metricas<-filter(umb_bt,umbral== umbral_final_rf2)
rf2_metricas

#creamos el objeto prediction
rf2_prediction<-prediction(dt_score,data_testing_over$Fraude)
#visualizamos la ROC
#roc(rf2_prediction)

RocPred_arb_over <-  ROCR::prediction(dt_score,data_testing_over$Fraude)
ROCpref_arb_over <-  performance(RocPred_arb_over,"tpr","fpr")
```

```
par(mfrow =c(1,2))

plot(ROCpref_arb_over,colorize=TRUE,type="l")
abline(a=0,b=1)


ROCpref_arb_over_sen <-  performance(RocPred_arb_over,"sens")
ROCpref_arb_over_esp <-  performance(RocPred_arb_over,"spec")

plot(ROCpref_arb_over_sen,lwd=2,col="cyan")
par(new=T)
plot(ROCpref_arb_over_esp,lwd=2,col="blue")
abline(v= umbral_final_rf2,lwd=3,col="gray62",lty="dotted")

par(mfrow =c(1,1))


#**Maquinas de soporte kernel LINEAL sin balanceo de datos****

#**aplicacion del modelo sin balanceo de datos****



#** Optimización de hiperparámetros mediante validación cruzada 10-fold**


datapartic_tv_prueba   <- stratified(Data, c("Fraude"),.05,bothSets = TRUE)
data_train_tv_prueba    <- datapartic_tv_prueba$SAMP1
datapartic_tv_prueba   <- stratified(data_train_tv_prueba, c("Fraude"),.8,bothSets = TRUE)
data_train_tv_prueba    <- datapartic_tv_prueba$SAMP1
data_testing_tv_prueba  <- datapartic_tv_prueba$SAMP2


#Optimización de hiperparámetros mediante validación cruzada 10-fold

set.seed(123)
hpar_train_tv_lin <- tune(svm, Fraude~ . , data = data_train_tv,
                          kernel = "linear",
                          ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 15, 20)),
                          scale = TRUE)
summary(hpar_train_tv_lin)

mod_svm_tv_lin_best <- hpar_train_tv_lin$best.model
summary(mod_svm_tv_lin_best)

#Aplicación del modelo SVM datos de testing
```

```
set.seed(123)
pred_svmlin_tv_test <- predict(mod_svm_tv_lin, data_testing_tv_prueba)


table(pred_svmlin_tv_test,data_testing_tv_prueba$Fraude,dnn=c("Predicho","Actual"))


matconf_smvlin_tv_test<- caret::confusionMatrix(pred_svmlin_tv_test,
                                                data_testing_tv_prueba$Fraude,
                                                positive="1")

balanceo_svm_tv_lin      <- c("Maquinas de Soporte")
sensibilidad_svm_tv_lin  <- c(matconf_smvlin_tv_test$byClass["Sensitivity"])
especificidad_svm_tv_lin  <- c(matconf_smvlin_tv_test$byClass["Specificity"])
accuracy_svm_tv_lin      <- c(matconf_smvlin_tv_test$overall["Accuracy"])
accuracy_bal_svm_tv_lin   <- c(matconf_smvlin_tv_test$byClass["Balanced Accuracy"])
comparacion_svm_tv_lin <- data.frame(balanceo_svm_tv_lin,
sensibilidad_svm_tv_lin ,especificidad_svm_tv_lin ,accuracy_svm_tv_lin ,accuracy_bal_svm_tv_lin )

comparacion_svm_tv_lin


#**Maquinas de soporte kernel LINEAL balanceo Undersampling****

#**aplicacion del modelo con datos balanceados (Undersampling)****

ata_train_under$Edad<-as.factor((data_train_under$Edad))
data_train_under$PersonasACargo<-as.factor((data_train_under$PersonasACargo))

#Optimización de hiperparámetros mediante validación cruzada 10-fold

set.seed(123)

hpar_train_under_lin <- tune(svm, Fraude~ . , data = data_train_under,
                             kernel = "linear",
                             ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 15, 20)),
                             scale = TRUE)
#Resumen de la validación cruzada de tuning


summary(hpar_train_under_lin)


#** Optimización de hiperparámetros mediante validación cruzada 10-fold**

mod_svm_under_lin_best <- hpar_train_under_lin$best.model
summary(mod_svm_under_lin_best)
```

```
#Aplicación del modelo SVM datos de testing


set.seed(123)
pred_svmlin_under_test <- predict(mod_svm_under_lin, data_testing_under)


table(pred_svmlin_under_test,data_testing_under$Fraude,dnn=c("Predicho","Actual"))


matconf_smvlin_under_test<- caret::confusionMatrix(pred_svmlin_under_test,
                                                   data_testing_under$Fraude,
                                                   positive="1")

balanceo_svm_under_lin      <- c("Maquinas de Soporte")
sensibilidad_svm_under_lin  <- c(matconf_smvlin_under_test$byClass["Sensitivity"])
especificidad_svm_under_lin  <- c(matconf_smvlin_under_test$byClass["Specificity"])
accuracy_svm_under_lin      <- c(matconf_smvlin_under_test$overall["Accuracy"])
accuracy_bal_svm_under_lin   <- c(matconf_smvlin_under_test$byClass["Balanced Accuracy"])
comparacion_svm_under_lin <- data.frame(balanceo_svm_under_lin,sensibilidad_svm_under_lin,
especificidad_svm_under_lin ,accuracy_svm_under_lin ,accuracy_bal_svm_under_lin )

comparacion_svm_under_lin


#**Maquinas de soporte kernel LINEAL balanceo oversampling****
#*
#*Particion de la base de datos
datapartic_over_prueba   <- stratified(over_train, c("Fraude"),.05,bothSets = TRUE)
data_train_over_prueba   <- datapartic_over_prueba$SAMP1
datapartic_over_prueba   <- stratified(data_train_over_prueba, c("Fraude"),.8,bothSets = TRUE)
data_train_over_prueba    <- datapartic_over_prueba$SAMP1
data_testing_over_prueba  <- datapartic_over_prueba$SAMP2

#**aplicacion del modelo con datos balanceados (oversampling)****

#Optimización de hiperparámetros mediante validación cruzada 10-fold
set.seed(123)
hpar_train_over_lin <- tune(svm, Fraude~ . , data = data_train_over_prueba,
                            kernel = "linear",
                            ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 15, 20)),
                            scale = TRUE)
#Resumen de la validación cruzada de tuning

summary(hpar_train_over_lin)


#** Optimización de hiperparámetros mediante validación cruzada 10-fold**
```

```
mod_svm_over_lin_best <- hpar_train_over_lin$best.model
summary(mod_svm_over_lin_best)

#Aplicación del modelo SVM datos de testing

set.seed(123)
pred_svmlin_over_test <- predict(mod_svm_over_lin, data_testing_over_prueba)


#table(data_testing_over_prueba$Fraude,pred_svmlin_over_test,dnn=c("Actual","Predicho"))

table(pred_svmlin_over_test,data_testing_over_prueba$Fraude,dnn=c("Predicho","Actual"))

matconf_smvlin_over_test<- caret::confusionMatrix(pred_svmlin_over_test,
                                                  data_testing_over_prueba$Fraude,
                                                  positive="1")

balanceo_svm_over_lin      <- c("Maquinas de Soporte")
sensibilidad_svm_over_lin  <- c(matconf_smvlin_over_test$byClass["Sensitivity"])
especificidad_svm_over_lin  <- c(matconf_smvlin_over_test$byClass["Specificity"])
accuracy_svm_over_lin      <- c(matconf_smvlin_over_test$overall["Accuracy"])
accuracy_bal_svm_over_lin   <- c(matconf_smvlin_over_test$byClass["Balanced Accuracy"])
comparacion_svm_over_lin <- data.frame(balanceo_svm_over_lin,sensibilidad_svm_over_lin,
especificidad_svm_over_lin ,accuracy_svm_over_lin ,accuracy_bal_svm_over_lin )

comparacion_svm_over_lin



#****Maquinas de soporte kernel POLINOMIAL sin balanceo****
 #*Aplicación del modelo SVM datos de entrenamiento

 set.seed(123)
 mod_svm_tv_pol <- svm(Fraude ~ ., data = data_train_tv,
 kernel = "polynomial", cost = 10, scale = TRUE)

 summary(mod_svm_tv_pol)

 set.seed(123)
 pred_svmpol_tv_train <- predict(mod_svm_tv_pol, data_train_tv)


 matconf_smvpol_tv_train <- table(predicho  = pred_svmpol_tv_train,
 real = data_train_tv$Fraude)
 matconf_smvpol_tv_train

 #*****Aplicación del modelo SVM datos de testing**********
```

```
set.seed(123)
pred_svmpol_tv_test <- predict(mod_svm_tv_pol, data_testing_tv)



table(pred_svmpol_tv_test,data_testing_tv$Fraude,dnn=c("Predicho","Actual"))


matconf_smvpol_tv_test<- caret::confusionMatrix(pred_svmpol_tv_test,
                                                data_testing_tv$Fraude,
                                                positive="1")

balanceo_svm_tv_pol      <- c("Maquinas de Soporte")
sensibilidad_svm_tv_pol  <- c(matconf_smvpol_tv_test$byClass["Sensitivity"])
especificidad_svm_tv_pol  <- c(matconf_smvpol_tv_test$byClass["Specificity"])
accuracy_svm_tv_pol      <- c(matconf_smvpol_tv_test$overall["Accuracy"])
accuracy_bal_svm_tv_pol   <- c(matconf_smvpol_tv_test$byClass["Balanced Accuracy"])
comparacion_svm_tv_pol <- data.frame(balanceo_svm_tv_pol,
sensibilidad_svm_tv_pol ,especificidad_svm_tv_pol ,
accuracy_svm_tv_pol ,accuracy_bal_svm_tv_pol )

comparacion_svm_tv_pol


#******Maquinas de soporte kernel POLINOMIAL balanceo Undersampling**

#*Aplicación del modelo SVM datos de entrenamiento


set.seed(123)
mod_svm_under_pol <- svm(Fraude ~ ., data = data_train_under,
kernel = "polynomial", cost = 10, scale = TRUE)

summary(mod_svm_under_pol)

set.seed(123)
pred_svmpol_under_train <- predict(mod_svm_under_pol, data_train_under)


matconf_smvpol_under_train <- table(predicho  = pred_svmpol_under_train,
real = data_train_under$Fraude)


#*****Aplicación del modelo SVM datos de testing**********

set.seed(123)
pred_svmpol_under_test <- predict(mod_svm_under_pol, data_testing_under)
```

```
table(pred_svmpol_under_test,data_testing_under$Fraude,dnn=c("Predicho","Actual"))

matconf_smvpol_under_test<- caret::confusionMatrix(pred_svmpol_under_test,
                                                   data_testing_under$Fraude,
                                                   positive="1")

balanceo_svm_under_pol      <- c("Maquinas de Soporte")
sensibilidad_svm_under_pol  <- c(matconf_smvpol_under_test$byClass["Sensitivity"])
especificidad_svm_under_pol  <- c(matconf_smvpol_under_test$byClass["Specificity"])
accuracy_svm_under_pol      <- c(matconf_smvpol_under_test$overall["Accuracy"])
accuracy_bal_svm_under_pol   <- c(matconf_smvpol_under_test$byClass["Balanced Accuracy"])
comparacion_svm_under_pol <- data.frame(balanceo_svm_under_pol,sensibilidad_svm_under_pol,
especificidad_svm_under_pol ,accuracy_svm_under_pol ,accuracy_bal_svm_under_pol )

comparacion_svm_under_pol


#******Maquinas de soporte kernel POLINOMIAL balanceo oversampling**

#*Aplicación del modelo SVM datos de entrenamiento


set.seed(123)
mod_svm_over_pol <- svm(Fraude ~ ., data = data_train_over_prueba,
kernel = "polynomial", cost = 1, scale = TRUE)

summary(mod_svm_over_pol)

set.seed(123)
pred_svmpol_over_train <- predict(mod_svm_over_pol, data_train_over_prueba)

matconf_smvpol_over_train <- table(predicho  = pred_svmpol_over_train,
real = data_train_over_prueba$Fraude)
matconf_smvpol_over_train

#*****Aplicación del modelo SVM datos de testing***********


set.seed(123)
pred_svmpol_over_test <- predict(mod_svm_over_pol, data_testing_over_prueba)


table(pred_svmpol_over_test,data_testing_over_prueba$Fraude,dnn=c("Predicho","Actual"))

matconf_smvpol_over_test<- caret::confusionMatrix(pred_svmpol_over_test,
```

```
                                                    data_testing_over_prueba$Fraude,
                                                    positive="1")

balanceo_svm_over_pol      <- c("Maquinas de Soporte")
sensibilidad_svm_over_pol  <- c(matconf_smvpol_over_test$byClass["Sensitivity"])
especificidad_svm_over_pol  <- c(matconf_smvpol_over_test$byClass["Specificity"])
accuracy_svm_over_pol    <- c(matconf_smvpol_over_test$overall["Accuracy"])
accuracy_bal_svm_over_pol   <- c(matconf_smvpol_over_test$byClass["Balanced Accuracy"])
comparacion_svm_over_pol <- data.frame(balanceo_svm_over_pol,sensibilidad_svm_over_pol,
especificidad_svm_over_pol ,accuracy_svm_over_pol ,accuracy_bal_svm_over_pol )

comparacion_svm_over_pol


#******Maquinas de soporte kernel RADIAL sin balanceo **


set.seed(123)
mod_svm_tv_rad <- svm(Fraude ~ ., data = data_train_tv,
kernel = "radial", cost = 10, scale = TRUE)

summary(mod_svm_tv_rad)

set.seed(123)
pred_svmrad_tv_train <- predict(mod_svm_tv_rad, data_train_tv)

matconf_smvrad_tv_train <- table(predicho  = pred_svmrad_tv_train,
real = data_train_tv$Fraude)
matconf_smvrad_tv_train

#*Aplicación del modelo SVM datos de entrenamiento
#*****Aplicación del modelo SVM datos de testing**********
#*
#*set.seed(123)
pred_svmrad_tv_test <- predict(mod_svm_tv_rad, data_testing_tv)


table(pred_svmrad_tv_test,data_testing_tv$Fraude,dnn=c("Predicho","Actual"))


matconf_smvrad_tv_test<- caret::confusionMatrix(pred_svmrad_tv_test,
                                                data_testing_tv$Fraude,
                                                positive="1")

balanceo_svm_tv_rad      <- c("Maquinas de Soporte")
sensibilidad_svm_tv_rad  <- c(matconf_smvrad_tv_test$byClass["Sensitivity"])
especificidad_svm_tv_rad  <- c(matconf_smvrad_tv_test$byClass["Specificity"])
accuracy_svm_tv_rad    <- c(matconf_smvrad_tv_test$overall["Accuracy"])
```

```
accuracy_bal_svm_tv_rad   <- c(matconf_smvrad_tv_test$byClass["Balanced Accuracy"])
comparacion_svm_tv_rad <- data.frame(balanceo_svm_tv_rad,
sensibilidad_svm_tv_rad ,especificidad_svm_tv_rad ,accuracy_svm_tv_rad ,accuracy_bal_svm_tv_rad )

comparacion_svm_tv_rad


#*****Maquinas de soporte kernel RADIAL balanceo Undersampling********

set.seed(123)
pred_svmrad_under_test <- predict(mod_svm_under_rad, data_testing_under)


table(pred_svmrad_under_test,data_testing_under$Fraude,dnn=c("Predicho","Actual"))


matconf_smvrad_under_test<- caret::confusionMatrix(pred_svmrad_under_test,
                                                   data_testing_under$Fraude,
                                                   positive="1")

balanceo_svm_under_rad      <- c("Maquinas de Soporte")
sensibilidad_svm_under_rad  <- c(matconf_smvrad_under_test$byClass["Sensitivity"])
especificidad_svm_under_rad  <- c(matconf_smvrad_under_test$byClass["Specificity"])
accuracy_svm_under_rad    <- c(matconf_smvrad_under_test$overall["Accuracy"])
accuracy_bal_svm_under_rad   <- c(matconf_smvrad_under_test$byClass["Balanced Accuracy"])
comparacion_svm_under_rad <- data.frame(balanceo_svm_under_rad,sensibilidad_svm_under_rad,
especificidad_svm_under_rad ,accuracy_svm_under_rad ,accuracy_bal_svm_under_rad )

comparacion_svm_under_rad

#******Aplicación del modelo SVM datos de entrenamiento************


set.seed(123)
pred_svmrad_under_test <- predict(mod_svm_under_rad, data_testing_under)
  table(pred_svmrad_under_test,data_testing_under$Fraude,dnn=c("Predicho","Actual"))


matconf_smvrad_under_test<- caret::confusionMatrix(pred_svmrad_under_test,
                                                   data_testing_under$Fraude,
                                                   positive="1")

balanceo_svm_under_rad      <- c("Maquinas de Soporte")
sensibilidad_svm_under_rad  <- c(matconf_smvrad_under_test$byClass["Sensitivity"])
especificidad_svm_under_rad  <- c(matconf_smvrad_under_test$byClass["Specificity"])
accuracy_svm_under_rad    <- c(matconf_smvrad_under_test$overall["Accuracy"])
accuracy_bal_svm_under_rad   <- c(matconf_smvrad_under_test$byClass["Balanced Accuracy"])
comparacion_svm_under_rad <- data.frame(balanceo_svm_under_rad,sensibilidad_svm_under_rad,
```

```
especificidad_svm_under_rad ,accuracy_svm_under_rad ,accuracy_bal_svm_under_rad )

comparacion_svm_under_rad

# *****Maquinas de soporte kernel RADIAL balanceo Oversampling*******

set.seed(123)
mod_svm_over_rad <- svm(Fraude ~ ., data = data_train_over_prueba,
kernel = "radial", cost = 1, scale = TRUE)

summary(mod_svm_over_rad)

set.seed(123)
pred_svmrad_over_train <- predict(mod_svm_over_rad, data_train_over_prueba)

matconf_smvrad_over_train <- table(predicho  = pred_svmpol_over_train,
real = data_train_over_prueba$Fraude)
matconf_smvrad_over_train

#******Aplicación del modelo SVM datos de entrenamiento************

set.seed(123)
pred_svmrad_over_test <- predict(mod_svm_over_rad, data_testing_over_prueba)
  table(pred_svmrad_over_test,data_testing_over_prueba$Fraude,dnn=c("Predicho","Actual"))

matconf_smvrad_over_test<- caret::confusionMatrix(pred_svmrad_over_test,
                                                  data_testing_over_prueba$Fraude,
                                                  positive="1")

balanceo_svm_over_rad      <- c("Maquinas de Soporte")
sensibilidad_svm_over_rad  <- c(matconf_smvrad_over_test$byClass["Sensitivity"])
especificidad_svm_over_rad  <- c(matconf_smvrad_over_test$byClass["Specificity"])
accuracy_svm_over_rad   <- c(matconf_smvrad_over_test$overall["Accuracy"])
accuracy_bal_svm_over_rad   <- c(matconf_smvrad_over_test$byClass["Balanced Accuracy"])
comparacion_svm_over_rad <- data.frame(balanceo_svm_over_rad,sensibilidad_svm_over_rad,
especificidad_svm_over_rad ,accuracy_svm_over_rad ,accuracy_bal_svm_over_rad )
comparacion_svm_over_rad
```

Referencias

Alenzi., H. Z., y Aljehane, N. O. (2020). Fraud detection in credit cards using logistic regression. *University, Tabuk*, *11*, 1-12.

Alenzi, H. Z., y Aljehane, N. O. (2020a, 1). Fraud Detection in Credit Cards using Logistic Regression. *International Journal of Advanced Computer Science and Applications*, *11*(12). Descargado de `https://doi.org/10.14569/ijacsa.2020.0111265` doi: 10.14569/ijacsa.2020.0111265

Alenzi, H. Z., y Aljehane, N. O. (2020b, 1). Fraud Detection in Credit Cards using Logistic Regression. *International Journal of Advanced Computer Science and Applications*, *11*(12). Descargado de `https://doi.org/10.14569/ijacsa.2020.0111265` doi: 10.14569/ijacsa.2020.0111265

Brownlee, J. (2020). Tour of data sampling methods for imbalanced classification [Manual de software informático]. Descargado de `https://tinyurl.com/bdcneayt` (Machine Learning Mastery)

CRC, P. (Ed.). (s.f.). *Uso de r y rstudio para la gestión de datos, análisis estadístico y gráficos.*

Hosmer, J. D. W., Lemeshow, S., y Sturdivant, R. X. (2013). *Regresión logística aplicada.* John Wiley and Sons.

Jain, Y., NamrataTiwari, ShripriyaDubey, y Jain, S. (2019). A comparative analysis of various credit card fraud detection techniques. *International Journal of Recent Technology and Engineering*, *7*, 402-403.

James, G., Witten, D., Hastie, T., y Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112). Springer.

Jaramillo Chaparro, J. A. (2015). Tendencias recientes en el pronóstico de series de tempo financieras usando máquinas de vectores de soporte. *Universidad Nacional de Colombia*, 1-79.

Kraiem, M. S., y cols. (2020). Resumen de tesis. clasificación a partir de conjuntos de datos no equilibrados. un marco para mejorar la aplicación de las estrategias de remuestreo.

Lamblet, A. (2019). Cómo lidiar con datos desequilibrados en problemas de clasificación [Manual de software informático]. Descargado de `https://tinyurl.com/4bwhbsus`

Martínez, J. (2019). *La paradoja del accuracy.* Descargado 2019, de `https://datasmarts.net/es/que-es-el-accuracy/`

Medina, F. (2001). *Consideraciones sobre el índice de gini para medir la concentración del ingreso.* Cepal.

Norman, A. T. (2019). *Aprendizaje automático en acción.* Litres.

Osorio, J. K. H. (2019). *Metodología de clasificación de datos desbalanceados basado en métodos de submuestreo* (Tesis de Magister). Universidad Tecnològica de Pereira, Colombia.

Payments, M. (2022). *Tendencia en medios de pago.* Descargado 2022-03-20, de `https://www.minsaitpayments.com/recursos/informe-2022`

Rebollo Neira, L., Plastino, Á., y Zyserman, F. (1994). Un método de máxima entropía para el análisis de una mezcla de tierras raras.

Springer (Ed.). (1996). *Bagging predictors.*

Torres-Vásquez, M., Hernández-Torruco, J., Hernández-Ocaña, B., y Chávez-Bosquez, O. (2021). Impacto de los algoritmos de sobremuestreo en la clasificación de subtipos principales del síndrome de guillain-barré. *Ingenius. Revista de Ciencia y Tecnología*(25), 20–31.

TransUnion. (2022). *Informe global sobre tendencias de fraude digital.* Descargado 2022-04-2022, de `https://tinyurl.com/bddj83zt`

Ulises Jiménez Cardoso, J. M. S. C. (2020). Detección de fraude interno utilizando metodologías de aprendizaje máquina. *Universidad Panamericana,Facultad de Ingeniería,México*, 1-12.

Yadav, A. (2021). *How to perform stratified sampling on dataset in r.* Descargado 2021-02-26, de `https://tinyurl.com/msrxem6v`