UNIVERSIDAD DE CONCEPCIÓN
Departamento de Ingeniería Informática
y Ciencias de la Computación

Profesor Guía
Guillermo Cabrera Vives

Puntuación de clientes de una entidad financiera mediante técnicas de minería de datos sobre un conjunto de datos incompleto

Aldo Andrés Concha Ortega

Tesis presentada para la obtención del grado de

MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN

Julio, 2019

RESUMEN

Una de las tareas más demandantes de los departamentos de ventas en el sector financiero, radica en el ofrecimiento de productos y servicios a los clientes a través de canales directos. El presente trabajo se centra en la creación de un ranking para guiar las labores de marketing directo de una entidad financiera de manera efectiva a partir de datos que poseen el principal problema de estar incompletos. Para conseguir este objetivo se plantea entrenar modelos de clasificación para identificar a los clientes más propensos a contratar los productos ofrecidos y modelos de regresión para estimar cuánto dinero se invertiría.

Desde el punto de la incompletitud de los datos, se comparan tres algoritmos de imputación: GAIN (Yoon, Jordon, & van der Schaar, 2018), MICE (Azur, Stuart, Frangakis, & Leaf, 2011) y missForest (Stekhoven & Bühlmann, 2011). A partir de la porción de los datos en los que todos sus atributos eran conocidos, se realizaron simulaciones sobre conjuntos con distintos niveles de incompletitud y, en todas ellas, missForest superó significativamente a los demás algoritmos de imputación en términos de RMSE y PFC. Sin embargo, dado el excesivo costo computacional de missForest y sumando el hecho de que no se pueden aplicar modelos ya entrenados de este algoritmo a datos nuevos, GAIN se muestra como mejor opción para las tareas de imputación de nuevos datos.

Desde los elementos que componen el ranking –modelos de clasificación y regresión– se implementan y comparan modelos de Random Forest (James, Witten, Hastie, & Tibshirani, 2013) entrenados desde los datos imputados, además de modelos de XGBoost (Chen & Guestrin, 2016) entrenados desde datos imputados e incompletos. XGBoost es el algoritmo que exhibe mejores resultados en términos de exactitud balanceada para las tareas de clasificación, superando a los demás algoritmos significativamente en dos de los tres productos analizados.

Por contraparte, en las tareas de regresión no se logran resultados satisfactorios, alcanzando errores absolutos medios de entre 86 y 420 veces el valor de las cifras que se deseaban predecir. Entonces, se decide utilizar solo las predicciones de los modelos de clasificación de clientes para guiar el marketing directo de la entidad financiera.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

NOMENCLATURA Y ABREVIACIONES

ACC	exactitud, del inglés accuracy
AFP	administradora de fondos de pensiones
ANN	red neuronal artificial, del inglés *artificial neural network*
BAL_ACC	exactitud balanceada, del inglés balanced accuracy
CAV	cuenta de ahorro voluntario
CCV	cuenta de cotización voluntaria
CCO	cuenta de cotización obligatoria
CDC	cuenta de depósitos convenidos
CLP	peso chileno
FN	falsos negativos, del inglés *false negatives*
FP	falsos positivos, del inglés *false positives*
GAIN	del inglés *generative adversarial imputation networks*
MAPE	error porcentual absoluto medio, del inglés *mean absolute percentage error*
MICE	del inglés *multivariate imputation by chained equations*
MSE	error cuadrático medio, del inglés *mean square error*
NRMSE	raíz del error cuadrático medio normalizada, del inglés *normalized root mean square error*
PFC	proporción de clasificados falsamente, del inglés *proportion of falsely classified*
PPV	precisión, del inglés *positive predictive value*
PR	proporción de elementos positivos, del inglés positive rate
RMSE	raíz del error cuadrático medio, del inglés *root mean square error*
RSS	suma de cuadrados de los residuos, del inglés *residual sum of squares*
TN	verdaderos negativos, del inglés *true negatives*
TP	verdaderos positivos, del inglés *true positives*
TPR	exhaustividad, del inglés *true positive rate*

CAPÍTULO 1. INTRODUCCIÓN

1.1. Antecedentes Generales

Las administradoras de fondos de pensiones (AFP) de Chile son instituciones financieras privadas encargadas de administrar los fondos de cuentas individuales de ahorros para pensiones. Una de las tareas más demandantes del área de ventas dentro de dichas entidades, radica en el ofrecimiento de un set de productos adicionales a la cotización previsional legal, a través de marketing directo, a aquellos clientes ya consolidados. Estos productos consisten en cuentas que permiten a los clientes un ahorro previsional adicional y voluntario con distintas características. Este trabajo se centra en apoyar el marketing directo de una entidad financiera que cuenta con tres productos de contratación voluntaria y la decisión de qué cliente contactar y qué producto ofrecer es tomada basada fundamentalmente en el criterio de cada ejecutivo.

Se propone crear un ranking de usuarios a través de la aplicación de técnicas de minería de datos sobre los perfiles de usuarios de la entidad financiera, con el objetivo de dirigir los esfuerzos del área de ventas de manera personalizada y más efectiva al momento de hacer la selección del cliente y producto a ofrecer. La puntuación de cada usuario asociada a cada producto constará de dos partes principales: primero, estimar la probabilidad de que el cliente contrate cada uno de los tres productos estudiados (asociado a un problema de clasificación binaria para cada uno de ellos) y segundo, estimar el monto de dinero a invertir en dicho producto (asociado a un problema de regresión). Ambas partes serán optimizadas por separado y unificadas mediante su multiplicación, obteniendo una métrica que represente la esperanza del monto a invertir.

El gran desafío que subyace a la resolución del problema consiste en el alto número de datos faltantes para distintos clientes, dificultando e incluso imposibilitando la aplicación de los enfoques tradicionales de minería de datos. Se plantea la resolución de este desafío mediante la revisión, implementación y comparación de diferentes enfoques y algoritmos del estado del arte orientados a lidiar con conjuntos de datos incompletos, entre los que destacan técnicas de imputación y XGBoost, en adición de un análisis del comportamiento de los mismos en función de distintitos niveles de completitud de los datos.

Otras características que suponen desafíos para el alcance de los objetivos planteados son la presencia de atributos categóricos con un dominio demasiado grande y el desbalance de las clases, es decir, se tiene más información de los clientes que no han contratado los productos de los que sí los han contratado, siendo estos últimos el foco de atención.

1.2. Hipótesis

La creación de un ranking de clientes a partir de la totalidad de perfiles disponibles y enfoques que manejen datos incompletos mejorará la calidad de la predicción en términos de exactitud balanceada[1] y RMSE[2] para las componentes de clasificación y regresión, respectivamente, en comparación con basar el ranking en métodos tradicionales aplicado solo a los perfiles completos.

1.3. Objetivo General

Crear un modelo para el ranking de clientes asociado a cada producto adicional ofrecido, mediante la revisión, implementación y comparación de técnicas de minería de datos que permitan la estimación tanto de la probabilidad de contratación como del monto a invertir obtenidas desde el análisis de perfiles de clientes incompletos.

1.4. Objetivos Específicos

1. Estudiar el comportamiento de los métodos de imputación en función de distintos niveles de completitud de los datos.
2. Comparar distintos modelos predictivos que puedan ser entrenados y utilizados sobre datos incompletos, aplicados al total de los perfiles de clientes disponibles y evaluados en función de la exactitud balanceada y RMSE para los problemas de clasificación y regresión, respectivamente.
3. Construir un ranking de clientes a partir del mejor modelo de clasificación y del mejor modelo de regresión.

[1] Del inglés *balanced accuracy.*
[2] Acrónimo del inglés *root mean square error* (raíz del error cuadrático medio).

CAPÍTULO 2. MARCO TEÓRICO

2.1. Minería de datos en marketing directo y entidades financieras

Gracias al desarrollo y accesibilidad de las tecnologías para la recopilación de información, casi todos los campos están siendo inundados por grandes volúmenes de datos: desde sensores en marte (Hong, Gerla, Wang, & Clare, 2002) e imágenes de múltiples telescopios (Akiyama et al., 2019) hasta datos desde las profundidades de los océanos (Akyildiz, Pompili, & Melodia, 2005), pasando con especial énfasis por el sector industrial (Yin, Ding, Xie, & Luo, 2014). En consecuencia, el siguiente paso consiste en capturar la información implícita de dichos datos y traducirla en valor para las empresas y clientes. La minería de datos, es entonces, un nexo entre datos e información con valor y sus técnicas han sido ampliamente estudiadas y utilizadas en enfoques como Gestión de Relaciones con Clientes (CRM)[3] (Chavan, Jadhav, Suryagandh, & Sharma, 2018; Linoff & Berry, 2011) y específicamente en el marketing directo (Mitik, Korkmaz, Karagoz, Toroslu, & Yucel, 2017; Crone, Lessmann, & Stahlbock, 2006).

Refiriéndose específicamente al rubro financiero, diversas técnicas de clasificación, regresión y *clustering* han sido utilizadas en tareas tales como:

- Análisis de series de tiempo para predecir comportamientos bursátiles e indicadores financieros (Cao & Tay, 2003; Yoo, Kim, & Jan, 2005; Lu, Lee, & Chiu, 2009; Das & Padhy, 2018).
- Segmentación y diferenciación de estrategias de marketing entre los usuarios (Niyagas, Srivihok, & Kitisin, 2006; Sánchez, Clempner, & Poznyak, 2015).
- Predicción del comportamiento de compras de clientes (Etaiwi, Biltawi, & Naymat, 2017).
- Calcular y predecir *scorecards, credit score* y *behavioral score*. Estos índices representan el valor y el riesgo crediticio de los clientes y ayudan a las instituciones a

[3] Gestión de Relaciones con Clientes, en adelante CRM por sus siglas del inglés *Customer Relationship Management*.

decidir si otorgar o no préstamos (Thomas, Crook, & Edelman, 2017; Cardoso et al., 2016).

2.2. Conceptos preliminares

En esta sección se describen los algoritmos bases de los cuales se sustentan y desprenden algunos de los enfoques para lidiar con datos faltantes discutidos en la sección 2.3.

2.2.1. Árboles de Decisión

Los árboles de decisión (Breiman, Friedman, Olshen, & Stone, 1984) son algoritmos de clasificación y regresión que consisten en dividir el espacio de los atributos en J regiones disjuntas, $R_1, R_2, \dots, R_J$, luego, para cada elemento nuevo que caiga en la región R_j se le asigna una misma predicción, la cual corresponde generalmente a la media o moda de la variable de respuesta de los elementos del conjunto de entrenamiento que pertenecen a la región R_j (James, Witten, Hastie, & Tibshirani, 2013).

El nombre del algoritmo se debe a que las decisiones empleadas para realizar las divisiones pueden ser representadas en un árbol binario y el objetivo del problema es encontrar la estructura del árbol que optimice una función objetivo que mida la calidad de sus predicciones. Dado que enumerar todas las estructuras posibles de árboles resulta inviable computacionalmente (James, Witten, Hastie, & Tibshirani, 2013), se utiliza un enfoque *greedy*[4]: se parte por un nodo que inicialmente contiene todo el conjunto de datos y se añaden iterativamente ramas (divisiones en los nodos) al árbol. Para decidir qué nodo dividir y en dónde hacerlo, se analiza exhaustivamente todas las posibles divisiones en términos de la reducción de la función objetivo, es decir, la diferencia del puntaje del árbol antes y después de la división. La suma de cuadrados de los residuos (RSS[5]) es una de las posibles funciones objetivos a minimizar para los árboles de regresión, y se define por:

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

[4] También conocido como enfoque avaro o voraz, consiste en elegir en cada iteración la opción óptima local, sin considerar necesariamente el óptimo global.
[5] RSS, del inglés *residual sum of square*.

donde $\hat{y}_{R_j}$ es la respuesta media de las observaciones del conjunto de entrenamiento que pertenecen a la *j*-ésima región. Mientras que para clasificación, el índice de Gini es una de las funciones objetivo utilizada, y se define por:

$$G = \sum_{j=1}^{J} \sum_{k=1}^{K} \hat{p}_{jk}(1 - \hat{p}_{jk}),$$

donde *K* denota el número de clases y $\hat{p}_{jk}$ representa la proporción de las observaciones de entrenamiento de la *j*-ésima región, que pertenecen a la *k*-ésima clase.

Los árboles de decisión son simples y fáciles de interpretar, sin embargo, normalmente no son competitivos en términos de precisión en las predicciones, no obstante, algoritmos que combinan múltiples árboles de decisión han logrado revertir esta situación. Los enfoques más utilizados para este fin son *random forest* y el *boosting* de árboles de decisión (un ejemplo de este último es discutido en el apartado III).

2.2.2. Random Forest

El algoritmo *random forest* consiste en la creación de múltiples árboles de decisión, en donde la predicción final es calculada a través del promedio de todos los modelos. La clave del algoritmo radica en que cada vez que se considera una división en un árbol, se elige un subconjunto aleatorio de los atributos totales como candidatos de división, en vez de la totalidad de atributos (James, Witten, Hastie, & Tibshirani, 2013). Este paso ayuda a generar árboles distintos entre sí logrando un estimador final robusto, además, el algoritmo es conocido por funcionar bien bajo condiciones complicadas tales como alta dimensionalidad, interacciones complejas y estructuras de datos no lineales (Stekhoven & Bühlmann, 2011).

2.2.3. Redes Neuronales Artificiales

Las redes neuronales artificiales (ANN)[6] están inspiradas por el comportamiento de las neuronas biológicas y su objetivo consiste en estimar una salida u *output* $\hat{y}$ a partir de un vector de entrada o *input* x. Están compuestas por un conjunto de neuronas artificiales organizadas en capas (ver Figura 1 (b)), en donde las capas ubicadas entre la entrada y salida

[6] ANN del inglés *artificial neural netwok*.

se denominan capas ocultas o *hidden layers* (Haykin, Haykin, Haykin, Elektroingenieur, & Haykin, 2009).

Una neurona k, con un vector de entrada x de m componentes está compuesta por (ver Figura 1 (a)):

- Conjunto de sinapsis o enlaces el cual es obtenido mediante la multiplicación de la señal –proveniente de la capa predecesora– y el peso w_{kj} asociado al enlace.
- Una función que suma todas las señales de entrada:

$$v_k = \sum_{j=0}^{m} w_{kj} \cdot x_j.$$

  Esta operación constituye una combinación lineal de los elementos de las entradas.
- Una función de activación denotada por $\varphi(\cdot)$ que restringe el rango de la salida y_k de una neurona
- El sesgo denotado por b_k correspondiente a la función lineal v_k.

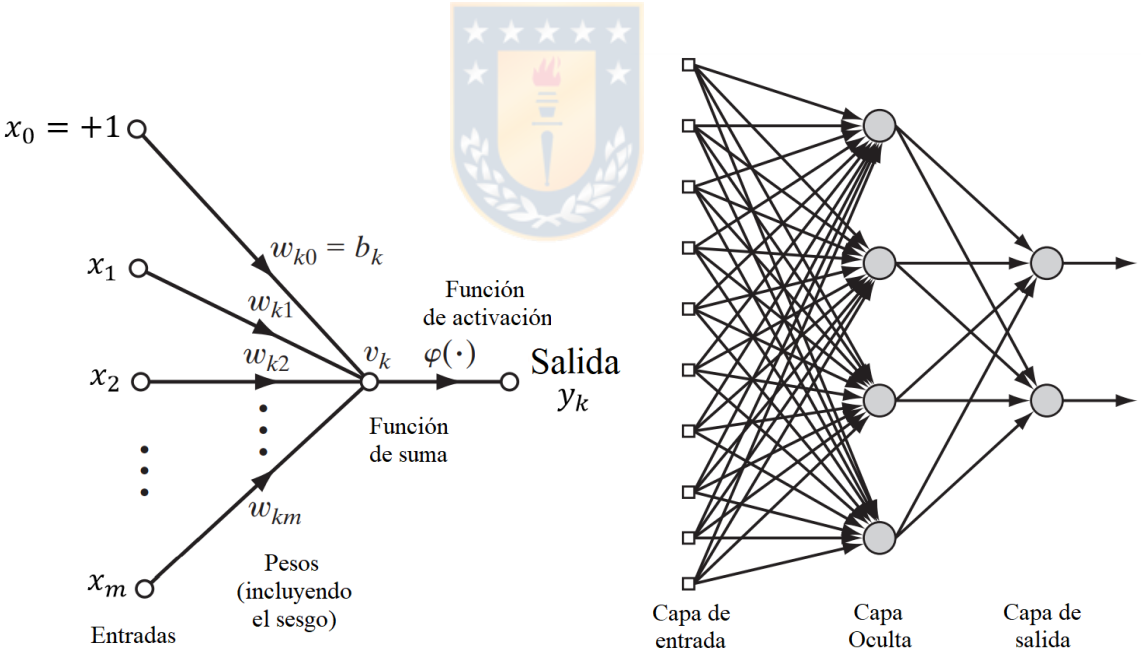

Figura 1. Representación de una red neuronal artificial. (a) Grafo del flujo de señales de una neurona. (b) Representación de una ANN totalmente conectada.
Fuente: Haykin et al. (2009)

Para calcular el valor de los pesos que optimicen las predicciones, en la etapa de entrenamiento se utiliza una función de pérdida que evalúa el error entre el valor $\hat{y}$ estimado por la ANN y el valor real y correspondiente. Este error, es propagado sucesivamente desde la capa de salida hasta la capa de entrada actualizando los pesos en un proceso llamado *backpropagation*.

2.3. Datos incompletos

Ya sea por motivos económicos, problemas en sensores o muchas otras razones, enfrentarse a conjuntos de la vida real incompletos resulta una situación normal, sin embargo, la mayoría de los procedimientos de análisis de datos no fueron diseñados para lidiar con este hecho. Basándose en el marco de referencia propuesto por García-Laencina et al. (2010) se realiza la revisión de algunos de los principales enfoques y métodos que permiten el análisis de conjuntos incompletos.

2.3.1. Mecanismos de incompletitud

Al enfrentarse ante datos faltantes, una de las principales cuestiones a aclarar es el patrón de incompletitud en los datos ya que dependiendo de esto distintos caminos podrían ser los más adecuados.

El mecanismo de incompletitud es caracterizado por la distribución condicional de M dado X: $p(M|X,\phi) = p(M|X_{obs}, X_{mis}, \phi)$, en donde $X = (x_{ij})$ es la matriz de datos. Los elementos observados de X son representados por la matriz X_{obs} , mientras que los elementos faltantes de X son representados por la matriz X_{mis}, también se define la matriz binaria indicadora de incompletitud $M = (m_{ij})$, en donde $m_{ij} = 1$ si x_{ij} es un dato faltante y $m_{ij} = 0$ si x_{ij} es un dato observado. Además, ϕ denota parámetros desconocidos que definen el mecanismo de incompletitud.

Bajo este contexto, Little y Rubin (2002) definen tres mecanismos de incompletitud de los datos:

- Completamente al azar (MCAR)[7], si el evento de que un dato falte es completamente aleatorio, es decir, que no depende de los datos observados ni de los datos faltantes. La condición MCAR puede ser expresada por la relación:

$$p(M|X_{obs}, X_{mis}, \phi) = p(M|\phi).$$

- Al azar (MAR)[8], si el evento de que un dato falte depende solo de los datos observados. La condición MAR puede ser expresada por la relación:

$$p(M|X_{obs}, X_{mis}, \phi) = p(M|X_{obs}, \phi).$$

- No al azar (MNAR)[9], si el evento de que un dato falte depende de los datos faltantes en sí. En este caso la distribución condicional $p(M|X_{obs}, X_{mis}, \phi)$ no puede ser simplificada.

Desde la perspectiva de las implicancias de los mecanismos de incompletitud de los datos, en el caso de MCAR, los parámetros estimados desde los datos observados no son sesgados por los elementos faltantes, entonces, se puede ignorar las razones de la incompletitud en el análisis de los datos y simplificar los métodos utilizados para su tratamiento. En el caso de MAR, los estimadores de los parámetros pueden ser segados, sin embargo, existen maneras de lidiar con esta condición y producir estimadores relativamente no sesgados (Lu & Copas, 2004). El caso de MNAR es más complicado, la única manera de obtener una estimación no sesgada de los parámetros es modelando la incompletitud en sí (Yu et al., 2012).

2.3.2. Técnicas para lidiar con datos faltantes

2.3.2.1. Eliminar datos incompletos

Comenzando con los enfoques para lidiar con la incompletitud de la información, la solución más sencilla es conservar solo el subconjunto de datos completos, sin embargo, esto deja de ser aplicable cuando el porcentaje de datos incompletos en el conjunto de entrenamiento es considerable o cuando se necesitan hacer predicciones sobre datos incompletos.

2.3.2.2. Imputación

Otra técnica comúnmente empleada sobre conjuntos incompletos es la *imputación*, la cual consiste en estimar los datos faltantes a partir de los datos observados, así obtener un

[7] MCAR, del inglés *Missing Completely at Random.*
[8] MAR, del inglés *Missing at Random.*
[9] MNAR, del inglés *Missing not at Random.*

conjunto completo desde donde es posible utilizar los métodos de minería de datos tradicionales. A continuación se presentan algunos de los principales métodos de imputación.

I. Media/Moda

Este método es el más sencillo y consiste en reemplazar los datos faltantes por su media o moda, dependiendo si corresponden a variables continuas o categóricas, respectivamente, obtenidas desde los datos conocidos de dicha variable.

II. Modelos de Machine Learning

Otro método de imputación consiste en entrenar modelos de regresión o clasificación para estimar los valores faltantes. Si bien este enfoque es ampliable a cualquier modelo, los siguientes cuatro apartados describen algunos de los algoritmos utilizados en este enfoque de imputación.

Vecinos más Cercanos

El algoritmo de los *k* vecinos más cercanos (kNN)[10] para imputación (Chen & Shao, 2000) consiste en estimar los valores faltantes de un vector específico promediando los atributos de los *k* vectores más cercanos del conjunto de entrenamiento ($k \in \mathbb{N}$). Un aspecto clave para la aplicación de este procedimiento es la función de distancia o similitud entre vectores, la cual debe aceptar elementos incompletos. Wilson y Martinez (1997) proponen la *heterogeneous euclidean-overlap metric* (HEOM), en la cual la distancia entre dos vectores $\boldsymbol{x_a}, \boldsymbol{x_b}$ se define como:

$$HEOM(\boldsymbol{x_a}, \boldsymbol{x_b}) = \sqrt{\sum_{i=1}^{m} d_i(x_{ai}, X_{bi})^2},$$

donde $d_i(x_{ai}, x_{bi})$ es la distancia entre $\boldsymbol{x_a}$ y $\boldsymbol{x_b}$ en su i-ésimo atributo:

$$d_i(x_{ai}, x_{bi}) = \begin{cases} 1, & \text{si } x_{ai} \vee x_{bi} \text{ son datos faltantes} \\ overlap(x_{ai}, x_{bi}), & \text{si } i \text{ es una variable categórica} \\ rn_{diff(x_i, y_i)}, & \text{si } i \text{ es una variable continua} \end{cases}$$

A su vez, las funciones *overlap* y *range normalized difference* (*rn_diff*) se definen como:

[10] kNN, del inglés *k-nearest neighbors.*

$$overlap(x_{ai}, x_{bi}) = \begin{cases} 0, & \text{si } x_{ai} = x_{ai} \\ 1, & \text{en otro caso'} \end{cases}$$

$$rn\_diff(x_{bi}, x_{bi}) = \frac{|x_{ia} - x_{ib}|}{\max(x_i) - \min(x_i)}.$$

Multiple Imputation by Chained Equations

El proceso general de *Multiple Imputation by Chained Equations* (MICE) puede ser descompuesto en los siguientes pasos generales (Azur, Stuart, Frangakis, & Leaf, 2011):

Paso 1: la inicialización es llevada a cabo mediante una imputación simple de los datos no observados, por ejemplo, utilizando su media o moda por variables.

Paso 2: los elementos imputados de una variable o columna específica ("*var*") son devueltos al estado de no observados o faltantes.

Paso 3: dependiendo si la variable "*var*" escogida en el Paso 2 es continua o categórica, un modelo de regresión o clasificación, respectivamente, es entrenado con el fin de predecir los elementos faltantes de dicha variable. El entrenamiento del modelo es conseguido a partir de todas las filas en donde "*var*" es observada, utilizando a esta como variable de respuesta y todas las demás columnas –incluyendo sus elementos imputados– son utilizadas como variables independientes.

Paso 4: los valores no observados de "*var*" son imputados mediante las predicciones obtenidas por el modelo entrenado en el Paso 3.

Los Pasos 2-4 son repetidos para cada una de las variables que contienen elementos faltantes completando así una iteración o ciclo. El proceso completo de MICE está conformado por múltiples iteraciones, en cada una de las cuales todos los valores imputados son actualizados. Un criterio de detención consiste en que al final de las iteraciones los parámetros que controlan las imputaciones (por ejemplo, los coeficientes en un modelo de regresión) deben converger hasta un valor estable.

A diferencia de una imputación simple –en donde se utiliza un solo valor para completar cada elemento faltante- la imputación múltiple, que es utilizada en este algoritmo, reemplaza cada elemento faltante por un conjunto de valores verosímiles los cuales representan la incertidumbre con respecto al valor a imputar. Estos múltiples elementos pueden ser combinados en procesos de inferencia.

MissForest

MissForest (Stekhoven & Bühlmann, 2011) sigue el mismo procedimiento general de MICE de imputar por ciclos a través de cada variable con elementos faltantes utilizando como modelo predictor –tal como su nombre lo sugiere– el algoritmo *random forest*.

Al utilizar *random forest* como base, se heredan todas sus ventajas ya mencionadas en la el apartado 2.2.2, además, es un modelo no paramétrico que no necesita realizar asunciones explícitas de la función a estimar.

Generative Adversarial Imputation Networks

Generative Adversarial Imputation Nets (GAIN) (Yoon, Jordon, & van der Schaar, 2018) es un algoritmo en donde dos ANN son entrenadas con objetivos opuestos: una red denominada generador, cuya meta es imputar con la mayor exactitud posible los datos faltantes, y otra red denominada discriminador, la cual tiene el objetivo de distinguir entre los componentes observados realmente y aquellos que fueron imputados.

Los elementos faltantes de la matriz de datos originales son reemplazados por números aleatorios y una matriz binaria es generada para almacenar las posiciones de los elementos observados y faltantes (matriz máscara). Estas componentes alimentan al generador, el cual es entrenado mediante una función de pérdida que busca por una parte minimizar el error entre los datos generados y los datos reales (MSE[11]), y por otra parte maximizar el error de clasificación del discriminador (entropía cruzada), es decir, conseguir que sea difícil discernir entre los datos reales e imputados. En contraste, el discriminador busca determinar qué elementos fueron observados y qué elementos fueron imputados. Este último es alimentado con la salida del generador y por una matriz pista que revela parcialmente la matriz máscara, y es entrenado mediante una función de pérdida que busca minimizar el error de clasificación (entropía cruzada) entre las probabilidades de salida y la matriz máscara. Así, estas dos redes son entrenadas utilizando un proceso adversario (Yoon et al., 2018). La Figura 2 esquematiza la arquitectura general de GAIN.

[11] Error cuadrático medio, del inglés *mean square error*.

Formalmente, se define la matriz de datos originales $X = \{\boldsymbol{x}_1, .. \boldsymbol{x}_n\}$, donde es n igual al número de datos y $\boldsymbol{x}_i = \{x_{i,1}, \dots, x_{i,d}\}$, con d igual al número de atributos. Se define la matriz máscara M, binaria, donde:

$$m_{i,j} = \begin{cases} 1, & \text{si } x_{i,j} \text{ es observado} \\ 0, & \text{si } x_{i,j} \text{ no es observado} \end{cases}$$

Además, se define la matriz de pista H, la cual copia y entrega información parcial de la matriz M, asignando aleatoriamente con una probabilidad p el valor de 0.5 a algunos de sus elementos, denotando incertidumbre. De esta forma:

$$h_{i,j} = \begin{cases} 1 \rightarrow m_{i,j} = 1 \\ 0 \rightarrow m_{i,j} = 0 \\ 0.5 \rightarrow (m_{i,j} = 1 \ \vee \ m_{i,j} = 0) \end{cases},$$

entonces, la función de pérdida del discriminador está definida por la minimización de:

$$\mathcal{L}_D(\boldsymbol{m}_i, \widehat{\boldsymbol{m}}_i, \boldsymbol{h}_i) = -\sum_{j:h_{i,j}=0.5} [m_{i,j} \log(\widehat{m}_{i,j}) + (1 - m_{i,j}) \log(1 - \widehat{m}_{i,j})],$$

donde $\widehat{m}_{i,j}$ es la estimación de $m_{i,j}$ realizada por el discriminador. Por su parte, la función de pérdida del generador está definida por la minimización de:

$$\mathcal{L}_G(\boldsymbol{m}_i, \widehat{\boldsymbol{m}}_i, \boldsymbol{h}_i, \boldsymbol{x}_i, \widehat{\boldsymbol{x}}_i) = -\sum_{j:h_{i,j}=0.5} (1 - m_{i,j}) \log(\widehat{m}_{i,j}) + \alpha \sum_{j:m_{i,j}=1} L_M(x_{i,j}, \hat{x}_{i,j}),$$

con L_M definido por:

$$L_M(\boldsymbol{x}_i, \widehat{\boldsymbol{x}}_i) = \begin{cases} (x_{i,j} - \hat{x}_{i,j})^2, & \text{si } x_i \text{ es continuo} \\ -x_{i,j} \log(\hat{x}_{i,j}), & \text{si } x_i \text{ es binario} \end{cases}$$

donde $\hat{x}_{i,j}$ es la estimación de $x_{i,j}$ realizada por el generador. De esta forma, la minimización del primer término de $\mathcal{L}_G$ es equivalente a la maximización de la función de pérdida del discriminador ($\mathcal{L}_D$) siguiendo un enfoque adversario, mientras que la minimización del segundo término tiene por objetivo que los elementos estimados por el generador sean parecidos a los datos observados. En este contexto, α es un hiperparámetro que pondera la importancia entre ambos términos.

Los parámetros de la ANN del discriminador (D) y de la ANN del generador (G) son entrenados y actualizados por lotes o *batches* de tamaño *k*, de acuerdo a:

$$\min_{D} \sum_{i=1}^{k} \mathcal{L}_D(\boldsymbol{m}_i, \widehat{\boldsymbol{m}}_i, \boldsymbol{h}_i),$$

$$\min_{G} \sum_{i=1}^{k} \mathcal{L}_G(\boldsymbol{m}_i, \widehat{\boldsymbol{m}}_i, \boldsymbol{h}_i, \boldsymbol{x}_i, \widehat{\boldsymbol{x}}_i),$$

para el discriminador y generador, respectivamente, aquí el índice i denota el i-ésimo elemento del lote de entrenamiento, en vez que del conjunto entero. El entrenamiento del modelo y la optimización de parámetros pueden ser llevados a cabo, por ejemplo, mediante un algoritmo de descenso estocástico.

Una imputación múltiple es fácilmente conseguible con GAIN al utilizar el generador en varias ocasiones sobre el conjunto de datos a imputar, lográndose predicciones distintas gracias al factor estocástico de los números aleatorios utilizados para completar los datos faltantes al inicio del proceso.

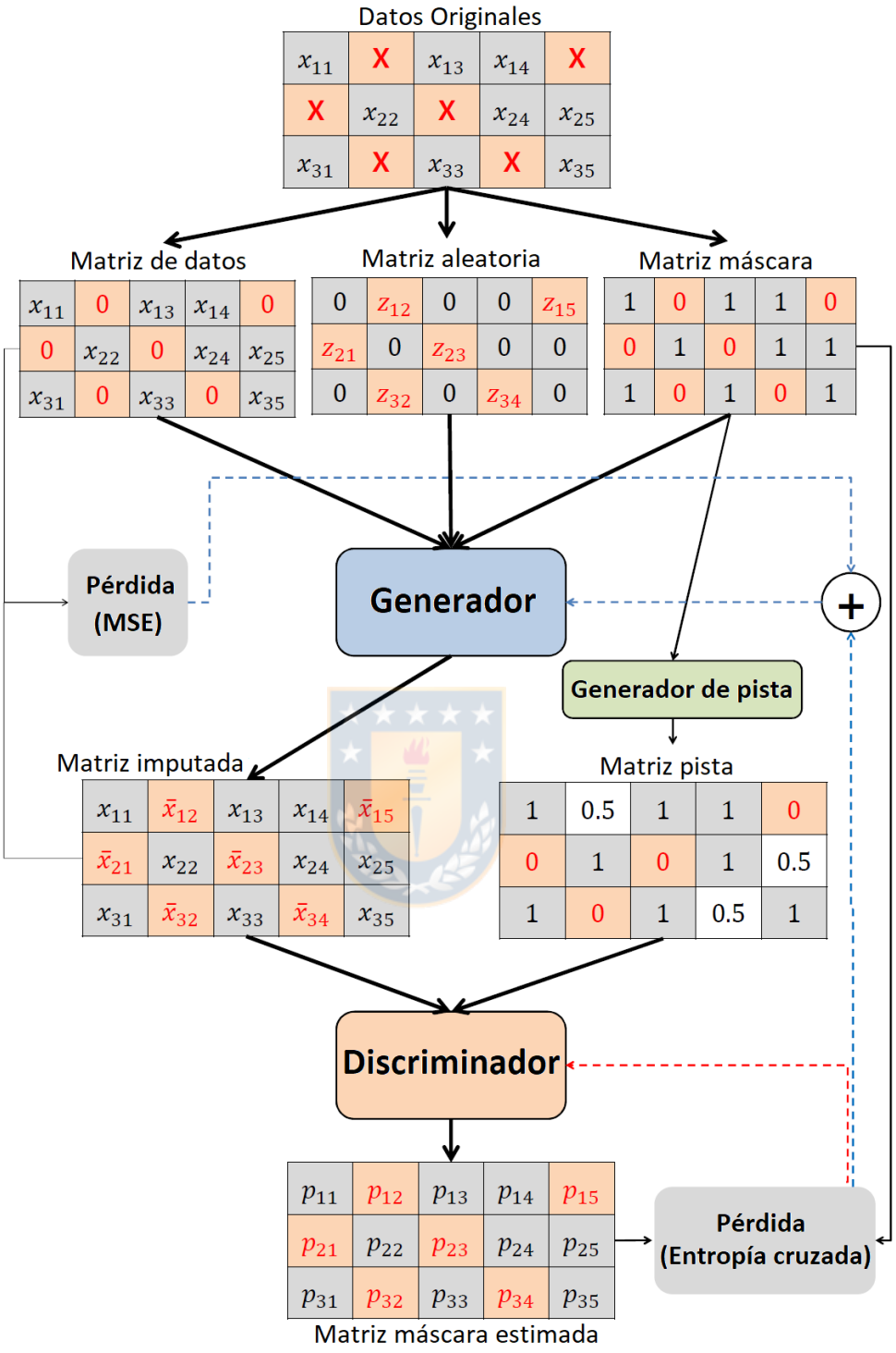

Figura 2. Arquitectura de GAIN. Con x_{ij} representando a los datos observados, X a los datos faltantes, $\bar{x}_{ij}$ a los datos imputados, en la matriz pista los números denotan la probabilidad que el elemento en cuestión sea realmente observado, del mismo modo que lo hacen los datos p_{ij} en la matriz máscara estimada.

Fuente: Yoon et al. (2018)

III. Mixture Densities

Mixture Densities es una técnica estadística para lidiar con datos incompletos. En términos generales, se basa en la suposición de que los datos son descritos por una función de densidad de probabilidad, entonces, la clave para su solución es estimar los parámetros que definen dicha función y luego, para cada vector, imputar los datos faltantes por sus esperanzas condicionadas a los datos observados (Redner & Walker, 1984).

En este enfoque es especialmente importante determinar el comportamiento de la incompletitud de los datos (MCAR, MAR o MNAR), ya que dependiendo de este ciertas simplificaciones pueden ser aplicadas.

El algoritmo *expectation-maximization* (EM) es el más utilizado para estimar los parámetros de la función de densidad desde datos incompletos alternando iterativamente entre dos pasos. En el paso *E* se calcula la esperanza de la log-versosimilitud condicionada a los parámetros en dicha iteración, luego, en el paso M se obtiene una nueva estimación de parámetros maximizando mediante gradiente (Xu & Jordan, 1996).

2.3.2.3. Métodos que trabajan con datos incompletos

En esta sección se discuten algunos enfoques y algoritmos que por sí solos pueden realizar predicciones a partir de datos incompletos.

I. Métodos basados en distancia

Como ya se ha discutido en la sección 2.2.2., la definición de una función de similitud entre vectores resulta útil para la imputación de los elementos faltantes, del mismo modo, este enfoque puede utilizarse para estimar directamente la variable objetivo sin tener que imputar previamente. En adición al algoritmo kNN antes planteado, este enfoque puede ser utilizado para cualquier algoritmo basado en distancias, por ejemplo, (Mesquita & Gomez, 2016; Yu et al., 2012) han utilizado la esperanza de la distancia cuadrática (ESD)[12] (Eirola, Doquire, Verleysen, & Lendasse, 2013) –el cual puede ser expresado en función de la esperanza y matriz de covarianza condicionales de los atributos, estimados mediante el algoritmo EM– en la construcción de *radial basis function neural network*, una red neuronal artificial en donde cada nodo de la capa oculta representa un punto en el espacio de características y su función

[12] ESD, acrónimo del inglés *Expected Squared Distance*.

de activación depende de la distanciad entre dicho punto y cada elemento del conjunto de entrenamiento.

II. C4.5

Quinlan (1994) propone el algoritmo C4.5 (y su posterior actualización C5.0) basado en árboles de decisión. Los árboles de decisión consisten en ir dividiendo consecutivamente el conjunto de entrenamiento de acuerdo a sus atributos, en donde cada división es representada por un nodo hijo asociada a cada criterio de división (nodo padre). C4.5 maneja los datos incompletos de modo que cada vez que se debe tomar una decisión sobre un atributo no observado, el elemento en cuestión es enviado a todos los nodos hijo, pero ponderado por la proporción de cada nodo al que es enviado. Por ejemplo, se puede considerar un conjunto de datos con el atributo "sexo" con 60 elementos masculinos, 40 femeninos y 20 sin etiquetar, si se dividiera en relación a esta variable, los 20 elementos sin etiquetar serían enviados al mismo tiempo al nodo masculino y al nodo femenino con una ponderación de $60/100$ y $40/100$, respectivamente.

III. XGBoost

El algoritmo empleado por XGBoost (Chen & Guestrin, 2016) se basa en el *boosting* de árboles de decisión para clasificación y regresión. Iterativamente se generan funciones aditivas, en donde cada función corresponde a un árbol distinto y depende de todas las funciones generadas anteriormente. Cada función mapea un vector de características x_i a la hoja correspondiente del árbol y retorna su peso w_j. La predicción final para un elemento es obtenida mediante la sumatoria de todas las funciones generadas

Formalmente, para un conjunto de datos $\mathcal{D} = \{(x_i, y_i)\}$ $(|\mathcal{D}| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$ el algoritmo busca minimizar la función objetivo:

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k), \qquad (1)$$

donde n es el número de elementos en el conjunto de datos, f_k son las funciones generadas, con $k = \{1, \dots, K\}$, l es una función de pérdida convexa y diferenciable que mide la diferencia entre la variable de respuesta y_i y su predicción $\hat{y}_i$ (como por ejemplo MSE), Ω es un término regularizador que se encarga de mantener el modelo sencillo y así evitar el sobre ajuste y está definido por:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2,$$

donde *T* es el número de hojas de cada árbol, w_j es el peso de la *j*-ésima hoja, γ y λ son parámetros de ponderación. El modelo presentado en la ecuación (1) no puede ser optimizado utilizando métodos convencionales, entonces, se sigue un enfoque *greedy*. En la *t*-ésima iteración se añade f_t para minimizar la siguiente función:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i{}^{(t-1)} + f_t(x_i)\right) + \Omega(f_t).$$

Para facilitar el proceso, la expansión de segundo orden de la serie de Taylor de la función es utilizada como aproximación. Luego de considerar que f_t evaluada en x_i, retorna el valor de la hoja w_j a la cual el elemento fue asignado, es decir, $f_t(x_i) = w_j$, eliminar los términos constantes y definir I_j como el conjunto de todos los elementos x_i asignados a la hoja j, donde $I_j = \{i | f(x_i) = w_j\}$, es posible derivar e igual a 0, para encontrar el peso óptimo de cada hoja:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda},$$

donde $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ y $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ son, respectivamente, los estadísticos de gradiente de primer y segundo orden de la función de pérdida. Así, es posible conseguir el valor óptimo (dependiente de cada estructura de árbol) de la función:

$$\tilde{\mathcal{L}}^{(t)} = -\frac{1}{2}\sum_{j=1}^{T} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T. \qquad (2)$$

La ecuación (2) puede ser utilizada como función objetivo para la construcción de árboles de decisión tal como se ha detallado en el apartado 2.2.10.

Para extender XGBoost a la aceptación de datos incompletos, la función objetivo es evaluada dos veces por cada división analizada: en un caso se envían todos los elementos con atributos no definidos hacía la partición izquierda y en el segundo caso se envían a la partición derecha. Entonces, cuando se defina el punto de división, los elementos con atributos faltantes se envían por defecto a la partición que haya alcanzado un mejor valor en la función objetivo.

CAPÍTULO 3. METODOLOGÍA

3.1. Descripción de los datos

Se cuenta con 846.932 perfiles de clientes y ex clientes quienes han contratado al menos una cuenta, ya sea obligatoria u opcional. Los datos provienen de dos fuentes: por un lado, se encuentran los datos demográficos y por otro, los datos relacionados con las cuentas contratadas por los clientes.

Luego de excluir los atributos que no aportan información a la investigación, tales como número de teléfono, correo electrónico, entre otros, la Tabla 1 resume los atributos demográficos utilizados.

Nombre	Tipo	Cantidad de categorías
sexo	binaria	2
edad	numérica	-
nacionalidad	categórica	50
estado civil	categórica	5
profesión	categórica	62
tipo trabajador (dependiente/independiente)	binaria	2
renta imponible	numérica	-
AFP actual	categórica	6
comuna particular	categórica	347
empleador	categórica	7.937
comuna empleador	categórica	345
actividad económica	categórica	595

Tabla 1. Descripción de variables demográficas.
Fuente: Elaboración propia.

Desde la Tabla 1 se observa que los atributos asociados a las comunas, actividad económica y empleador contienen del orden de las cientos y miles de categorías, representado un problema. Esta circunstancia y su tratamiento son abarcados en la sección 3.3.2.

Como se ha introducido ya en los capítulos anteriores, el principal desafío presente en los datos consiste en que estos no están totalmente completos. Para entender de mejor forma este hecho, la Figura 3 muestra el nivel de completitud por atributo demográfico.

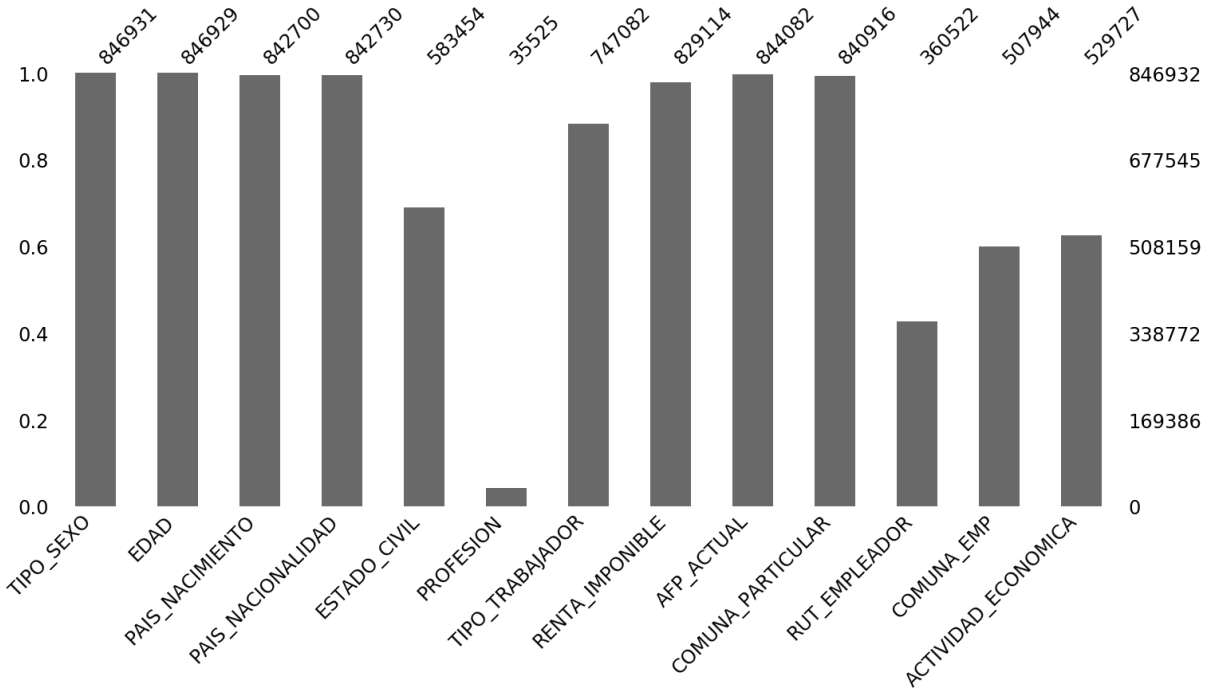

Figura 3. Nivel de incompletitud de datos por atributo. Las cifras al lado izquierdo corresponden a la fracción de datos completos en relación al total de datos, mientras que las cifras ubicadas arriba y a la derecha indican el número de elementos completos.

Fuente: Elaboración propia.

De acuerdo a conversaciones con trabajadores de la entidad financiera, la incompletitud de los datos se debe principalmente a dos causas:

- Dentro del sistema interno, no es necesario completar toda la información para cerrar una venta, en consecuencia, para los vendedores resulta conveniente, en términos de tiempo, completar la tramitación solo con información parcial.
- Por algún problema en el sistema, no se ha guardado en el repositorio la totalidad de la información ingresada. Esta anomalía ha ocurrido especialmente en la variable “profesión”.

Estas omisiones de información ocurren independientemente de los valores de los datos, entonces, se asume que dado un atributo específico, el evento de un elemento faltante es un suceso aleatorio considerándose así un mecanismo de incompletitud en los datos MCAR. No obstante, algunos atributos son más propensos a no estar presentes (como por ejemplo profesión y aquellos campos no obligatorios para concretar ventas) y esta diferencia de probabilidad de completitud es considerada también.

Desde el punto de vista de los datos relacionados a las cuentas de los clientes, la información es más precisa. Los clientes pueden tener dinero invertido en cuatro cuentas distintas:

- Cuenta de Cotización Obligatoria (CCO): en esta cuenta se deposita el dinero recaudado mensualmente de acuerdo al ahorro previsional legal.
- Cuenta de Cotización Voluntaria (CCV): también denominada Ahorro Previsional Voluntario (APV), en esta cuenta se deposita el dinero recaudado mensualmente de forma voluntaria, con fines previsionales.
- Cuenta de Depósitos Convenidos (CDC): son aportes en dinero para Ahorro Previsional Voluntario, realizados de forma esporádica o mensual. Estos aportes, hasta un tope de 900 UF anuales, no constituyen remuneración, de modo que no están afectos a impuesto.
- Cuenta de Ahorro Voluntario: es un instrumento de inversión con fines no previsionales, es decir, luego de ser depositados, estos pueden ser rescatados en cualquier momento.

Se denomina obligatoria a la cuenta CCO dado que corresponde a los dineros provenientes del descuento previsional legal del cliente, sin embargo, no es necesario que esta cuenta esté contratada para adquirir alguna cuenta voluntaria, puesto que el cliente podría estar trabajando el dinero de su descuento legal en otra AFP.

Para los productos CCO y CDC cada cliente puede tener solo una cuenta asociada, pero para los productos CCV y CAV un cliente puede utilizar distintos regímenes tributarios, cada uno de ellos relacionados con una cuenta distinta. Además, el dinero de cada cuenta antes descrita puede estar invertido en hasta dos de los cinco fondos de riesgo disponibles (A, B, C, D y E). La Figura 4 esquematiza las posibilidades de inversión que tiene cada cliente.

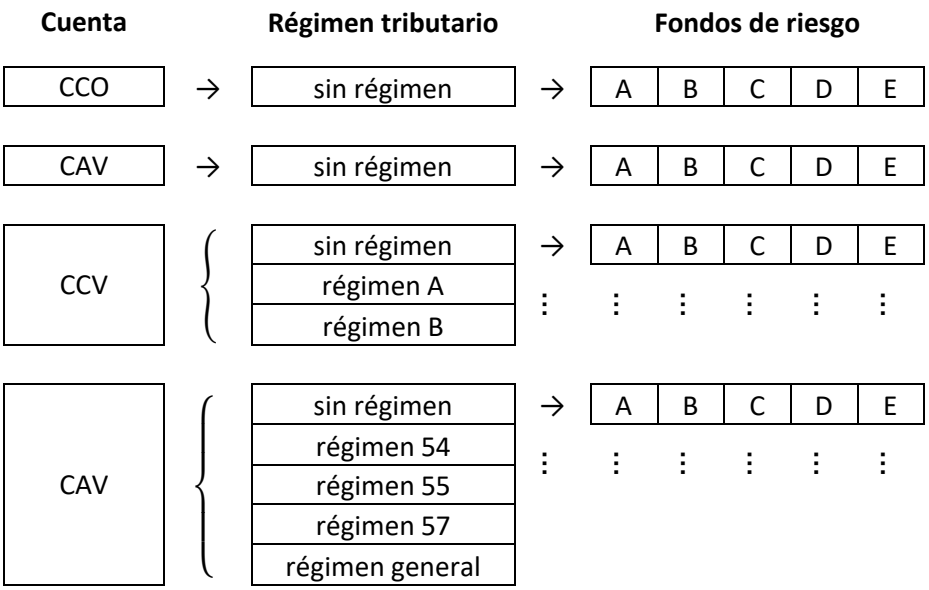

Figura 4. Representación de las posibilidades de inversión.
Fuente: Elaboración propia.

Para poder agrupar la información de cuentas en una sola fila por cliente –obteniendo así una estructura adecuada para su análisis– se agregan los distintos montos de regímenes tributarios y fondos de riesgo a solo un valor total por cuenta y se indica a través de variables binarias si el dinero se invierte o no en cada régimen y fondo. Además se cuenta con la fecha en que se contrató la cuenta. Para descripción extendida de las variables utilizadas en el análisis ver Anexo 1.

La Figura 5 muestra la distribución de los clientes con y sin cuentas voluntarias contratadas. Se observa que para las tres cuentas se dispone de más información de clientes que no han contratado una cuenta que de los que sí lo han hecho, siendo más notorio el desbalance en la cuenta CDC. Esta situación podría conllevar a un problema, el cual es discutido en la sección 3.3.1.

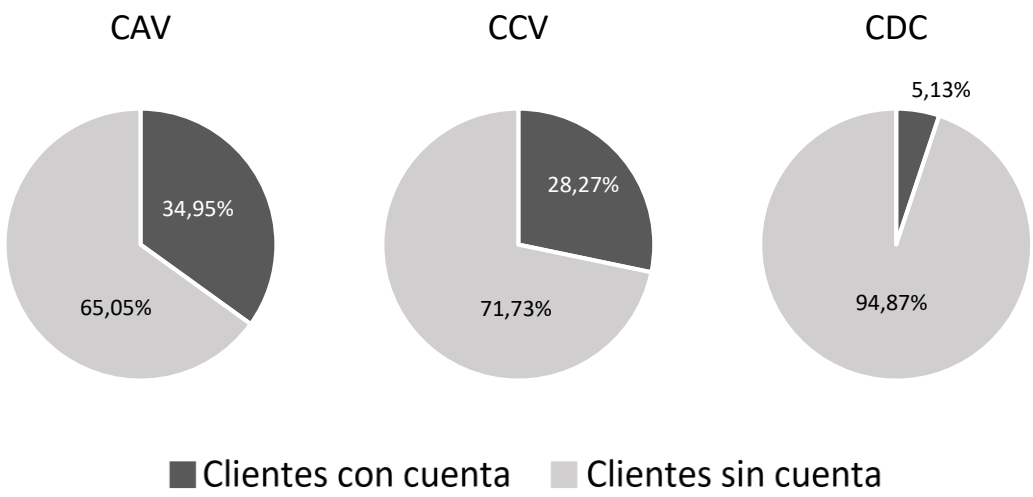

Figura 5. Proporciones de clientes con y sin cuentas voluntarias contratadas.
Fuente: Elaboración propia.

Finalmente, para describir las cuentas contratadas, la Tabla 2 muestra su media y desviación estándar, mientras que la Figura 6 presenta histogramas de las distribuciones de los montos, en escala logarítmica.

Cuenta	Media	Desviación estándar
CAV	2,69	21,99
CCV	4,23	20,47
CDC	15,18	66,26

Tabla 2. Media y desviación estándar de inversiones en cuentas voluntarias en millones de pesos.

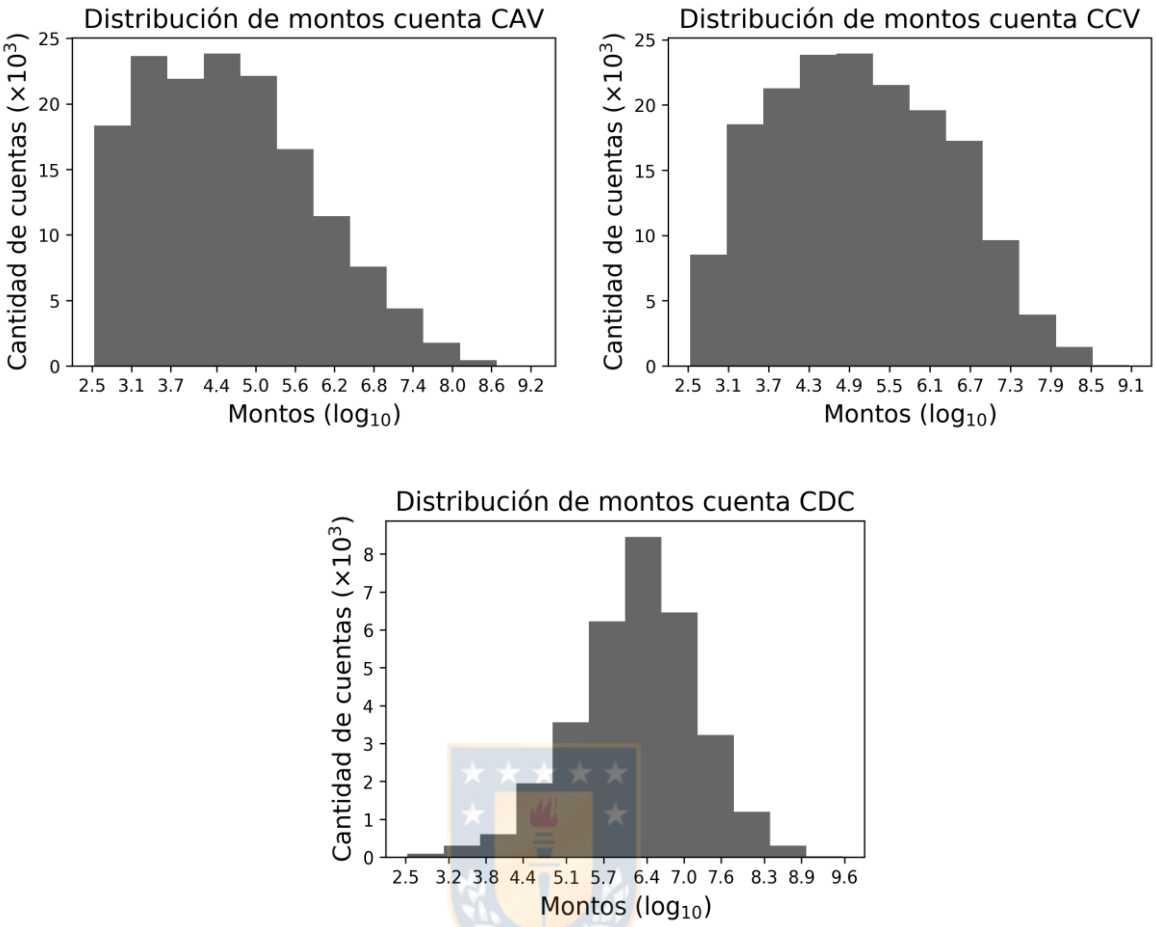

Figura 6. Distribución de montos por cuenta en escala logarítmica.

Fuente: Elaboración propia

3.2. Definición formal del problema

3.2.1. Problema de Clasificación

Predecir la probabilidad de que cada cliente de la entidad financiera contrate cada una de las cuentas adicionales disponibles (CCV, CDC y CAV). Esto se traduce en un problema de clasificación binaria, en donde la variable dependiente utilizada para entrenar los modelos se define como:

$$y_{i,c} = \begin{cases} 1, & \text{si el cliente } i \text{ ha contratado (alguna vez) el producto } c \\ 0, & 0\ ,\ e.o.c. \end{cases}$$

donde $i \in \{1, \dots, \mathrm{n}\}$, con n = número de clientes, $c \in \{\mathrm{CCV}, \mathrm{CDC}, \mathrm{CAV}\}$.

La salida esperada entregada por los modelos de clasificación es la variable $p_{i,c}$ correspondiente a la probabilidad de que $y_{i,c} = 1$.

3.2.2. Problema de Regresión

Predecir el monto que cada cliente de la entidad financiera invertiría en cada producto adicional, si es que fuera contratado. Se define la variable dependiente $t_{i,c} \in \mathbb{R}$ como el monto total en CLP invertido por el cliente i en la cuenta c.

3.2.3. Ranking Final

El ranking final de usuarios se construye en función de la multiplicación entre las variables $p_{i,c}$ y t_{ic}:

$$r_{i,c} = p_{i,c} \times t_{ic},$$

donde $r_{i,c}$ representa la esperanza de la inversión realizada por cliente i en la cuenta j. De esta forma, un cliente con un alto valor de $r_{i,c}$ resulta atractivo al marketing directo.

3.3. Preprocesamiento

3.3.1. Desbalance de clases

El problema que se desprende de la Figura 5, como ya se mencionó anteriormente, consiste en que se tienen datos de menos clientes que han contratado cada cuenta (clase de interés) que de los que no. Esto podría provocar que los modelos entrenados bajo estas condiciones tiendan a clasificar con mayor precisión la clase mayoritaria pero de muy mal forma a la clase minoritaria (Ganganwar, 2012).

Para subsanar el problema existen por un lado enfoques orientados a los datos y por otro lado enfoques orientados a los algoritmos (Ganganwar, 2012).

Desde el lado de los datos existen técnicas para realizar un *undersampling* de la clase mayoritaria, es decir, utilizar un menor número de elementos en relación al conjunto original. Cuando no se cuenta con suficientes datos, es posible realizar un *oversampling*, proceso en el cual artificialmente se generan datos de la clase minoritaria para balancearla con la clase

mayoritaria. También es posible realizar una combinación de ambas técnicas (Lemaître, 2017).

El enfoque de resolución del problema orientado a los algoritmos gira entorno a asignar costos distintos a las clases con el objetivo de contrarrestar su desbalance (Ganganwar, 2012).

Dado que se cuenta con un conjunto de datos con el orden de los cientos de miles de elementos, se decide realizar un *undersampling* aleatorio de la clase mayoritaria hasta igualar el número de elementos de la clase minoritaria.

3.3.2. Atributos con demasiadas categorías distintas

En general, los algoritmos de minería de datos solo aceptan como entrada válida datos numéricos, entonces, es necesario codificar las variables categóricas. La codificación normalmente utilizada para este propósito es *one hot encoding*, mediante la cual se crea una variable artificial o *dummy* por cada categoría distinta, de esta forma, para una fila dada solo una de las variables artificiales tendrá el valor de 1 indicando la categoría a la que pertenece, mientras que el resto de las variables artificiales tendrá un valor de 0. La Figura 7 ilustra un ejemplo de *one hot encoding.*

id	color	y
0	rojo	1
1	verde	1
2	rojo	0
3	azul	1
4	verde	0
5	rojo	1

One hot encoding →

	color			
id	rojo	verde	azul	y
0	1	0	0	1
1	0	1	0	1
2	1	0	0	0
3	0	0	1	1
4	0	1	0	0
5	1	0	0	1

Figura 7. Ejemplo de *one hot encoding*. A la izquierda los datos originales, a la derecha los datos luego de ser codificados, donde se genera una variable artificial por cada categoría (color) distinta. Para cada fila, la variable artificial con valor 1 indica la categoría a la que pertenece el elemento en el conjunto original.

Fuente: Elaboración propia.

Si es que se utilizara *one hot encoding* en las variables categóricas del conjunto de datos con el cual se está trabajando, se tendrían que agregar miles de variables artificiales para su codificación, debido a la gran cantidad de categorías presentes en, por ejemplo, las variables

"empleador" y "actividad económica" (ver Tabla 1). Aumentar en tal medida la dimensionalidad de los datos complejizaría demasiado la resolución del problema.

Una codificación alternativa es *target encoding* (Micci-Barreca, 2001), el proceso básico de este método consta de dos pasos:

- Para todos los elementos de una categoría dada, calcular la media de la variable de respuesta.
- Reemplazar la variable categórica por la media calculada.

De esta forma es posible obtener una representación numérica de las variables categóricas sin aumentar la dimensionalidad del problema. La Figura 8 ilustra un ejemplo de *target encoding.*

id	color	y
0	rojo	1
1	verde	1
2	rojo	0
3	azul	1
4	verde	0
5	rojo	1

Target encoding →

$\bar{y}_{rojo} = 0.\bar{6}$

$\bar{y}_{verde} = 0.5$

$\bar{y}_{azul} = 1$

id	color	y
0	$0.\bar{6}$	1
1	0.5	1
2	$0.\bar{6}$	0
3	1	1
4	0.5	0
5	$0.\bar{6}$	1

Figura 8. Ejemplo de *target encoding*. A la izquierda los datos originales, a la derecha los datos luego de ser codificados. El nuevo valor de la variable categórica, representa el promedio de la variable de respuesta de la categoría (color) a la cual pertenecía el elemento originalmente.
Fuente: Elaboración propia.

La principal cualidad que posee *one hot enconding* es que mantiene la totalidad de la información puesto que, una vez codificados, los datos pueden decodificarse al dominio original, a costa del aumento de dimensionalidad ya mencionado. Por su parte, *target encoding* consigue una representación numérica de las variables categóricas sin aumenta su dimensionalidad, sin embargo, este proceso supone una pérdida de información debido a que la decodificación al dominio original no es siempre posible.

Para buscar un balance en el *trade off* entre una dimensionalidad adecuada y la mantención de la información, para codificar las variables categóricas del problema, se decide utilizar las dos técnicas descritas, siguiendo la siguiente estrategia: para los atributos con relativamente

pocas categorías (“estado civil” y “AFP actual”) se utiliza *one hot encoding* y para los demás atributos con relativamente muchas categorías se emplea *target encoding*. Como paso previo adicional se eliminan todas las categorías que tuvieran menos de 10 elementos, dado que estas aportan poca información y complejizan el problema.

3.3.3. Escalado de Datos

Para ayudar al mejor funcionamiento de los modelos de predicción y estandarizar la comparación de errores de predicción entre atributos, todos los datos son transformados a valores entre 0 y 1 utilizando el escalado MinMax, definido por:

$$x_{ij}^{esc} = \frac{x_{ij} - x_j^{min}}{x_j^{max} - x_j^{min}},$$

donde x_j^{max} y x_j^{min} son los valores máximos y mínimos por atributo, respectivamente.

3.4. Imputación

Como se ha descrito en el capítulo 2, uno de los enfoques para lidiar con datos no observados es la imputación. Una buena imputación es conseguida cuanto los datos predichos se asemejan a los datos reales, sin embargo, medir esto resulta la mayoría de las veces imposible puesto que los datos reales no son conocidos.

Para analizar el comportamiento de los algoritmos de imputación aplicados al conjunto de datos de trabajo, se propone experimentar con subconjuntos incompletos creados artificialmente a partir de los datos completos conocidos y así poder comparar datos imputados con los datos realmente observados. A continuación, se describen los pasos necesarios para la experimentación:

1. Encontrar el subconjunto de elementos que presentan todos sus atributos completos, este conjunto se denominará C.
2. Eliminar secuencialmente datos C, simulando el mecanismo de incompletitud del conjunto inicial (ver sección 3.4.1), de modo de obtener conjuntos con distintos niveles de completitud.
3. Imputar los conjuntos incompletos generados artificialmente.

4. Comparar los datos reales de C con los datos imputados.

3.4.1. Mecanismo de generación de conjuntos incompletos

El mecanismo de incompletitud de los datos descrito en la sección 3.1. indica que, para un atributo específico, un elemento no observado es un suceso aleatorio con una probabilidad de ocurrencia dada, entonces, la clave para generar conjuntos con distintos niveles de incompletitud es encontrar la incompletitud de cada atributo para cada nivel. Para lograr esto se define las funciones $p_m(p^*)$, en donde p^* denota la incompletitud media objetivo en fracción de todos los atributos, $m \in \{1,2,\dots,\mathrm{M}\}$, con M igual al número de atributos demográficos, así p_m resulta la incompletitud en fracción necesaria de cada atributo para obtener la incompletitud media p^*.

Para encontrar la forma de las funciones $p_m(p^*)$ se describen 4 situaciones:

- Cuando la incompletitud media es 0, todos los atributos tienen una incompletitud de 0, es decir, se conocen todos los datos.
- Para la incompletitud media del conjunto original, denotada por $\bar{p}^o$, la incompletitud de cada atributo, denotada por p_m^o, es conocida desde la distribución original (ver Figura 3), donde:

$$\bar{p}^o = \frac{1}{M}\sum_{m=1}^{M} p_m^o.$$

- Cuando la incompletitud media es 1, todos los atributos tienen una incompletitud de 1, es decir, no se conoce dato alguno.
- Se asume que las tasas de cambio entre los 3 puntos definidos anteriormente son lineales.

Es posible llegar a una solución que incluya las condiciones antes mencionadas a través de una función lineal por tramos que interpole, para cada atributo, los puntos $(0,0)$, $(\bar{p}^o, p_m^o)$ y $(1,1)$. La necesidad de utilizar una función por tramos radica en que al aumentar proporcionalmente el nivel de incompletitud –con respecto al nivel de incompletitud original por atributo– habrían variables que llegarían rápidamente a no tener datos, imposibilitando con esto su posterior imputación.

Utilizando esta notación se puede llegar fácilmente a la función por tramos:

$$p_m(p^*) = \begin{cases} \frac{p_m^o}{\bar{p}^o} \cdot p^*, & \text{si } p^* < \bar{p}^o \\ \frac{p_m^o - 1}{\bar{p}^o - 1} \cdot (p^* - 1) + 1, & \text{si } p^* \geq \bar{p}^o \end{cases}.$$

La Figura 9 ilustra, con un ejemplo de conjunto de datos de dos atributos, cómo se puede emplear la función para determinar el nivel de incompletitud de cada variable a partir de la incompletitud media objetivo. El primer paso consiste en encontrar la incompletitud original (p_m^o) para los atributos 1 y 2 (0.5 y 0.3, respectivamente) y la incompletitud media original ($\bar{p}^o$) asociada a ellos (0.4) demarcadas en líneas segmentadas rojas. Con estos datos es posible construir una función para cada atributo la cual permita calcular su incompletitud necesaria (p_m) a partir de cualquier incompletitud media objetivo (p^*). Por ejemplo, si se desea tener un conjunto con un 20% de incompletitud, los atributos 1 y 2 debiesen presentar 25% y 15% de elementos faltantes, respectivamente (línea segmentada gris), mientras que si se desea un conjunto con un 70% de incompletitud media, los atributos 1 y 2 debiesen presentar 75% y 65% de elementos faltantes, respectivamente (línea segmentada azul).

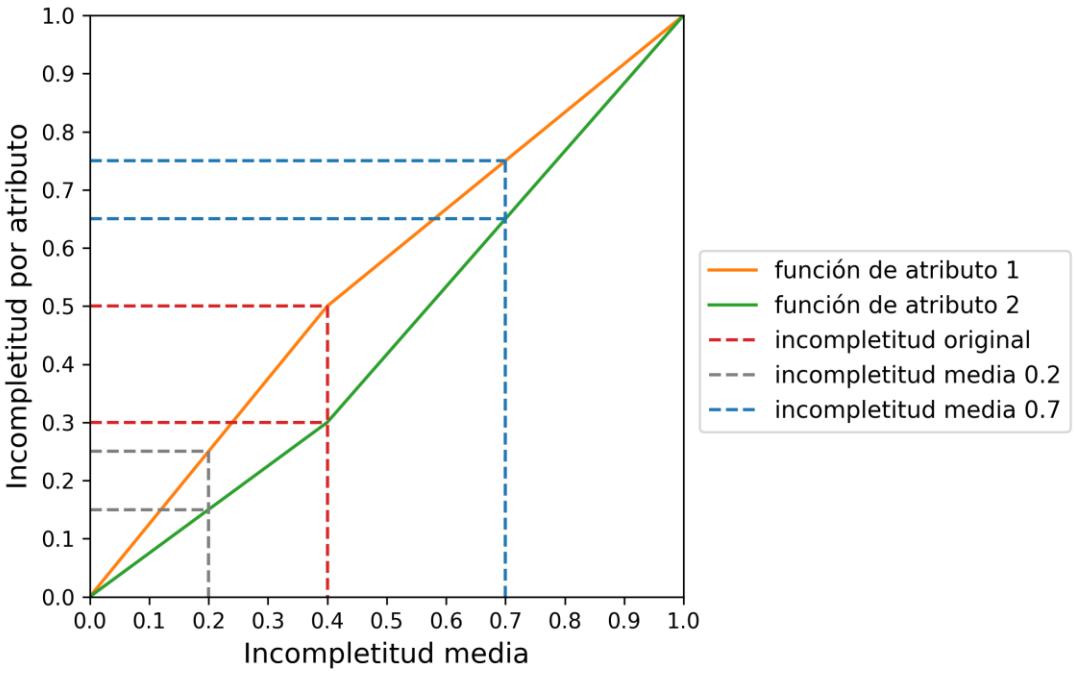

Figura 9. Ejemplo de utilización de función de incompletitud. Las líneas continuas anaranjada y verde representan las funciones de los atributos 1 y 2 respectivamente, y sus niveles de incompletitud original son denotados por las líneas segmentadas rojas. Las líneas segmentadas gris y azul representan los niveles de incompletitud por atributo necesarios para lograr las incompletitudes medias de 0.2 y 0.7, respectivamente.

Fuente: Elaboración propia.

3.4.2. Algoritmos de Imputación

Se ha seleccionado implementar los algoritmos de missForest y MICE para las tareas de imputación debido a que ambos poseen la capacidad de trabajar con datos compuestos por variables numéricas y categóricas.

Adicionalmente, se ha decidido implementar GAIN debido a los mejores resultados obtenidos al aplicarlo en conjuntos de prueba, en comparación precisamente de los algoritmos de missForest y MICE (Yoon, Jordon, & van der Schaar, 2018).

3.4.3. Métricas de Evaluación

Para evaluar y comparar los distintos modelos de imputación se analizan por separado los errores de predicción entre variables numéricas y categóricas. Para las variables numéricas se utiliza el RMSE entre los datos estimados y reales, definido por:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(\hat{x}_{ij} - x_{ij})^2 \cdot m_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{M} m_{ij}}},$$

en donde $m_{ij} = 1$ si x_{ij} es un dato faltante y $m_{ij} = 0$ si x_{ij} es un dato observado. Mientras que para las variables categóricas se utilizó la proporción de clasificados falsamente (PFC[13]), definida por:

$$PFC = \frac{\text{número de datos categóricos imputados incorrectamente}}{\text{número de datos categóricos imputados}}.$$

En ambos casos, valores cercanos a 0 denotan un buen desempeño, mientras que valores cercanos a 1 denotan un mal desempeño.

[13] PFC, del inglés *proportion of falsely classified.*

3.5. Predicciones

3.5.1. Algoritmos de Clasificación y Regresión

Para resolver los problemas planteados en la sección 3.2 se utilizan modelos de XGBoost entrenados sobre el conjunto de datos incompleto y luego se comparan con modelos del mismo algoritmo entrenado sobre datos imputados para así evaluar la efectividad del manejo de datos incompletos de XGBoost.

Además, se utilizan modelos de Random Forest para comparar XGBoost con un modelo basado en árboles de decisión y que no admite elementos faltantes, complementado por un enfoque de imputación.

Finalmente, se entrenan modelos solo sobre el subconjunto de datos completos y se comparan con los resultados anteriormente obtenidos, de modo de evaluar la hipótesis planteada.

Los modelos son entrenados a partir de los atributos demográficos en conjunto con los atributos asociados a las cuentas, sin embargo, no forman parte de las variables predictoras los datos de la cuenta de respuesta. Por ejemplo, para predecir comportamientos de la cuenta opcional CAV, se utilizan los datos demográficos de los clientes, sumados a la información disponible de las cuentas CCO, CDC y CCV.

Con el objetivo de poder comparar modelos optimizados de forma similar, tanto para los modelos de XGBoost como para los de Random Forest, se optimizan los hiperparámetros asociados a la profundidad de los árboles y el criterio mínimo para la creación de una nueva partición, mediante validación cruzada (ver sección 3.5.3).

3.5.2. Métricas de Evaluación

Para comprender de mejor forma las métricas de evaluación se presenta la matriz de confusión de una clasificación binaria (Figura 10), en ella se representan los resultados de la siguiente manera:

- En el cuadrante superior izquierdo se ubican los verdaderos positivos (TP[14]), aquellos elementos que eran realmente positivos y el modelo los predijo como positivos.
- En el cuadrante superior derecho se ubican los falsos positivos (FP[15]), aquellos elementos que eran negativos pero que el modelo los predijo como positivos.
- En el cuadrante inferior izquierdo se ubican los falsos negativos (FN[16]), aquellos elementos que eran positivos pero que el modelo los predijo como negativos.
- En el cuadrante inferior derecho se ubican los verdaderos negativos (TN[17]), aquellos elementos que eran realmente negativos y el modelo los predijo como negativos.

		Clases reales	
		Positivo	Negativo
Clases predichas	Positivo	Verdaderos positivos (TP)	Falsos positivos (FP)
	Negativo	Falsos negativos (FN)	Verdaderos negativos (TN)

Figura 10. Matriz de confusión.
Fuente: Elaboración propia.

La exactitud o *accuracy* (ACC) es una de las métricas más utilizadas para evaluar modelos de clasificación. En palabras sencillas la exactitud es la fracción o porcentaje de predicciones realizadas correctamente y se define por:

$$ACC = \frac{\text{elementos predichos correctamente}}{\text{elementos totales}} = \frac{TP + TN}{TP + TN + FP + FN}.$$

No obstante, al entrenar y evaluar modelos sobre conjuntos desbalanceados solo mediante la exactitud podría provocar que estos tiendan a clasificar con mayor precisión la clase mayoritaria pero de muy mal forma a la clase minoritaria. Por este motivo, la principal métrica para evaluar los modelos será la exactitud balanceada o *balanced accuracy* (BAL_{ACC}), la cual es un promedio de la exactitud por clase, definida por:

[14] TP del inglés *true positives*
[15] FP del inglés *false positives*
[16] FP del inglés *false negatives*
[17] TP del inglés *true negatives*

$$BAL_{ACC} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right).$$

Si bien la principal métrica de evaluación utilizada es la exactitud balanceada, también serán reportadas las métricas de exactitud, precisión y exhaustividad, estas dos últimas son discutidas a continuación. En palabras sencillas la precisión, *precision* o *positive predictive value* (PPV) representa qué fracción de los elementos clasificados como positivos son realmente positivos y está definida por:

$$PPV = \frac{TP}{TP + FP}.$$

Por su parte, la exhaustividad, *recall* o *true positive rate* (TPR) consiste en la fracción de los elementos positivos que fueron clasificados correctamente y está definida por:

$$TPR = \frac{TP}{TP + FN}.$$

Por otro lado, para evaluar los modelos de regresión se utiliza la raíz del error medio cuadrático (RMSE) como métrica principal, definida por:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{t}_i - t_i)^2},$$

donde t_i es el monto real invertido por el cliente i, $\hat{t}_i$ es el monto estimado y N es el número de datos. Además, se reportan las métricas raíz del error medio cuadrático normalizada ($NRMSE$; Oba et al., 2003) y error porcentual absoluto medio ($MAPE$), definidos por:

$$NRMSE = \sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N}(\hat{t}_i - t_i)^2}{\mathrm{Var}(t)}},$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{t}_i - t_i|}{t_i},$$

donde $\mathrm{Var}(t)$ denota la varianza de los datos observados.

3.5.3. Validación Cruzada

Todas las evaluaciones se realizan mediante validación cruzada, es decir, se separan los datos en conjuntos de entrenamiento y prueba, con el objetivo de ajustar los modelos sobre un grupo de datos (conjunto de entrenamiento) y probar su capacidad de predicción sobre datos que el algoritmo no ha visto antes (conjunto de prueba), de modo de obtener resultados que generalicen de mejor manera predicciones futuras.

Para separar los datos en los conjuntos de prueba y entrenamiento se utiliza el método *k-Folds*, el cual opera dividiendo los datos aleatoriamente en *k* subconjuntos disjuntos de aproximadamente igual tamaño, luego, se utilizan *k*-1 subconjuntos para el entrenamiento del modelo y el subconjunto restante se emplea para la validación a través del cálculo de métricas de desempeño, este proceso se repite k veces y en cada iteración un subconjunto distinto es utilizado para la validación. Al finalizar, *k* realizaciones de cada métrica son obtenidas permitiendo calcular datos estadísticos tales como media y desviación estándar. Se decide utilizar el método de *k-Folds* dividiendo los datos en 5 subconjuntos.

CAPÍTULO 4. RESULTADOS Y ANÁLISIS

4.1. Resultados de Imputación

A partir de los 12.624 perfiles completos, es decir, aquellos elementos en que todos sus atributos son conocidos, disponibles en el conjunto de datos total, se fueron eliminando atributos paulatinamente siguiendo el proceso descrito en la sección 3.4.1, obteniendo distintos niveles de incompletitud en el conjunto experimental (ver Figura 12). En cada nivel de incompletitud, los conjuntos fueron imputados y dichas estimaciones fueron comparadas con los datos originales, los algoritmos de imputación utilizados fueron GAIN, missForest y MICE.

Algunas de las clases de los atributos codificados por *one hot encoding* ("estado civil" y "AFP actual") en el subconjunto de datos completos se encuentran poco pobladas e incluso, después de la eliminación de atributos para la generación de los conjuntos experimentales, se quedan sin elementos. Por este motivo se decide no trabajar con estas dos variables durante los experimentos de imputación, quedando así 49 variables.

Para GAIN se utilizan arquitecturas de ANN con dos capas ocultas de 20 y 10 nodos, donde la capa de entrada y salida tiene la misma cantidad de nodos que variables en el conjunto, esta arquitectura se representa en la Figura 11. El entrenamiento se realiza por lotes de 32 elementos y los parámetros son ajustados mediante optimización estocástica –optimizador de Adam (Kingma & Ba, 2014)–. Los hiperparámetros *número de épocas* y α son optimizados a través de validación cruzada.

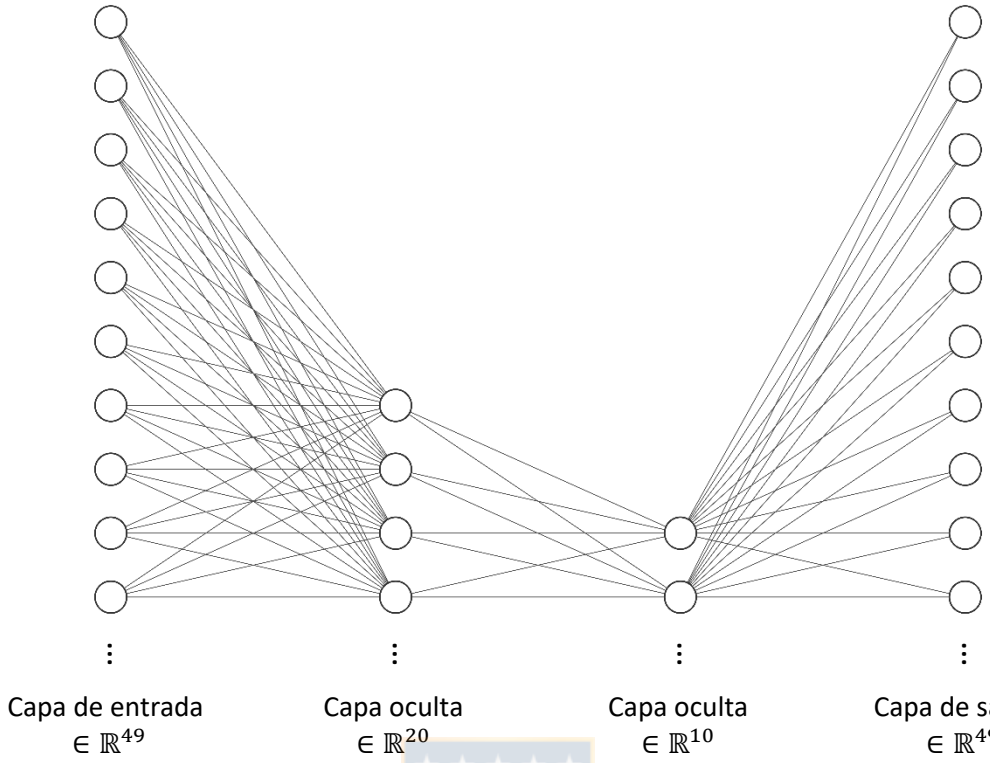

Figura 11. Representación de las arquitecturas de ANN de GAIN. Tanto el generador como el discriminador cuentan con capas de entrada y salida con 49 nodos (de acuerdo a la dimensionalidad del conjunto a imputar), además de dos capas ocultas de 20 y 10 nodos.
Fuente: Elaboración propia.

La Figura 12 grafica los resultados de las imputaciones experimentales en términos de RMSE para las variables numéricas (gráfico superior) y la proporción de elementos mal clasificados (PFC) para la variables categóricas (gráfico inferior).

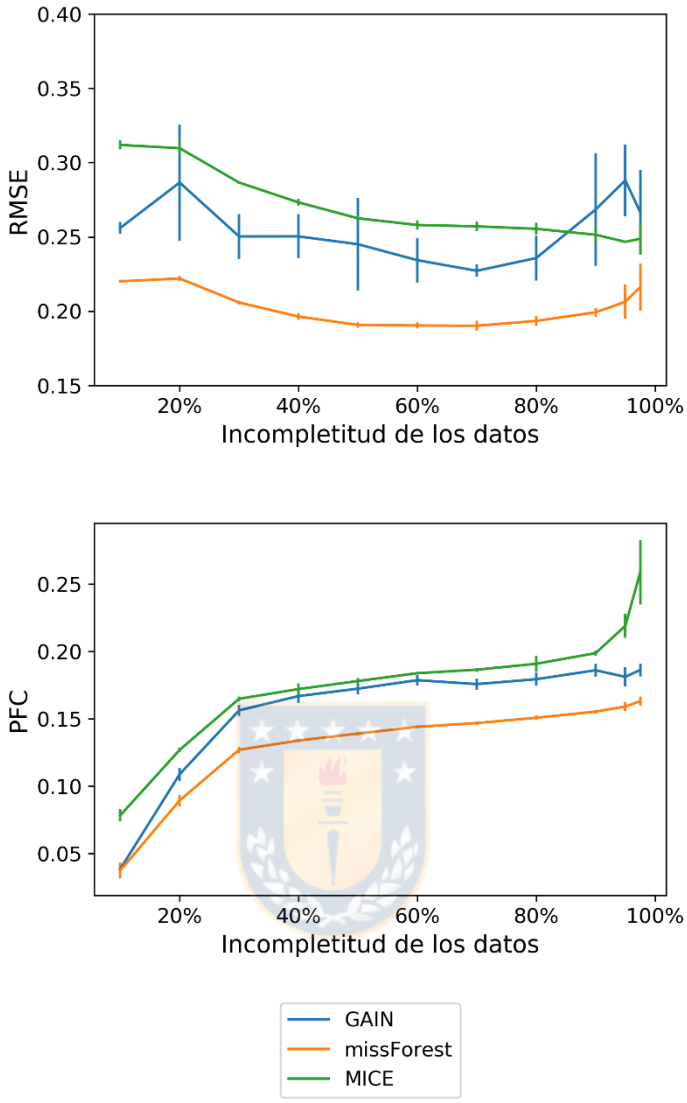

Figura 12. Resultados de imputaciones experimentales.
Fuente: Elaboración propia.

El primer hecho que resalta al analizar los resultados, es que missForest fue el algoritmo que obtuvo mejor desempeño tanto en términos de RMSE como PFC en todos los niveles de incompletitud de los datos. Por su parte, GAIN, muestra mejores resultados general que MICE hasta el 80% de incompletitud.

A pesar de que missForest fue el método de imputación con mejores resultados experimentales, dos motivos dificultan su aplicación en el conjunto de datos completo:

1. Utilizando un computador con un procesador Intel® Core™ i5-7200U (2.5 GHz, 3 MB de caché y 2 núcleos), 12 GB de RAM y sin GPU, luego de 24 horas missForest no es capaz de finalizar una iteración con la totalidad de los datos y de acuerdo a las imputaciones experimentales, el algoritmo tardaría entre 5 y 10 iteraciones en converger. Además, este proceso debiese ser llevado a cabo 3 veces, debido a que al utilizar *target encoding* los datos predictores cambian al tratarse de variables de respuesta distintas.
2. El algoritmo de missForest entrena un modelo de Random Forest (compuesto de múltiples árboles de decisión) para cada variable con datos faltantes, en cada iteración, además, cada iteración depende de los resultados de la iteración anterior de manera recursiva. En base a estos antecedentes, resulta difícil almacenar todos los modelos que componen el algoritmo y missForest no posee ningún método para guardar los modelos ya ajustados, por tal, no se puede utilizar para datos nuevos fuera del conjunto de entrenamiento.

Para resolver el problema de missForest relacionado al gran costo computacional que conlleva imputar los 846.932 perfiles, se decide imputar por lotes de tamaño menor que el conjunto total, de modo de reducir dicho costo.

Con el objetivo de determinar un tamaño adecuado para los lotes, se evalúa el error de las imputaciones de missForest sobre el conjunto experimental, en función de la cantidad de datos utilizados en cada imputación. Los resultados de este experimento son exhibidos en la Figura 13, en ella se observa que tanto el RMSE como el PFC se estabilizan al utilizar el orden de los 10^4 datos, de esta forma se decide imputar con missForest por lotes de 15.000 elementos. Luego de imputar todos los lotes, el conjunto es desordenado aleatoriamente.

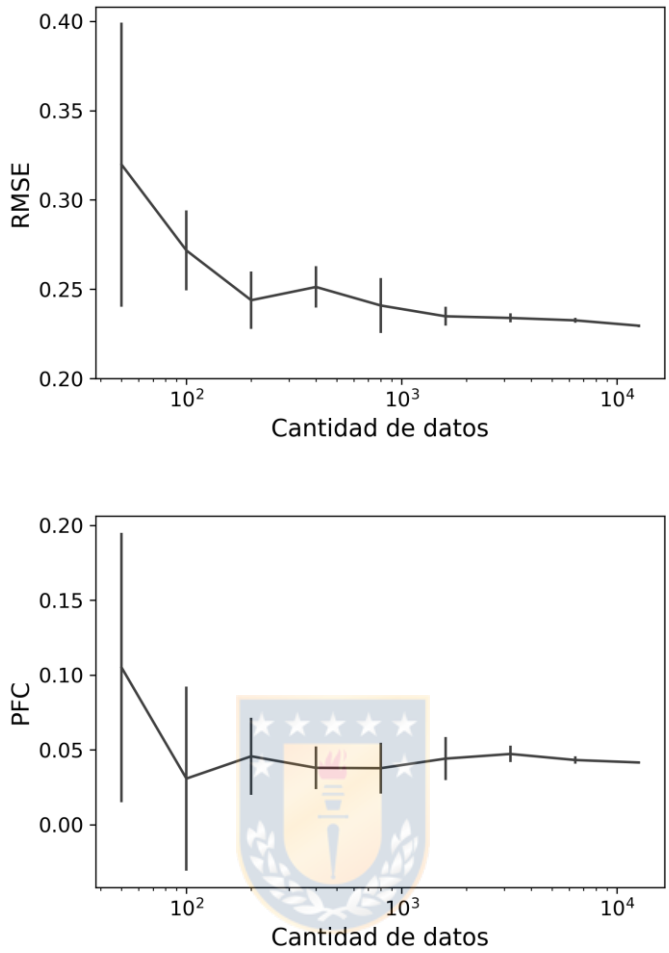

Figura 13. Error de imputación de missForest en función de la cantidad de datos utilizados, en escala logarítmica. El gráfico superior muestra el RMSE en función de la cantidad de datos, mientras que el gráfico inferior muestra el PFC en función de la cantidad de datos.
Fuente: Elaboración propia.

En relación al segundo problema de missForest, referente a no poder utilizar modelos ya entrenados para imputar datos nuevos, se decide imputar el conjunto completo mediante GAIN, con el fin de tener un método complementario, que si permita su utilización en datos nuevos.

4.2. Resultados de Clasificación

Para predecir la probabilidad de que cada cliente de la entidad financiera contrate cada uno de las cuentas adicionales disponibles, se entrenaron modelos de XGBoost y Random Forest.

Los 846.932 datos fueron separados en conjuntos de entrenamiento y validación, a través del método *k-Folds* (ver sección 3.5.2), obteniendo 5 conjuntos de entrenamiento de aproximadamente 677.000 datos y 5 conjuntos de validación de aproximadamente 169.000 datos, recordando que para cada cuenta específica los conjuntos de entrenamiento fueron balanceados (ver sección 3.3.1), reduciendo con esto su tamaño.

Para determinar los hiperparámetros de cada modelo de XGBoost se siguió la siguiente estrategia:

- En base a los hiperparámetros por defecto, se establece una taza de aprendizaje alta (0.1) y se determina el número de iteraciones óptimo para dicha tasa, mediante validación cruzada. La tasa de aprendizaje alta permite que los modelos tomen menos iteraciones en converger y así lograr una optimización de parámetros más rápida.
- Con la taza de aprendizaje y el número de iteraciones fijos, se optimizan el resto de los hiperparámetros mediante validación cruzada.
- Una vez determinados los hiperparámetros para los distintos modelos, es posible compararlos y encontrar el que presente mejor desempeño.
- Para el mejor modelo, se disminuye la tasa de aprendizaje y se determina un nuevo número de iteraciones óptimo, en función de la nueva tasa (0.01). La disminución de la tasa de aprendizaje implica un mayor costo computacional del entrenamiento a cambio de una posible mejora en el desempeño del modelo.

Por su parte, los modelos de Random Forest fueron entrenados siguiendo la siguiente estrategia:

- Se establecen 100 árboles por modelo.
- Con el número de árboles fijos, se optimizan el resto de los hiperparámetros mediante validación cruzada.
- Una vez determinados los hiperparámetros para los distintos modelos, es posible compararlos y encontrar el que presente mejor desempeño.
- Para el mejor modelo, determina el número de árboles óptimo.

La Tabla 3 resume el desempeño en términos de exactitud balanceada de los modelos de XGBoost (XGB) ajustados sobre los datos incompletos originales, XGBoost ajustados sobre los

datos imputados mediante GAIN y missForest (MF), y Random Forest (RF) bajo la misma condición.

	XGB	GAIN + XGB	GAIN + RF	MF + XGB	MF + RF
CDC	**0,8388 ± 0,0017**	**0,8386 ± 0,0007**	**0,8364 ± 0,0014**	0,8316 ± 0,0008	0,8298 ± 0,0010
CCV	**0,7714 ± 0,0004**	0,7701 ± 0,0006	0,7681 ± 0,0003	0,7678 ± 0,0009	0,7677 ± 0,0002
CAV	**0,7202 ± 0,0007**	0,7174 ± 0,0011	0,7148 ± 0,0016	0,7068 ± 0,0005	0,7032 ± 0,0011

Tabla 3. Comparación de exactitud balanceada entre modelos (media ± desviación est.).
Fuente: Elaboración propia.

De acuerdo a la Tabla 3, XGBoost ajustado a los datos originales fue el modelo que obtuvo mejores resultados para las tres cuentas, aunque para la cuenta CAV y considerando las desviaciones estándar, tanto XGBoost ajustado a los datos originales, como Random Forest y XGBoost entrenado sobre los datos imputados previamente con GAIN, obtienen resultados similares. El modelo entrenado sin imputación previa es seleccionado dado que evita este paso en su implementación. La Tabla 4 expande el informe del modelo incluyendo las métricas de evaluación secundarias.

	ACC	BAL_ACC	PPV	TPR
CDC	0,8311 ± 0,0012	0,8388 ± 0,0017	0,2124 ± 0,0033	0,8473 ± 0,0043
CCV	0,7755 ± 0,0003	0,7714 ± 0,0004	0,5781 ± 0,0007	0,7620 ± 0,0009
CAV	0,7088 ± 0,0008	0,7202 ± 0,0007	0,5619 ± 0,0012	0,7581 ± 0,0009

Tabla 4. Resultados de XGBoost entrenado sobre datos incompletos (media ± desviación est.).
Fuente: Elaboración propia.

Al comparar los valores de la exactitud y exactitud balanceada estos se muestran similares en todas las cuentas, esto permite inferir que los modelos clasifican de forma similar tanto a la clase positiva como a la negativa, indicando que las medidas adoptadas para solucionar el desbalance de clases fueron efectivas. Por otra parte, los valores de TPR indican la fracción de clientes que han contratado la cuenta en cuestión y que fue correctamente capturado por el modelo. Finalmente, la precisión (PPV) toma a todos los clientes que el modelo señaló han contratado una cuenta e indica que fracción de estos realmente lo ha hecho.

La precisión de 21.24% de la cuenta CDC se explica por el desbalance de las clases expuesto en la Figura 5. Para facilitar el análisis de la incidencia del desbalance de las clases en los resultados, en la Tabla 5 se comparan la proporción de elementos positivos por cuenta, precisión del conjunto total y la precisión del subconjunto del 5% más probable. El objetivo

principal del trabajo es orientar el marketing directo de una empresa el cual, por lo general, dispone de recursos limitados. En consecuencia, la elección más racional sería dirigir el marketing directo hacia los clientes que posean la mayor probabilidad de reaccionar positivamente al producto ofrecido. A modo de ejemplo, supongamos que la empresa posee recursos suficientes para aplicar un marketing diferenciado al 5% del total de los clientes, el cual sería dirigido a las personas que los modelos indiquen tengan mayor probabilidad de pertenecer a la clase positiva. De este trabajo se desprende la precisión del 5% más probable, presente en la Tabla 5.

	PR	PPV	5% más probable
CDC	0,0513	0,2124	0,4640
CCV	0,2827	0,5781	0,9547
CAV	0,3495	0,5619	0,9262

Tabla 5. Comparación entre proporción de elementos positivos por cuenta, precisión del conjunto total y la precisión del subconjunto del 5% más probable.
Fuente: Elaboración propia.

De acuerdo a la Tabla 5, si se escogiera un cliente al azar solo se tendría una probabilidad de 5.13% de que este haya contratado la cuenta CDC, mientras que si se escogiera un cliente que el modelo clasifico como positivo dicha probabilidad aumentaría 4 veces llegando a 21.24%.

Análogamente, la probabilidad de escoger un cliente que haya contratado la cuenta CDC aumentaría 9 veces comparamos el seleccionar un cliente al azar (5.13%), con escoger a alguno de los clientes dentro del 5% con mayor probabilidad estimada de contratación (46.40%). Siguiendo esta estrategia, la cuenta CCV, por ejemplo, alcanza una precisión de 95.47%.

Una de las ventajas de trabajar con algoritmos basados en árboles de decisión radica en que estos permiten identificar a los atributos más incidentes del modelo de acuerdo a la cantidad de veces que son utilizados como criterios de división durante la construcción de los árboles. La Figura 14 grafica a los 10 atributos más importante por cuenta, cuya incidencia es representada en el eje horizontal. La importancia de los atributos se encuentra normalizada de modo que su sumatoria entre todas las variables es igual a 1 y un mayor valor representa una mayor importancia.

Desde la observación de la Figura 14 se puede destacar que, para las tres cuentas, los tres atributos más importantes son la renta, la comuna de residencia y el monto total invertido en la cuenta obligatoria (CCO).

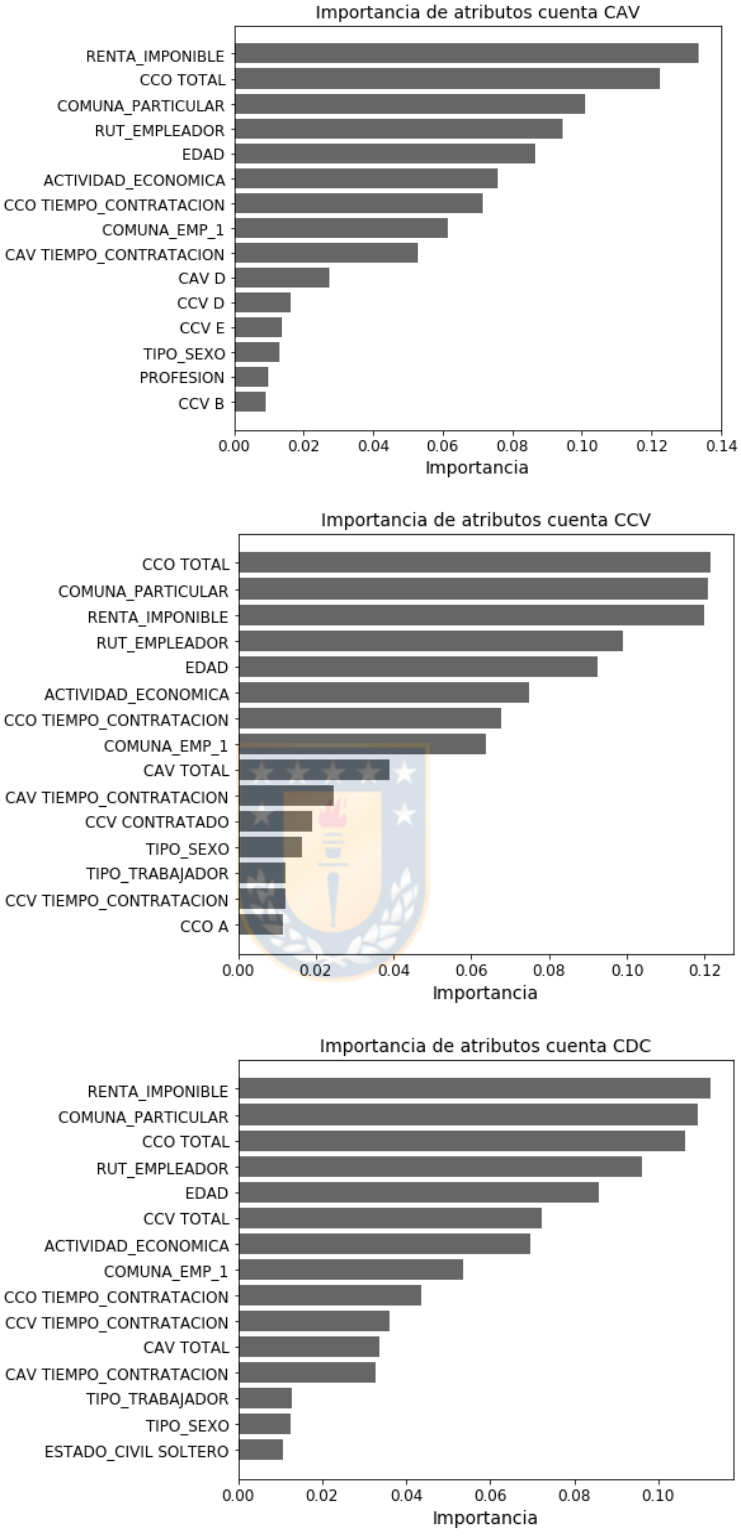

Figura 14. Importancia de atributos por cuenta.

Fuente: Elaboración propia.

Para finalizar el análisis de los modelos de clasificación, la Tabla 6 toma las exactitudes balanceadas de los modelos XGBoost ajustados a partir de datos incompletos exhibidos con anterioridad y los compara con los resultados obtenidos al seguir el enfoque más sencillo a la hora de lidiar con datos incompletos: el de eliminar las filas incompletas hasta llegar a un subconjunto en donde todos las variables sean observadas y luego ajustar un modelo de una forma tradicional. Para realizar una comparación justa, el modelo elegido para ser ajustado a los datos completos es XGBoost.

	Datos totales	Subconjunto completo
CDC	0,8388 ± 0,0017	0,8372 ± 0,0068
CCV	0,7714 ± 0,0004	0,6933 ± 0,0049
CAV	0,7202 ± 0,0007	0,6595 ± 0,0058

Tabla 6. Comparación de exactitud balanceada de modelos ajustados a la totalidad de datos y ajustados al subconjunto de datos completos (media ± desviación est.).
Fuente: Elaboración propia.

De acuerdo a la Tabla 6, para las cuentas CCV y CAV se observa un considerable aumento en el desempeño de las predicciones al utilizar la totalidad de los datos, recompensando el trabajo realizado en relación al análisis de los datos incompletos. No obstante, la mejora en el desempeño de la cuenta CDC es ínfima y al contrastarla con las desviaciones estándares resulta no significativa.

4.3. Resultados de Regresión

En base a la experiencia adquirida desde el análisis de los modelos de clasificación, el primer algoritmo que se utiliza para predecir los montos invertidos en cada cuenta es XGBoost, entrenado a partir de la totalidad de los datos, incluyendo aquellos incompletos. La Tabla 7 muestra sus resultados, expresados en millones de pesos.

	RMSE	MAPE	NRMSE
CDC	54,24	86,40	0,8186
CCV	16,02	316,95	0,7827
CAV	20,16	420,22	0,9167

Tabla 7. Resultados de modelos de XGBoost de regresión, RMSE expresado en millones de pesos.
Fuente: Elaboración propia.

Al observar los RMSE expuestos en la Tabla 7 y compararlos con las medias y desviaciones de las cuentas expuestos en la Tabla 2, es posible señalar que, a pesar del trabajo de optimización de parámetros, las predicciones son demasiado imprecisas como para poder incluirlas en la construcción del ranking de clientes.

Por otro lado, el NRMSE normaliza el RMSE en término de la varianza de los datos. Por ejemplo, una predicción sencilla sería utilizar la media para estimar los valores desconocidos, en cuyo caso el valor del NRMSE sería de 1, entonces, valores inferiores indicarían una mejora en la exactitud de la predicción.

La métrica MAPE nos indica, por ejemplo, que las predicciones para la cuenta CAV, en promedio, tuvieron un error absoluto de 420 veces el valor que se deseaba predecir. Errores absolutos de tales magnitudes son posibles debido a la distribución de los datos: de acuerdo a la Figura 6, una gran fracción de las inversiones en las cuentas son inferiores a los 10^4 pesos, por otro lado, según la Tabla 2, las medias de las inversiones fluctúan los órdenes de 10^6 y 10^7 pesos, entonces, predicciones dentro de ese rango son totalmente verosímiles. Bajo este contexto, si una cuenta con una inversión verdadera del orden de los 10^4 pesos fue estimada por el modelo en el orden de los 10^6 pesos, obtendría un error absoluto del orden de 10^2. En coherencia con lo descrito, las cuentas CAV y CCV son las que presentan mayores porcentajes de cuentas con inversiones bajas y mayores valores en la métrica MAPE.

En base al análisis de los resultados expuestos, se cree que resulta deficiente predecir montos totales en función de los perfiles de los clientes y más que seguir probando modelos, la solución radica en cambiar el enfoque de la predicción. Una idea a explorar en trabajo futuro consiste en predecir montos anualizados, mensuales o estandarizados de alguna forma en función del tiempo, labor para la cual se necesitaría información adicional.

CAPÍTULO 5. CONCLUSIONES Y TRABAJO FUTURO

En la búsqueda de un ranking que permita dirigir los esfuerzos de marketing directo de una entidad financiera que permita identificar a los clientes más propensos a contratar un producto y a los más rentables a la vez, más de 800.000 perfiles fueron analizados. Puesto que algunas de las variables del conjunto presentan elementos no observados, ha sido necesario incursionar en distintos enfoques para lidiar con información faltante.

A través de conjuntos incompletos generados artificialmente, fue posible realizar imputaciones experimentales a distintos niveles de completitud en los datos, usando los algoritmos MICE, GAIN y missForest. Al comparar los resultados de estos algoritmos missForest supera significativamente a GAIN y MICE en términos de RMSE y PFC, independientemente del nivel de incompletitud en los datos. Sin perjuicio de lo anterior, el costo computacional de aplicar missForest al conjunto incompleto total resulta excesivo y el hecho de que el algoritmo no permite evaluaciones posteriores a datos nuevos puesto que no es posible guardar sus modelos entrenados, resuelven a GAIN como mejor opción para imputar los datos de clientes nuevos.

Desde el ámbito de la construcción del ranking, destacan los malos resultados de los modelos de regresión encargados de estimar los montos que los clientes invertirían en los diversos productos. Los errores relativos de las predicciones, en promedio, son entre 86 y 460 veces la magnitud que se desea estimar, concluyendo que sus resultados no son lo suficientemente buenos como para ser utilizados. El análisis de las distintas métricas de regresión, permite suponer que el problema de su mal desempeño radica en intentar predecir montos ahorrados totales, en vez de hacerlo en base a cifras mensuales o anualizadas, proceso que necesitaría información adicional para realizarse.

Por contra parte, los algoritmos de clasificación muestran buenos desempeños siendo XGBoost, entrenado a partir de datos incompletos, el modelo con mejores resultados, superando al enfoque de imputación. Dentro de los resultados representativos, el modelo obtuvo exactitudes balanceadas entre 72% y 83%, y al utilizar el enfoque de considerar al 5% de los clientes que el modelo predice más probables como positivos, se puede aumentar la probabilidad de encontrar un cliente con cuenta contratada hasta en 9 veces, comparado con una elección aleatoria. Se concluye además que el análisis de enfoques que lidian con datos

faltantes condujo a una mejora significativa en al menos dos de las tres cuentas, comparado a utilizar modelos entrenados solo a partir del subconjunto de datos completos.

El enfoque de entrenar un modelo por producto es posible debido a que son relativamente pocos productos a analizar. Una alternativa y posible propuesta de trabajo futuro consiste en entrenar una única red neuronal con una salida asociada a cada producto.

Finalmente, otra propuesta de trabajo futuro consiste en realizar un estudio del costo de realizar una conversión mediante el marketing directo en función de la probabilidad entregada por los modelos. Dicho estudio podría ayudar a definir límites óptimos para maximizar ganancias.

REFERENCIAS

Akiyama, K., Alberdi, A., Alef, W., Asada, K., Azulay, R., Baczko, A.-K., . . . others. (2019). First M87 Event Horizon Telescope Results. I. The Shadow of the Supermassive Black Hole. *The Astrophysical Journal Letters*.

Akyildiz, I. F., Pompili, D., & Melodia, T. (2005). Underwater acoustic sensor networks: research challenges. *Ad hoc networks, 3*(3), 257-279.

Azur, M. J., Stuart, E. A., Frangakis, C., & Leaf, P. J. (2011). Multiple imputation by chained equations: what is it and how does it work? *International journal of methods in psychiatric research, 20*(1), 40-49.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Wadsworth.

Cao, L.-J., & Tay, F. E. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks, 14*(6), 1506-1518.

Cardoso, D. O., Carvalho, D. S., Alves, D. S., Souza, D. F., Carneiro, H. C., Pedreira, C. E., . . . França, F. M. (2016). Financial credit analysis via a clustering weightless neural classifier. *Neurocomputing, 183*, 70-78.

Chavan, S., Jadhav, A., Suryagandh, P., & Sharma, N. (2018). Data Mining Techniques to Improve Customer Relationships Management. *IRJET*.

Chen, J., & Shao, J. (2000). Nearest neighbor imputation for survey data. *Journal of Official statistics, 16*(2), 113-131.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, (pp. 785-794). San Francisco, CA.

Crone, S. F., Lessmann, S., & Stahlbock, R. (2006). The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research, 173*(3), 781-800.

Eirola, E., Doquire, G., Verleysen, M., & Lendasse, A. (2013). Distance estimation in numerical data sets with missing values. *Information Sciences, 240*, 115-128.

Etaiwi, W., Biltawi, M., & Naymat, G. (2017). Evaluation of classification algorithms for banking customer's behavior under apache spark data processing system. *Procedia Computer Science, 113*, 559-564.

Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering, 2*(4), 42-47.

García-Laencina, P. J., Sancho-Gómez, J. L., & Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: a review. *Neural Computing and Applications, 19*(2), 263-282.

Haykin, S. S., Haykin, S. S., Haykin, S. S., Elektroingenieur, K., & Haykin, S. S. (2009). *Neural networks and learning machines.* Pearson Upper Saddle River.

Hong, X., Gerla, M., Wang, H., & Clare, L. (2002). Load balanced, energy-aware communications for mars sensor networks. *Proceedings, IEEE Aerospace Conference*.

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

Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization.*

Lemaître, G. a. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research, 18*(1), 559-563.

Linoff, G. S., & Berry, M. J. (2011). *Data mining techniques: for marketing, sales, and customer relationship management.* John Wiley & Sons.

Little, R. J., & Rubin, D. B. (2002). *Statistical analysis with missing data.* Wiley.

Lu, C.-J., Lee, T.-S., & Chiu, C.-C. (2009). Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems, 47*(2), 115-125.

Lu, G., & Copas, J. B. (2004). Missing at random, likelihood ignorability and model completeness. *The Annals of Statistics, 32*(2), 754-765.

Micci-Barreca, D. (2001). A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter, 3*(1), 27-32.

Mitik, M., Korkmaz, O., Karagoz, P., Toroslu, I. H., & Yucel, F. (2017). Data Mining Approach for Direct Marketing of Banking Products with Profit/Cost Analysis. *The Review of Socionetwork Strategies, 11*(1), 17-31.

Niyagas, W., Srivihok, A., & Kitisin, S. (2006). Clustering e-banking customer using data mining and marketing segmentation. *ECTI Transactions on Computer and Information Technology, 2*(1), 63-69.

Oba, S., Sato, M. A., Takemasa, I., Monden, M., Matsubara, K. I., & Ishii, S. (2003). A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics, 19*(16), 2088-2096.

Quinlan, J. R. (1994). *C4. 5: programs for machine learning.*

Redner, R. A., & Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM review, 26*(2), 195-239.

Sánchez, E. M., Clempner, J. B., & Poznyak, A. S. (2015). A priori-knowledge/actor-critic reinforcement learning architecture for computing the mean-variance customer portfolio: the case of bank marketing campaigns. *Engineering Applications of Artificial Intelligence, 46*, 82-92.

Stekhoven, D. J., & Bühlmann, P. (2011). MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics, 28*(1), 112-118.

Thomas, L., Crook, J., & Edelman, D. (2017). *Credit scoring and its applications.* Siam.

Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. *Journal of artificial intelligence research, 6*, 1-34.

Xu, L., & Jordan, M. I. (1996). On convergence properties of the EM algorithm for Gaussian mixtures. *Neural computation, 8*(1), 129-151.

Yin, S., Ding, S. X., Xie, X., & Luo, H. (2014). A review on basic data-driven approaches for industrial process monitoring. *IEEE Transactions on Industrial Electronics, 61*(11), 6418-6428.

Yoo, P. D., Kim, M. H., & Jan, T. (2005). Machine learning techniques and use of event information for stock market prediction: A survey and evaluation. *Conference on Computational Intelligence for Modeling, Control and Automation (CIMCA 2005)*, (pp. 835-841). Piscataway, NJ.

Yoon, J., Jordon, J., & van der Schaar, M. (2018). Gain: Missing data imputation using generative adversarial nets. *arXiv preprint arXiv:1806.02920*.

Yu, Q., M. Y., Eirola, E., Van Heeswijk, M., SéVerin, E., & Lendasse, A. (2012). Regularized extreme learning machine for regression with missing data. *Neurocomputing, 102*, 45-51.

Anexo 1. Descripción de variables

		Descripción	Dominio
	TIPO_SEXO	Sexo del cliente	$\in bin \begin{cases} 1: masculino \\ 0: femenino \end{cases}$
	EDAD	Edad del cliente en años	$\in \mathbb{N}$
	PAIS_NACIMIENTO	Codificado con *target encoding*. Se eliminaron los elementos poco poblados.	$\lvert nacimiento_{cat} \rvert = 46$ $nacimiento_{target} \in \mathbb{R} \rightarrow [0,1]$
	PAIS_NACIONALIDAD		$\lvert nacionalidad_{cat} \rvert = 50$ $nacionalidad_{target} \in \mathbb{R} \rightarrow [0,1]$
	PROFESION	Codificado con *target encoding*.	$\lvert prof_{cat} \rvert = 62$ $prof_{target} \in \mathbb{R} \rightarrow [0,1]$
	TIPO_TRABAJADOR	Diferenciación entre trabajadores dependientes e independientes.	$\in bin \begin{cases} 1: dependiente \\ 0: independiente \end{cases}$
	RENTA_IMPONIBLE	Monto de renta imponible en CLP	$\in \mathbb{N}$
	COMUNA_PARTICULAR	Codificado con *target encoding*.	$\lvert comuna\_part_{cat} \rvert = 347$ $comuna\_part_{target} \in \mathbb{R} \rightarrow [0,1]$
	RUT_EMPLEADOR	Codificado con *target encoding*.	$\lvert rut\_emp_{cat} \rvert = 7937$ $rut\_emp_{target} \in \mathbb{R} \rightarrow [0,1]$
	COMUNA_EMP	Codificado con *target encoding*.	$\lvert comuna\_emp_{cat} \rvert = 345$ $comuna\_emp_{target} \in \mathbb{R} \rightarrow [0,1]$
	ACTIVIDAD_ECONOMICA	Codificado con *target encoding*.	$\lvert actividad_{cat} \rvert = 595$ $actividad_{target} \in \mathbb{R} \rightarrow [0,1]$
ESTADO_CIVIL	**SOLTERO**	Codificación *One hot encoding* de ESTADO_CIVIL	$\in bin \begin{cases} 1: posee\ el\ estado\ civil \\ 0: no\ posee\ el\ estado\ civil \end{cases}$
	CASADO		
	CONVIVIENTE CIVIL		

	DIVORCIADO		
	VIUDO		
AFP_ACTUAL	**CUPRUM**	Codificación *One hot encoding* de AFP_ACTUAL	$\in bin \begin{cases} 1: pertenece\ a\ la\ AFP \\ 0: no\ pertenece\ a\ la\ AFP \end{cases}$
	CAPITAL		
	HABITAT		
	MODELO		
	PLANVITAL		
	PROVIDA		
CCO	**A**	Variables binarias que indican si el cliente posee o no saldo de la cuenta CCO en el fondo en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ fondo \\ 0: no\ posee\ esaldo\ en\ el\ fondo \end{cases}$
	B		
	C		
	D		
	E		
	TIEMPO_CONTRATACION	Tiempo en años que ha transcurrido desde que se contrató la cuenta	$\in \mathbb{N}$
	CONTRATADO	Variable binaria que indica si el cliente ha contratado o no alguna vez la cuenta	$\in bin \begin{cases} 1: ha\ contratado \\ 0: no\ ha\ contratado \end{cases}$
	TOTAL	Monto total en CLP de la cuenta	$\in \mathbb{R}$

CAV	R54	Variables binarias que indican si el cliente posee o no saldo de la cuenta CAV en el régimen en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ régimen \\ 0: no\ posee\ esaldo\ en\ el\ rég. \end{cases}$
	R55		
	R57		
	RG		
	A	Variables binarias que indican si el cliente posee o no saldo de la cuenta CAV en el fondo en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ fondo \\ 0: no\ posee\ esaldo\ en\ el\ fondo \end{cases}$
	B		
	C		
	D		
	E		
	TIEMPO_CONTRATACION	Variable binaria que indica si el cliente ha contratado o no alguna vez la cuenta	$\in \mathbb{N}$
	CONTRATADO	Variable binaria que indica si el cliente ha contratado o no alguna vez la cuenta	$\in bin \begin{cases} 1: ha\ contratado \\ 0: no\ ha\ contratado \end{cases}$
	TOTAL	Monto total en CLP de la cuenta	$\in \mathbb{R}$
CCV	RA	Variables binarias que indican si el cliente posee o no saldo de la cuenta CCV en el régimen en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ régimen \\ 0: no\ posee\ esaldo\ en\ el\ rég. \end{cases}$
	RB		
	A	Variables binarias que indican si el cliente posee o no saldo de la cuenta CCV en el fondo en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ fondo \\ 0: no\ posee\ esaldo\ en\ el\ fondo \end{cases}$
	B		

	C		
	D		
	E		
	TIEMPO_CONTRATACION	Tiempo en años que ha transcurrido desde que se contrató la primera cuenta	$\in \mathbb{N}$
	CONTRATADO	Variable binaria que indica si el cliente ha contratado o no alguna vez la cuenta	$\in bin \begin{cases} 1: ha\ contratado \\ 0: no\ ha\ contratado \end{cases}$
	TOTAL	Monto total en CLP de la cuenta	$\in \mathbb{R}$
CDC	**A**	Variables binarias que indican si el cliente posee o no saldo de la cuenta CDC en el fondo en cuestión.	$bin \begin{cases} 1: posee\ saldo\ en\ el\ fondo \\ 0: no\ posee\ esaldo\ en\ el\ fondo \end{cases}$
	B		
	C		
	D		
	E		
	TIEMPO_CONTRATACION	Tiempo en años que ha transcurrido desde que se contrató la cuenta	$\in \mathbb{N}$
	CONTRATADO	Variable binaria que indica si el cliente ha contratado o no alguna vez la cuenta	$\in bin \begin{cases} 1: ha\ contratado \\ 0: no\ ha\ contratado \end{cases}$
	TOTAL	Monto total en CLP de la cuenta	$\in \mathbb{R}$