UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

REDES NEURONALES PARA PREDICCIÓN DE PAGO DE DEUDAS DE CLIENTES DE UNA EMPRESA DE RETAIL FINANCIERO

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

BENJAMÍN MELLADO VILLANUEVA

PROFESOR GUÍA:
JUAN PABLO ROMERO GODOY

MIEMBROS DE LA COMISIÓN:
EDGARDO JULIO JUAN SANTIBÁÑEZ VIANI
FELIPE ESTEBAN VILDOSO CASTILLO

SANTIAGO DE CHILE
2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS
DE LA INGENIERÍA
POR: **BENJAMÍN MELLADO VILLANUEVA**
FECHA: 2021
PROF. GUÍA: Juan Pablo Romero G.

REDES NEURONALES PARA PREDICCIÓN DE PAGO DE DEUDAS DE CLIENTES DE UNA EMPRESA DE RETAIL FINANCIERO

Este trabajo se centra en una Fintech enfocada en brindar soluciones de cobranza digital, la cual utiliza diversas aristas de la inteligencia artificial, tanto para poder realizar gestiones a deudores sin la intervención humana como para optimizar las gestiones más tradicionales.

Debido a la pandemia ocurrida en el año de este trabajo, la empresa se vio enfrentada a dos principales problemas: La abrupta llegada de un gran número de clientes debido a la necesidad de abaratar costos utilizando la cobranza digital y la necesidad de reducir los costos internos debido a problemas en el flujo de caja por la recesión que afecto a los clientes de la compañía a causa de la pandemia mencionada.

Gracias a esto, se propone optimizar el uso de los recursos de gestión de deudores ya existentes en la empresa mediante la utilización de una predicción de la probabilidad de pago de deuda de los clientes, tanto para poder manejar los nuevos clientes como para reducir costos operativos. Más específicamente, el objetivo de este trabajo es "Diseñar, realizar y proponer aplicaciones de un modelo predictivo de pago para los deudores con mora menor a 30 días y con vencimiento en la primera quincena del mes de una empresa de retail financiero mediante algoritmos de clasificación utilizando redes neuronales". Esto con el fin de lograr un modelamiento que pueda generalizarse para otras empresas clientes.

Para realizar esto, se utilizó como metodología CRISP-DM, la cual se basa en un proceso iterativo y centrado en el negocio para el modelamiento de algoritmos de aprendizaje de máquinas, con 6 pasos que van desde el entendimiento del negocio hasta el planteamiento de posibles aplicaciones del modelo.

Los resultados del modelo base realizado con LIGHTGBM muestran un valor F1 de un 60,82 % en el set de testeo general y un 68,84 % en el top 10 % deudores con mayor monto facturado del set de testeo, mientras que la red neuronal presenta un F1 de 64,2 % y de 71,18 % para cada set, respectivamente.

Los resultados si bien positivos tienen un gran espacio de mejora. Debido al rendimiento similar de ambos modelos, se propone trabajar en un futuro con el preprocesamiento de los datos o con la base como tal. A pesar de esto, tanto a nivel de métricas de rendimiento como en el análisis monetario que será presentado más adelante, se ve que la red neuronal presenta marginalmente mejores resultados, lo cual se beneficia con la posibilidad de generalizar la red para tanto más clientes de la empresa.

Agradecimientos

Gracias papá, que aunque no estés con nosotros me inspiras cada día a ser la mejor versión de mi. Gracias mamá, que tu confianza plena en mi me da fe y seguridad en todo lo que hago. Gracias a ambos por su amor incondicional.

Gracias al resto de mi familia, en especial a mi hermanos Mono, Dani, Vicky y Mónica por su ejemplo, apoyo y cariño. Fueron son y serán muy importantes para mi.

Gracias a todo aquel que hizo mi paso por la universidad una grata experiencia, en especial a mis amigos José, Nati, Angélica, Nico, Yerko, Pablo, Nicole, Lucas, Tere, Marco, Ani, Diego, Eybie, Agustín y Pipe. No imagino estar donde estoy sin su compañía y ayuda.

Gracias por el apoyo para la realización de la tesis a todo integrante de la empresa, en especial a Gabriel y Arturo por su ayuda en la formulación del proyecto y la estructura del trabajo realizado.

Gracias también a los profesores Astrid Contreras, Juan Pablo Romero y Edgar Santibáñez por su apoyo y constante feedback a lo largo del curso.

Gracias.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción, Justificación y Objetivos

El rubro de la cobranza presenta un ambiente competitivo para cualquier participante de este, debido a la alta rotación en la asignación de una cartera de clientes por cobrar entre las diversas empresas. Así, para mantenerse competitivo dentro de la industria, un actor de esta debe poder demostrar resultados eficaces a sus clientes tanto en cantidad de deudores que pagan y cantidad de monto recuperado.

Bajo este contexto, una Fintech de cobranza, orientada a la cobranza con base en la inteligencia artificial para brindar herramientas y estrategias de cobranza de manera innovadora a sus clientes, busca, en base a esta misma inteligencia artificial, poder crear un método para predecir la probabilidad de pago de los deudores de sus clientes, para así poder tanto mejorar las métricas de rendimiento y a su vez poder optimizar sus recursos omnicanales de contacto con clientes.

Para esto, el memorista Benjamín Mellado Villanueva propone la utilización de un modelo de clasificación utilizando redes neuronales en base a la información que se tiene sobre los clientes, tanto demográfica como de comportamiento para predecir el pago o no pago de deuda que cada deudor posee. Así, a lo largo del siguiente trabajo, se evaluarán diversos métodos para generar las redes para poder llegar a un algoritmo óptimo y se propondrán diversas aplicaciones para este.

1.1. Características generales de la empresa

La empresa con la cual se trabajará decidió mantenerse en el anonimato, sin embargo, es necesario caracterizarla como una Fintech B2B enfocada en la cobranza. A términos generales, una Fintech proporciona nuevas soluciones financieras a sus clientes mediante la incorporación de nuevas tecnologías al mercado. Estas ofrecen soluciones tanto para personas físicas como para empresas (B2B) y son especialistas en áreas concretas de los servicios financieros [1]

Como se ha mencionado anteriormente, el área en donde la empresa brinda sus soluciones financieras es el área de la cobranza, y la principal tecnología que esta usa para brindar estas soluciones es el fuerte uso de inteligencia artificial, tanto para apoyar las estrategias de gestión de sus empresas clientes, como para brindar herramientas a estas mismas para la realización de los contactos con deudores. Con estas tecnologías la empresa tiene como misión el ayudar a las instituciones financieras a optimizar sus métodos de cobranza para lograr mayor eficiencia

y a su vez evitar la sobre gestión de deudores que puede causar molestia en el público.

La compañía a la fecha provee dos diferentes servicios a sus clientes. El primero y principal de estos es la instalación de un *Software as a Service* o plataforma digital con un cobro por renta la cual provee herramientas de reportería, asignación de carteras o clientes a gestionar y *business intelligence* para ayudar a las empresas a gestionar sus carteras de clientes de manera más eficiente, dando visibilidad de lo que sucede y ayudando a implementar estrategias de gestión. El segundo de estos es un servicio de agencia digital de cobranza con el cual, mediante el uso de herramientas digitales, en especial bots, se realizan parte de las gestiones de cobranza de sus clientes, cobrándole al cliente ya sea por recupero o por gestiones realizadas a cada deudor. Para realizar esto, la empresa tiene como principal decisión el identificar a qué clientes gestiones y con que fuerza.

Para finales del primer semestre del año 2020, la empresa cuenta con 15 clientes ubicados en 3 países distintos, los cuales se ubican en los sectores financieros, telecomunicaciones y retail. Por confidencialidad, las cifras de ventas concretas no serán presentadas, pero a modo de estimación, el precio de los servicios varía entre los $USD 5 mil y 20 mil.

En términos organizacionales, la empresa presenta una estructura jerárquica horizontal y posee una metodología la cual llama *remote-ready* al ser 100 % funcional con un trabajo remoto, presentando partes del equipo tanto en diversos países y la realización del trabajo es en base a los objetivos fijados.

La figura 1.1 muestra la estructura organizacional de la empresa.

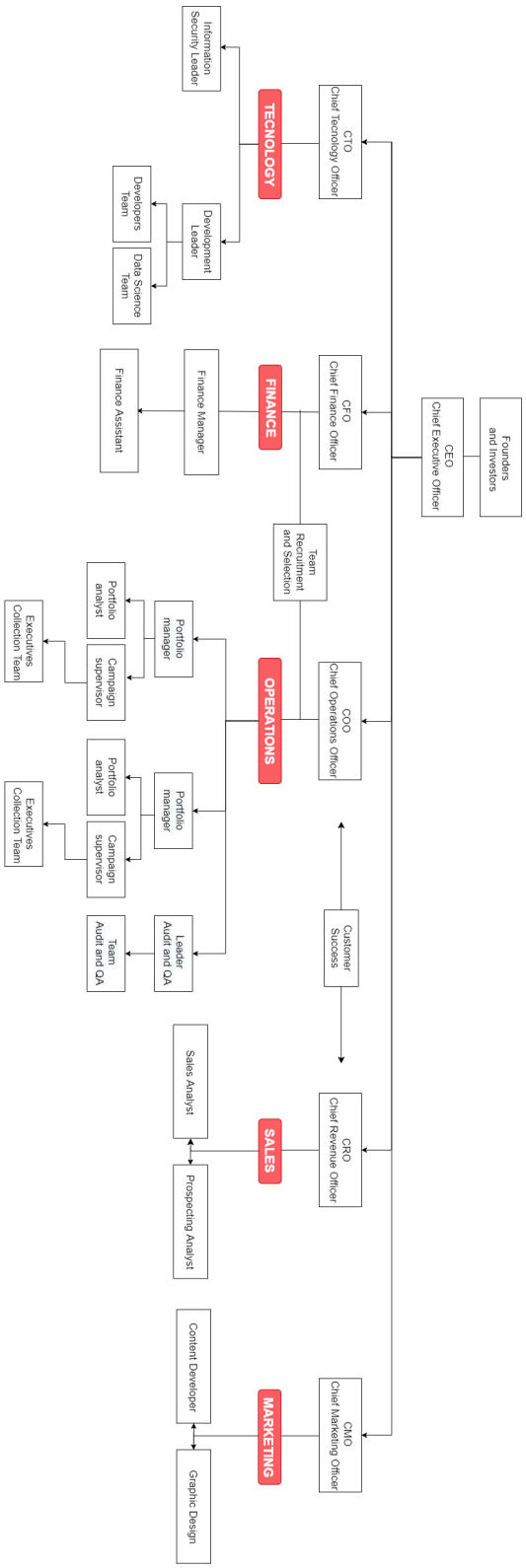

Figura 1.1: Organigrama de la empresa. Fuente: Proporcionado por la empresa

1.2. Mercado

Al ofrecer herramientas digitales mediante la utilización de softwares, el mercado del cual la empresa participa es el mercado de los softwares, en particular las Fintech orientadas a la cobranza. Existen diversos actores relevantes dentro de este mercado. En términos de cliente, las empresas que utilizan este tipo de software tienen en común el hecho de realizar cobranza masiva. De estas, los actores más relevantes son instituciones financieras, empresas de seguros, comunicaciones y retail, entre otras.

En las Figuras 1.2 y 1.3 se puede ver una distribución de los tipos de empresas existentes en el rubro Fintech.

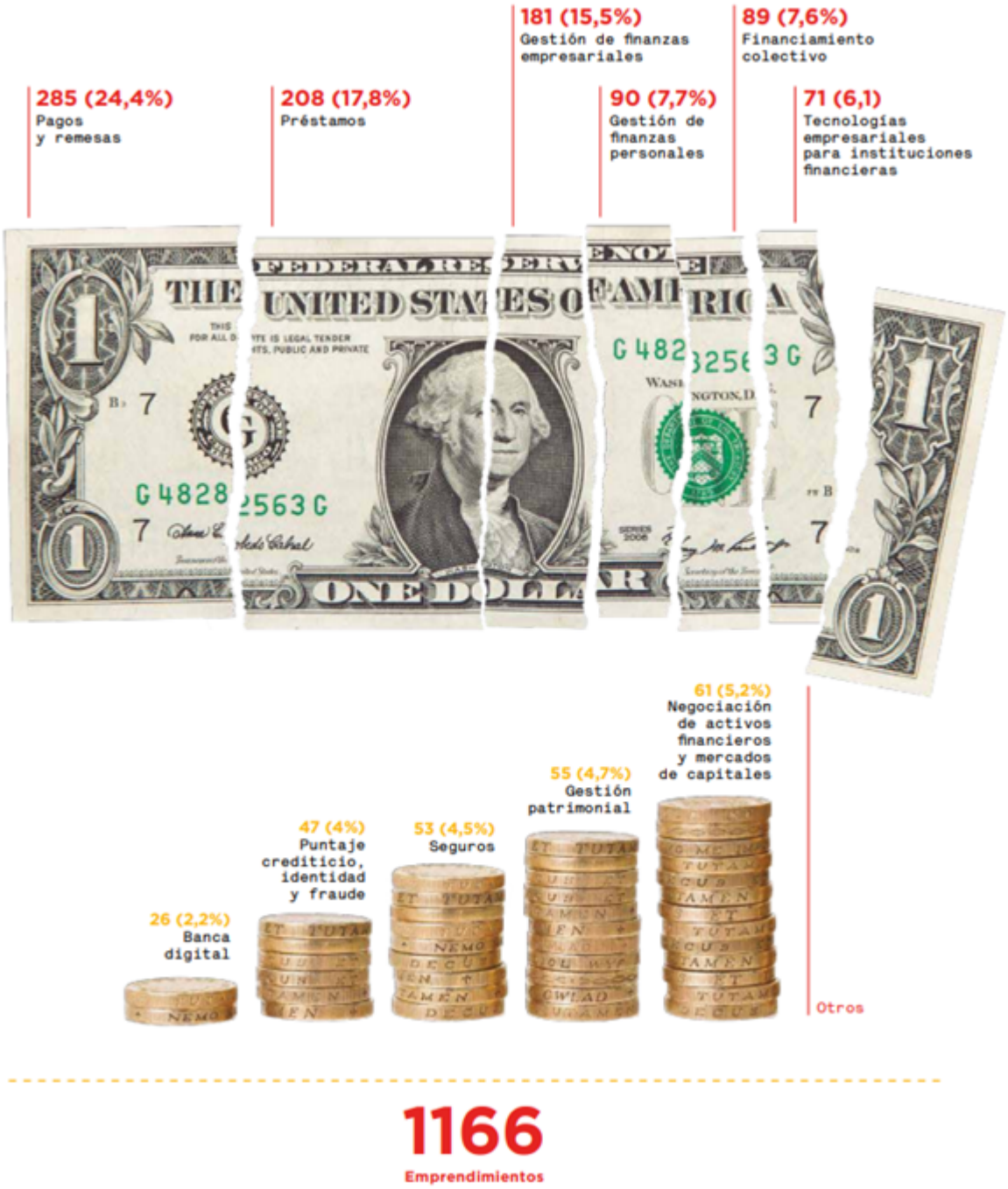

Figura 1.2: Distribución de segmentos Fintech. Se presentan cantidad de empresas en Chile junto con su porcentaje en relación al resto de Fintechs chilenas. Fuente: Encuesta BID y Finnovista (2018) [2].

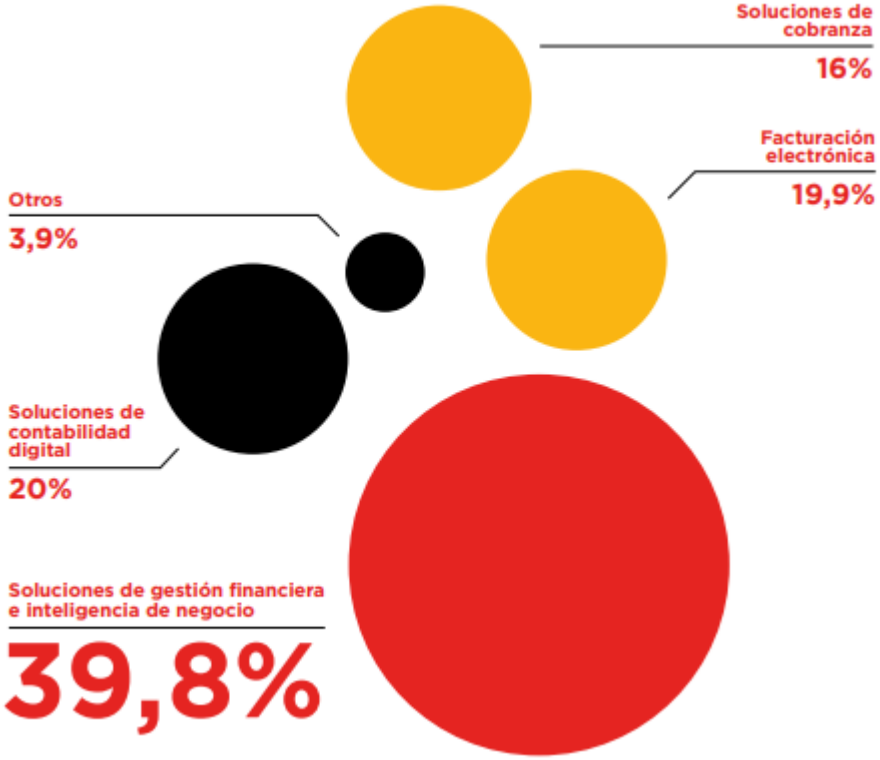

Figura 1.3: Subsegmentación: gestión de finanzas empresariales. Fuente: Encuesta BID y Finnovista (2018) [2].

Como se puede ver, la empresa irrumpe el mercado en un nicho poco explorado con respecto a las demás empresas financieras tecnológicas, el de soluciones de cobranza dentro de las gestiones de finanzas empresariales. Según Fintechile dentro de este segmento de gestiones de finanzas empresariales, se puede estimar una participación de mercado de la empresa viendo que los rangos de ingresos de este subsector el año 2018 fue de entre USD$100 mil y 500 mil. Sin embargo, ya que este sector está compuesto también empresas cuyas soluciones tecnológicas brindan servicios en otros ámbitos financieros como la contabilidad y el sistema de gestiones, además de haber empresas en este segmento dedicadas a soluciones para personas, es difícil poder estimar la participación de mercado, además de ser la cobranza una industria reservada al momento de divulgar información. [3]

La empresa presenta una fuerte ventaja competitiva frente a las demás dentro del rubro al no tan solo estar enfocada en el uso de inteligencia artificial, sino también justificar el uso de esta con los resultados que ha tenido con sus clientes y también por utilizar esta tecnología para ser menos invasivos de cara al deudor al momento de gestionar, lo que también ofrece un beneficio en la imagen de las empresas clientes de esta.

1.3. Desempeño organizacional

La empresa desde sus inicios el año 2018 se ha enfocado en una estrategia de crecimiento acelerado, utilizando sus ingresos para poder fomentar fuertemente el desarrollo del negocio,

tanto para sus herramientas y servicios principales como para generar y evaluar nuevos actores del negocio en diversas áreas de la tecnología. La figura 1.4 muestra el crecimiento porcentual de sus ingresos año a año.

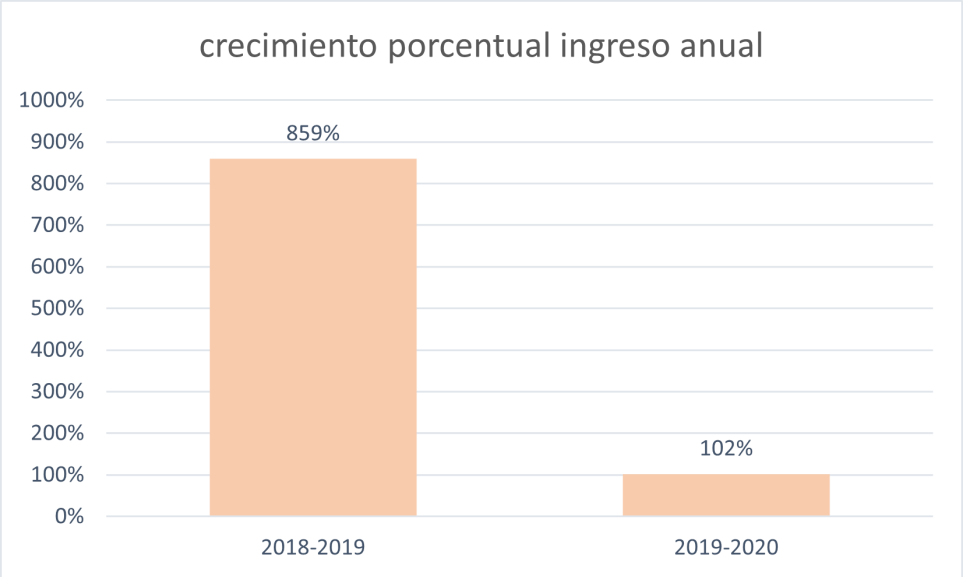

Figura 1.4: Crecimiento porcentual del ingreso anual de la empresa. Fuente: Elaboración propia.

En temas de consolidación, la empresa funciona de manera rápida y con cambios constantes, pero ha logrado definir las áreas de trabajo y los roles de cada una de estas y lograr tener un crecimiento constante en el número de clientes a los cuales proporciona el servicio. Además, ha logrado conseguir clientes en otros 5 países dentro de América latina. Esto ya que desean llegar a ser líder en cobranzas digitales en todo Latinoamérica.

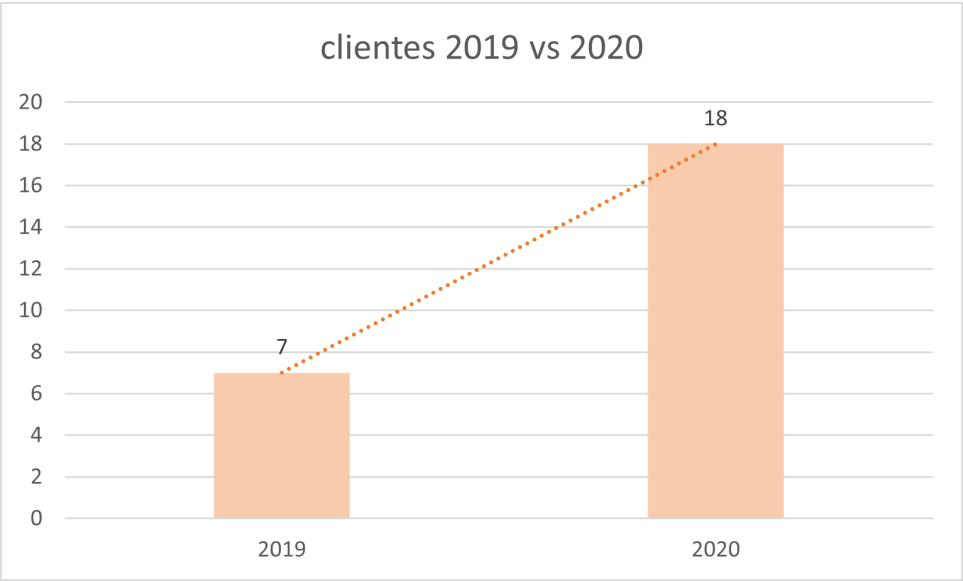

Figura 1.5: Crecimiento en cantidad de clientes de la empresa. Fuente: Elaboración propia.

Este acelerado crecimiento los ha llevado a aumentar el tamaño de la cartera de deudores que maneja en 400 veces lo manejado el año anterior, por lo que uno de los mayores desafíos para la empresa el 2020 fue mantener su operatividad al máximo y a la vez disminuir sus gastos. Para finales del año, lograron aumentar las gestiones realizadas a sus clientes en un 310 %, lo que logró que la empresa sobreviviera a este abrupto crecimiento y a su vez aumentando su productividad en todas las áreas gestionadas.

1.4. Empresa cliente

Para la realización de este proyecto, se utilizará la cartera de deudores de una de las empresas a las cuales se brindan los servicios de cobranza, la cual a su vez es su mayor cliente. La empresa principalmente se dedica al rubro del retail, vendiendo productos de diferentes categorías tales como tecnología y decoración, entre otras.

Este retail, como parte de su negocio, también brinda diversos servicios financieros, tales como el proporcionar una tarjeta de crédito, para la cual la empresa brinda sus servicios para gestionar los pagos de deudas de sus usuarios. [1]

Dentro de estos, se manejan diferentes carteras de clientes. Cada cliente asigna un día del mes el cual debe pagar su deuda. A este día se le conoce como **vencimiento**. Para dividir las carteras de clientes, se toman los días que han pasado sin que el cliente pague desde el vencimiento de su deuda hasta la fecha, más conocido en el rubro como la **mora** de un cliente. Así, se hace una distinción entre dos agencias o carteras de deudores, la primera con los clientes que tienen una mora menor o igual a 180 días, la cual a su vez se divide en deudores con mora menor a 30 días o **Flujo** (también conocido como tramo 0) y deudores con mora mayor a 30 o **Vigente**; y la otra agencia con clientes con mora mayor a 180 días, más conocida por ellos como **Castigo**.

1.5. Información del área de la empresa

El trabajo mencionado se centra en el área de producto, en especial en la subárea dedicada al data science. Dentro de producto, se busca diseñar y dar mantenimiento a la plataforma de cobranza. Esta plataforma está conformada por diferentes arquitecturas, ya sea front-end o el diseño disponible para los clientes, el back-end o la lógica interna de la plataforma o microservicios de gestión a clientes. Además, existen procesos analíticos para reportería dentro del software y también se desarrollan herramientas para uso interno de la compañía.

Con respecto al subárea, un área enfocada en el data science extrae valor desde la data recolectada, ya sea de manera estratégica o creando productos con base en la misma información, así generando nueva información [4]. Dentro de la empresa, esta área se centra en la creación de modelos y análisis de datos para la optimización de procesos, para la creación de estrategias tanto para la misma empresa como para los clientes de esta y creación de productos para integrar a la plataforma que la empresa proporciona a sus clientes.

[1] Para evitar confusiones, desde ahora se llamará empresa a la empresa con la cual se está realizando el proyecto y empresa cliente a la empresa que brinda los datos.

A junio de 2020, producto cuenta con 12 personas trabajando en el área, con dos de estos dedicados al área de data science. Uno de estos está dedicado a la creación de modelos y reportería, y el otro, el memorista, se encarga de la creación de modelos y herramientas para los clientes, y diversas tareas enfocadas al manejo de los datos para la extracción de información estratégica para el negocio. Ambos responden directamente hacia el CEO.

La principal contraparte para este proyecto es la misma área de producto. Debido a las problemáticas planteadas, existen dos posibles escenarios en donde el modelo afecta el rendimiento de la empresa: ya sea mediante soluciones orientadas a call center que den herramientas que ayuden a la toma de decisiones de los operadores a la hora de contactar clientes, o mediante herramientas de preprocesamiento de la carga diaria de deudores para optimizar tanto el call center como los contactos digitales. Cada uno de estos será analizado más adelante.

1.6. Predicción de pago de deuda: Justificación de la oportunidad

Debido a la pandemia ocurrida en el año de este trabajo, la digitalización de los servicios financieros, en especial la cobranza, se han visto en un gran crecimiento de demanda de diversas compañías debido a una reducción de costos o a la necesidad de implementación de trabajo remoto. Si bien este crecimiento conlleva beneficios a la compañía, también incentiva el ingreso de nuevos actores dentro del mercado. Así, la empresa pronostica que el mercado se pondrá cada vez más competitivo y se ve en la necesidad de encontrar herramientas para cerciorar su posición de líder en la industria. Frente a esto y gracias a su modelo de negocio enfocado al crecimiento y al uso de inteligencia artificial, la empresa encuentra una oportunidad de mejora en el servicio brindado al poder predecir la posibilidad de pago de un cliente ya que es una herramienta valiosa para sus clientes y a su vez ayuda a mejorar las gestiones hechas por ellos mismos. Con esto, se tiene el propósito de mejorar las métricas de rendimiento de los clientes que la empresa ya posee y a su vez mostrar una nueva herramienta para incentivar a nuevos clientes de utilizar la plataforma. Ejemplos de cómo la predicción de pago de deuda puede mejorar el rendimiento serán profundizadas en el marco teórico.

Por otra parte, la contingencia actual ha llevado a la compañía a tener dificultades en su flujo de caja debido a su estrategia de crecimiento y métodos de pago de sus clientes, por lo que como estrategia principal para abordar este periodo se ha buscado reducir los costos administrativos y de llamadas que tiene la empresa. Para esto, la empresa tiene dos posibles opciones: o disminuir el presupuesto asignado a uno de estos puntos u optimizar sus recursos existentes, obteniendo los mismos resultados a un menos costo. Bajo este contexto, la predicción de pago de deuda ha demostrado empíricamente reducciones significativas en gastos de operaciones de gestión de deudores al optimizar el contacto con cada cliente [5]. Esto refuerza la idea de que la implementación de un modelo de esta índole sería adecuada para ellos en la situación actual en la que se encuentran al tener diversas aplicaciones que pueden lograr tanto subir los estándares de calidad existentes como mantener los estándares actuales a un menor costo.

1.7. Por qué redes neuronales

Dentro de esta tesis se utilizarán redes neuronales para la resolución de dicha oportunidad. Los detalles más técnicos de este tipo de modelos serán ahondados más adelante, pero a modo general, se eligieron estos algoritmos para atacar este trabajo por diversas razones.

En primer lugar, Las redes neuronales poseen una gran ventaja con respecto a diversos programas lineales, la cual es su habilidad de procesamiento en paralelo. Esto permite a este tipo de algoritmos poder procesar gran cantidad de datos, en término de cantidad y variables involucradas, de una manera más optimizada con respecto a los demás algoritmos.

En segundo lugar, dentro de los algoritmos que poseen esta característica de procesamiento, en el caso planteado en este trabajo, las redes neuronales destacan de diversos otros modelos por ser generalizables. Esto en resumen nos habla de la capacidad de un algoritmo de redes neuronales de adaptarse y lograr optimizar cualquier función no lineal y de manera empírica esto nos permite por una parte cerciorarnos que el modelo será utilizable sin importar el método en el cual se quiera atacar la oportunidad y por otra parte les da la característica de adaptarse a datos desconocidos de mejor manera que otros modelos.

Esto último es imprescindible dentro del proyecto planteado, ya que dentro del proyecto en el cual se encuentra este trabajo (aunque fuera de los alcances de esta memoria) se busca que este modelo pueda ser replicable de manera fácil a más empresas dentro de la compañía, e incluso, en un escenario utópico, poder adaptarlo a un modelo que pueda trabajar con todo cliente de la compañía y pueda sufrir tan solo pequeñas reducciones en su rendimiento al utilizar data no conocida. Debido a esto, una red neuronal, optimizada con técnicas de normalización que serán descritas más adelante es la manera óptima para realizar este trabajo.

1.8. Hipótesis del trabajo

Para la realización del siguiente trabajo existen 3 supuestos principales en los cuales se basa tanto la eficacia como la aplicabilidad del modelo: La base de datos utilizada es capaz de inferir si un deudor pagará o no en base a las características que cada uno posee, los modelos planteados con redes neuronales presentarán una mejor métrica a definir que un modelo base, y los modelos presentados podrán ser generalizables a datos de otras empresas clientes, sin importar el rubro o país, entre otras características.

Con respecto a la base de datos, a modo general, si la red neuronal es capaz de inferir una función con los datos utilizados, esta presentará poder predictivo, lo que para este caso significa que puede realizar tareas de clasificación [6]. Así, se puede ver que a modo inicial si se debería poder inferir información, pero esto debe ser probado analizando los datos existentes. Además, para apoyar esta demostración de poder predictivo, también lo podremos ver con un modelo base.

Este modelo base nos ayudará a probar la siguiente hipótesis, al deber probar que nuestra red neuronal será un mejor modelo respecto a este modelo base. Para poder ver que la red

neuronal es mejor, debemos fijarnos ciertos criterios de comparación. En nuestro caso, se fijarán métricas por las cuales podremos tanto comparar estos dos modelos, como poder enunciar los resultados del modelo y poder explicar la aplicabilidad que estos tengan. Estas métricas son un punto clave a considerar para el trabajo total y se indagará en ellas más adelante.

Por último, una hipótesis imprescindible para el proyecto en general es que el trabajo realizado podrá ser generalizable al resto de las carteras de clientes. Por alcance del trabajo y por temas de confidencialidad de la compañía, esta hipótesis no será demostrada dentro del informe y quedará propuesta para un trabajo futuro.

1.9. Objetivos generales

El objetivo principal de este trabajo de título es "Diseñar, realizar y proponer aplicaciones de un modelo predictivo de pago para los deudores con mora menor a 30 días y con vencimiento en la primera quincena del mes de una empresa de retail financiero mediante algoritmos de clasificación utilizando redes neuronales".

1.10. Objetivos específicos

I. Entender el negocio en general y el problema a atacar en específico para poder realizar un trabajo enfocado en una solución utilizable.

II. Definir métricas de evaluación específicas para el problema, para así obtener un método de comparación y evaluación de los modelos acorde al negocio.

III. Definir e implementar un modelo base para definir un nivel estándar de las métricas definidas con los cuales se evaluarán las redes.

IV. Realizar y optimizar modelo con base en redes neuronales y evaluar resultados obtenidos para ver eficacia y rescatar puntos a mejorar.

V. Definir las aplicaciones e investigar insights del trabajo finalizado tanto para la empresa como para el cliente.

1.11. Alcances y resultados esperados

En términos de alcances de este trabajo, como se mencionó anteriormente, se buscará generar los diversos modelos y sus resultados, pero la implementación y aplicación en el negocio quedarán fuera de los alcances y quedarán propuestos para ser realizados en un futuro. Además, se buscará optimizar el resultado de un solo modelo basado en redes neuronales, ya que debido al nivel de complejidad bajo de este problema, utilizar topologías más avanzadas de redes neuronales sería menos eficiente que lograr optimizar de manera eficaz esta red que será planteada más adelante.

Dentro del trabajo se realizarán y evaluarán los modelos solo utilizando la data de los deudores tramo 0 o de **Flujo** con vencimientos en la primera quincena de la empresa cliente seleccionada, por lo que además quedará fuera de los alcances poder probar la data con otras carteras de deudores y con esto comprobar si el modelo es generalizable.

Con respecto a los resultados esperados, estos estarán directamente relacionados con cada uno de los objetivos específicos. Así, estos serán la métrica de evaluación para los modelos (objetivo II), el modelo base de predicción de pago de deuda (objetivo III), la configuración del modelo de redes neuronales creado con sus resultados dependiendo de la métrica de evaluación (objetivos IV) y las aplicaciones del trabajo realizado (objetivo V).

Capítulo 2

Marco Conceptual y Metodológico

2.1. Metodologías de mejora en la cobranza

Existen diversas herramientas que se utilizan para poder mejorar los procesos de cobranza en las empresas. Debido a que dentro de la compañía en la que se centra este trabajo las metodologías implementadas deben poder ser aplicables a todo rubro en el que exista cobranza masiva, un análisis de cada uno de estos permite encontrar factores en común que logren llegar a soluciones generalizables.

Viendo ejemplos en la industria de telecomunicaciones, un trabajo relevante muy aplicable a la empresa es el trabajo "Rediseño de proceso de cobranzas para una empresa de telecomunicaciones", de Nicole Flores, 2012. Aquí se proponen maneras de optimizar el proceso de cobranza mediante un rediseño de procesos, el cual integra una plataforma "middleware" para la gestión y análisis de las estrategias y campañas de cobranza que se realizarán, y a su vez se implementa una metodología simple y robusta para un cálculo de probabilidad de pago de los deudores. Esto, según la investigación, podría llegar a optimizar las gestiones desde un 21 % en un caso pesimista hasta un 77 % en un caso optimista mediante la disminución del gasto en cobranza, generando un ahorro en la deuda incobrable de hasta un 0,33 %. Este caso es relevante hacia el estudio por realizar en este trabajo debido a que, dentro de su rediseño, el software que ofrece la empresa donde se centra este trabajo califica como un "middleware" para las gestiones de cobranza y se pueden ver resultados en la utilización de la probabilidad de pago en términos numéricos y de manera simplificado, justificando dentro de esta industria la utilización de esto [5].

Dentro de la industria financiera, se puede ver el trabajo "Identificación de mejoras en el proceso de cobranza preventiva y temprana de productos de banca personal del Banco Interamericano de Finanzas" por Kenneth Figueroa, 2018. Aquí, se plantean tres diferentes propuestas de mejoras para optimizar el proceso de cobranza de la banca persona de este banco: Optimización del horario de gestión, utilización de estimaciones estadísticas para definir las estrategias de cobranza y optimización del número de registros enviados diariamente al sistema de administración de la cobranza. Con esto, se mejoró significativamente el porcentaje de contacto con los deudores. A priori, existe la posibilidad de realizar estos tres tipos de mejoras utilizando la probabilidad de pago como base en las decisiones estratégicas, por lo que queda por analizar dentro de las aplicabilidades del modelo estos tres puntos [7].

2.2. Modelos predictivos para el pago de deuda

Ya teniendo una noción de las metodologías utilizadas para las mejoras existentes en la cobranza, es necesario entender cómo se ha visto enfrentado el problema de poder predecir una probabilidad de pago de los clientes con deuda. Si bien se han nombrado anteriormente estudios que presentan implementaciones de modelos predictivos con este propósito, existen dos trabajos que nos ayudan tanto a fundamentar las metodologías propuestas a priori para la creación de modelos, así como también a visualizar el estado de arte en este campo.

2.2.1. Algoritmos de modelamiento y herramientas de optimización

Para tener diversos enfoques para abordar el problema tratado, se analizó el trabajo "A Mathematical Model for Predicting Debt Repayment: A Technical Note", por Udani Wijewardhana, Chinthaka Bandara y Thesath Nanayakkara, 2018 [8]. Dentro de este, se explican 3 principales metodologías para la predicción de pago.

En primer lugar, se genera un modelo de regresión logística para clasificar el pago de un cliente. Un modelo de regresión logística es una metodología de análisis estadístico multivariable con la cual se pueden predecir resultados, específicamente binarios, mediante la asociación de dependencias de esta variable con diversas variables de control [9]. En este trabajo, los resultados mostraban una correcta clasificación del pago de un 88 % de los deudores, pero el cual se veía influenciado por el desbalance de las bases en termino de cantidad de clientes que pagaron y no pagaron y su fuerte relación con solo una variable del modelo.

En segundo lugar, se analiza el uso de redes neuronales artificiales. Una red neuronal artificial se modela en base a la idea de simular el funcionamiento del cerebro humano, en específico las neuronas. Tiene la capacidad de modelar relaciones complejas no lineales al ser generalizable a cualquier función y nos ayudan a desencubrir las relaciones causales escondidas entre diversas características y variables [10]. Dentro de la aplicación estudiada, se aplicó para la clasificación del pago con diversas herramientas de preprocesamiento, como la estandarización de datos, el análisis de factores por componentes y la generación previa de clusters o segmentos en la data para añadir como variable en el modelo. Si bien no se detallaron estas metodologías de preprocesamiento en este informe, cabe destacar que cada una de estas será analizada al momento de la implementación de cada uno de los modelos. Los resultados encontrados fueron que la red neuronal por si sola logró clasificar al 10 % mayor de clientes de deuda con un 57,22 % de exactitud, mientras que cada una de las metodologías de preprocesamiento lograron una exactitud de análisis de un 62,9 %, 37,23 % y 64,4 %, respectivamente.

Por último, se realizó un modelo de análisis de afinidad. Este no se verá en más detalle debido a su poca aplicabilidad en los modelos, mostrando una clasificación de un 22 % en el 10 % mayor de clientes.

2.2.2. Estado del arte para la predicción de deuda: GBDT

Para definir un estado del arte en el cual se encuentra la recolección de deuda con base en la predicción de pago, nos basaremos en el trabajo "Data-Driven Consumer Debt Collection via Machine Learning and Approximate Dynamic Programming" [11]. Si bien el enfoque de dicho trabajo tiene un objetivo más específico a la predicción de pago en su forma más general (predecir día en el que sea más probable pagar), se rescata que se llegaron a resultados sobresaliente mediante la utilización de GBDT (y al uso de información histórica), por lo que se decide utilizar un modelo el cual utilice este algoritmo como modelo base con el cual se compararán las redes neuronales.

2.2.3. GBDT

GBDT o *Gradient Boosting Decision Tree* es un algoritmo de aprendizaje de máquinas el cual junta tres principales términos: Descenso por gradientes, *boosting* y árboles de decisiones.

El descenso por gradientes es una manera de optimizar una función objetivo $J(\theta)$ parametrizada por los parámetros de un modelo θ mediante la actualización de dichos parámetros con el uso de la dirección opuesta de la gradiente de la función objetivo $\nabla_\theta J(\theta)$ con respecto a sus parámetros, en pasos determinados por una tasa de aprendizaje η con la cual se busca llegar a un mínimo (local). En otras palabras, se busca utilizar las pendientes de la superficie creada por la función objetivo para descender hacia un óptimo local [12].

Por otra parte, *boosting* es un método para mejorar la exactitud de un algoritmo de aprendizaje mediante el ensamblaje de diversos modelos "débiles" para obtener un modelo más potente. Así, el *boosting* se basa en encontrar diversas reglas de aprendizaje mediante la creación de varios modelos base entrenados con diferentes subconjuntos de la data de entrenamiento (o siendo más específicos, una diferente distribución de datos en cada uno de estos subconjuntos), para luego combinar estas reglas en una única regla de predicción que será en teoría mas exacta que cada una por si sola.

Por último, un árbol de decisión para la clasificación consiste en la partición recursiva de una instancia en el espacio. Esto se genera mediante la creación de nodos conectados dirigidos desde una raíz hasta las hojas. En este árbol, cada nodo divide la instancia en el espacio en subespacios, por lo que en la raíz se puede encontrar la instancia completa o base de entrenamiento y en las hojas los subsegmentos finales. Cada nodo está asociado a un atributo, mientras que las hojas están asociadas con una clase de las variables objetivos. Así, se generan reglas de asociación para dividir la base de entrenamiento que, al recorrer el árbol, nos entrega una hoja final, la cual nos indica una clase predominante a la cual se le asigna como predicción. Se caracterizan por su simplicidad y usualmente son de rápido procesamiento [13].

Así, GBDT utiliza un método *boosting* en el cual utiliza arboles de decisiones como los modelos "débiles", con los cuales luego combina a un solo modelo mediante una optimización de la función objetivo utilizando descenso por gradientes. Utilizamos este para la realización de este trabajo debido a su gran eficacia y a su habilidad superior para encontrar interacciones no lineales automáticamente.

2.2.4. LightGBM

Para poder utilizar el algoritmo GBDT, se utilizará el framework LightGBM [14]. Las ventajas con otros frameworks con el cual se implementa este algoritmo son:

1. Entrenamiento más rápido y mayor eficiencia
2. Menor uso de memoria
3. Exactitudes similares (o mejores) que otros modelos GBDT
4. Óptimo para cantidades grandes de data

La razón de su eficacia es su algoritmo de crecimiento por hoja, el cual, a diferencia de los demás frameworks que crecen en profundidad, hace que el modelo converja con mayor rapidez.

2.3. Redes neuronales

En general, una red neuronal es un mecanismo de modelamiento particularmente útil para resolver problemas no lineares y descubrir las reglas de asociación aproximadas que describen la solución óptima de un problema. Son algoritmos que no siguen un orden específico al procesar datos, sino que utilizan estos mismos datos para encontrar las reglas que los gobiernan. Estos algoritmos son una caja negra, es decir, logran entregar una respuesta coherente y óptima al problema planteado, pero en los cuales no es directo el poder extraer el cómo se llegó a dicha solución. Cabe destacar que existen diversos tipos de estas que se diferencian tanto en la manera en que el modelo aprende de la información como en las maneras en las que la información y los cuales tienen diversas aplicaciones según los tipos de datos que se cuentan y el resultado que se espera, entre otros factores [15].

Ya entendiendo por qué se elige a priori la utilización de redes neuronales para este trabajo, es necesario entender cómo funciona una red neuronal. Una red neuronal artificial o ANN por sus siglas en inglés (*Artificial Neural Network*), es un modelo basado en las redes neuronales biológicas. Al igual que estas, la ANN es una interconexión de nodos, análogo a lo que son las neuronas en el cerebro. Cada red neuronal posee tres elementos claves: Características del nodo, topología de red y reglas de aprendizaje. La característica de nodo determina como un nodo interpreta cada señal, como el número de inputs y outputs asociados a este, el peso de cada uno y su función de activación, la topología de red determina cómo cada nodo está conectado y organizado entre sí, y las reglas de aprendizaje determinan como inician y se ajustan cada uno de los pesos. Cada uno de estos términos se explicará a continuación.

2.3.1. Características del nodo

En la figura 2.1 se puede ver un ejemplo de una red neuronal básica. Cada nodo es representado como una función de activación por el cual una serie de inputs, ponderados por un peso, son ingresados en la función de activación. Luego, si el valor de la suma producto

de los pesos y los inputs ingresados excede cierto límite, esta señal es activada y llevada a un siguiente nodo. Matemáticamente, esto puede ser expresado como:

$$y = f(\sum_{i=0}^{n} w_i x_i - T) \tag{2.1}$$

Caracterización de un nodo.

En donde y es el output de la función, f es la función de activación, w_i es el peso asociado al input x_i y T es el valor límite. Estos son los parámetros que definen las características de un nodo [10]. Si bien existen diversas funciones de activación, en este caso se indagará en las tres más utilizadas: las funciones sigmoides, tangente hiperbólica y ReLU.

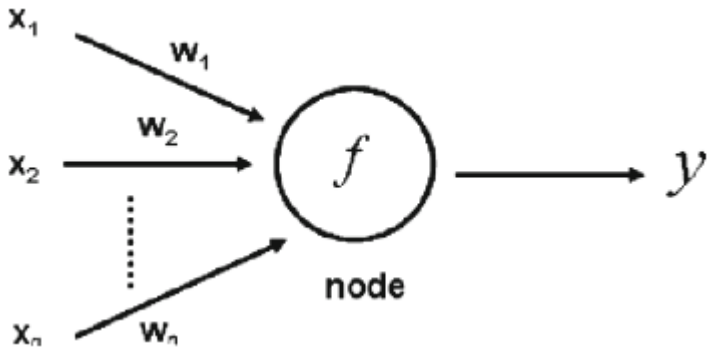

Figura 2.1: Ejemplo básico de red neuronal: $x = inputs, w = pesos, y = outputs$. Fuente: *Overview of artificial neural networks* [10].

2.3.1.1. Función sigmoide

La función sigmoide tiene la siguiente forma:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{2.2}$$

Así, si z toma un valor negativo, el valor de la función irá tendiendo a 0 a medida que el número es menor, y si el valor es positivo, el valor ira tendiendo a 1 a medida que aumenta, como se puede ver en la figura 2.2. Gracias a esto, la función sigmoide puede ser utilizado cuando en un nodo final se desean valores binarios, siendo muy útil para la clasificación [16].

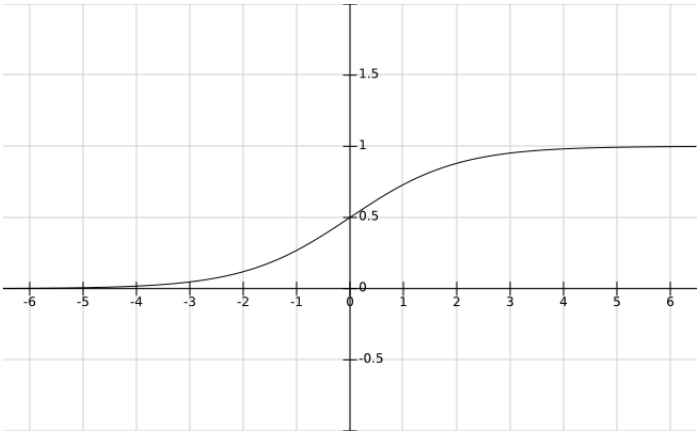

Figura 2.2: Función sigmoide. Fuente: Elaboración propia.

2.3.1.2. Función tangente hiperbólica

Similar a la función sigmoide, la función tangente hiperbólica presenta una forma de S, como se puede ver en la figura 2.3, y es representada por la función:

$$f(z) = tanh(z) \tag{2.3}$$

La diferencia más característica con la función sigmoide es que presenta valores entre -1 y 1 y está centrada en 0, lo que en algunos casos no lineales presenta mejores resultados que la función sigmoide [16].

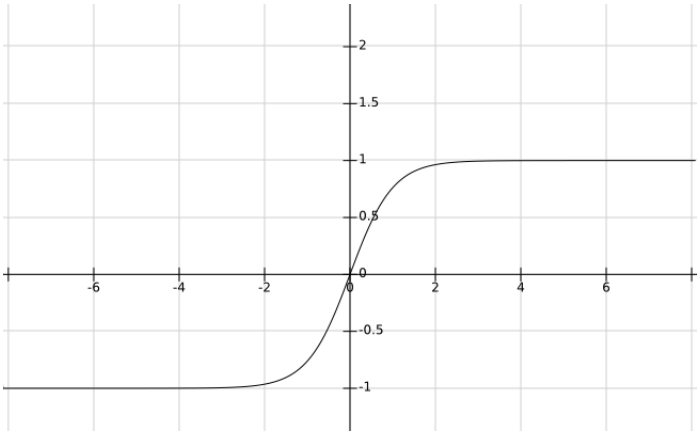

Figura 2.3: Función tangente hiperbólica. Fuente: Elaboración propia.

2.3.1.3. Función ReLU

La función ReLU o *Restricted Linear Unit* presenta la siguiente función:

$$f(z) = max(0, z) \tag{2.4}$$

Por lo que toma la forma mostrada en la figura 2.4, eliminando todo valor negativo dentro de la función [16].

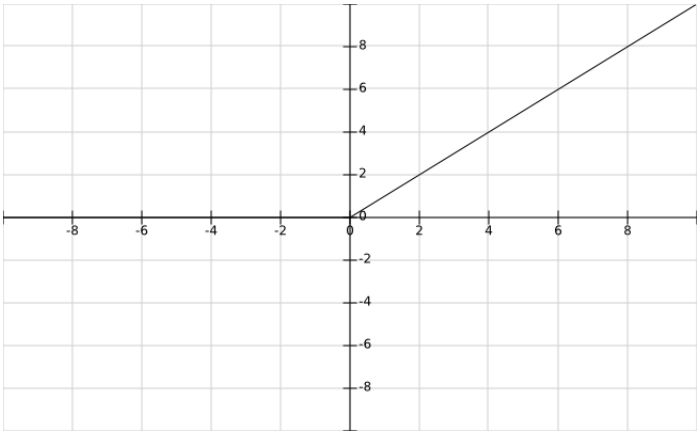

Figura 2.4: Función ReLU. Fuente: Elaboración propia.

Cabe destacar que estas son las principales funciones de activación, pero es necesario realizar una indagación más detallada sobre los diversos tipos existentes para llegar al mejor algoritmo posible.

2.3.2. Topología de red

La topología de red organiza la estructura por la cual los nodos se organizan dentro de la red neuronal. Los nodos se organizan en matrices lineales llamadas capas. A modo simplificado, existen tres tipos de capas: las capas input, las capas output y las capas ocultas. La capa input y output son una capa y las capas ocultas pueden ser o ninguna o un gran número, dependiendo de la arquitectura del modelo. La configuración de la topología de red implica determinar el número de capas en el modelo, el número de nodos en cada capa y las conexiones que existirán entre cada uno de los nodos. Usualmente, estos factores se ingresan de manera intuitiva y se van optimizando a medida que se experimenta con el modelo [10].

Con respecto a la conexión entre nodos, existen dos principales métodos de conexión los cuales implican 2 diferentes clasificaciones de redes neuronales: redes *feedforward* y redes *feedback*.

2.3.2.1. Redes feedforward

Las redes neuronales *feedforward* solo tienen conexiones con neuronas en una capa posterior, por lo que no hay iteraciones dentro del flujo en sí y son estáticas, es decir, un input al modelo presenta un único output [16]. Un ejemplo de esto se puede ver en la figura 2.5.

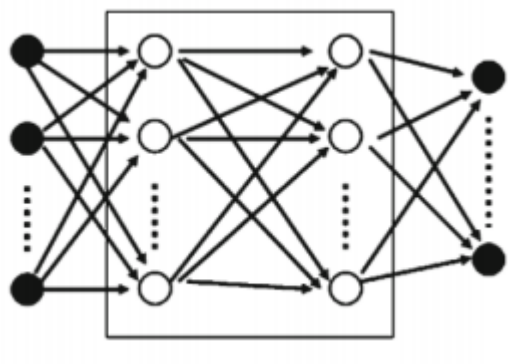

Figura 2.5: Ejemplo simple de red *feedforward* con 4 capas. Fuente: *Fundamentals of Deep Learning* [16].

2.3.2.2. Redes feedback

Por otro lado, las redes neuronales *feedback* presentan iteraciones y son dinámicas, es decir, para un input, el estado de la red neuronal cambia durante varios ciclos hasta llegar a un punto de equilibrio, por lo que puede resultar en una serie de outputs [16]. Un ejemplo de esta arquitectura se puede ver en la figura 2.6

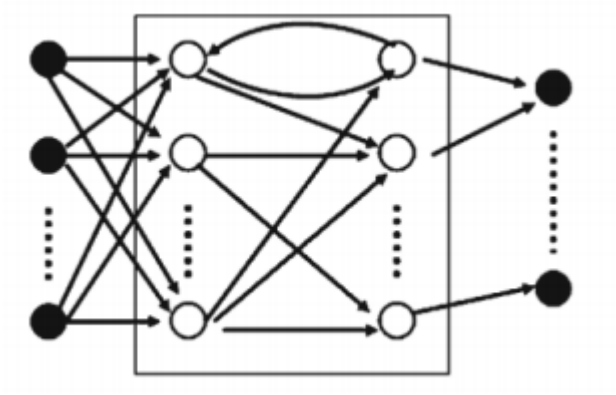

Figura 2.6: Ejemplo simple de red *feedback*. Fuente: *Fundamentals of Deep Learning* [16].

2.3.3. Reglas de aprendizaje

Para entrenar la red, es necesario definir un proceso de aprendizaje. En este, los pesos involucrados en la red son ajustados hasta lograr un valor deseado. Este aprendizaje se puede clasificar en dos categorías, aprendizaje supervisado y no supervisado, los cual difieren porque, entre otras cosas, en el aprendizaje supervisado existe un set de entrenamiento que posee los outputs esperados, mientras que en el no supervisado no se tienen claro a priori los outputs. En el caso de este trabajo, el tipo de aprendizaje será supervisado [10].

Además de definir este proceso, es necesario entender como se hace que la red neuronal aprenda. Para el caso de la clasificación, se calcula un error en base a la relación entre los valores reales de la clasificación y los valores predichos por el modelo. Estos valores predichos

se calculan con los inputs que se entregan y los pesos que se asignan a cada neurona. Luego, mediante algún algoritmo de optimización, estos pesos se actualizan iterativamente para llegar a un valor mínimo [16]. A continuación se presentan los algoritmos de optimización más famosos.

2.3.3.1. SGD

SGD o *Stochastic Gradient Descent* se basa en el descenso por gradientes explicado anteriormente. La diferencia con este es que el descenso por gradiente estocástico recibe solo un input en cada iteración para actualizar los pesos, por lo que el requerimiento computacional disminuye considerablemente en intercambio por un mayor número de iteraciones necesarias para la convergencia. Es el algoritmo más conocido para optimización [16].

2.3.3.2. RMSProp

RMSProp o *Root Mean Square Propagation* es otro método de optimización que simplifica el proceso mediante la utilización del promedio de magnitudes de las iteraciones y el tamaño de las gradientes anteriores para disminuir el tamaño de la gradiente actual, lo que ayuda al procesamiento del algoritmo para tasas de aprendizajes pequeñas y mantiene las magnitudes de las gradientes en un ratio siempre similar, haciéndolo robusto y útil para procesamiento de datos en lotes (lo que convenientemente ocurre en redes neuronales) [12].

2.3.3.3. Adam

Adam o *Adaptive Moment Estimation* utiliza una metodología similar a RMSProp pero lo combina con la utilización del momento, lo que en simples palabras significa que genera una combinación lineal entre el gradiente y el incremento anterior, y considera los gradientes recientemente aparecidos en las actualizaciones para mantener diferentes tasas de aprendizaje por variable. Suele ser considerado el estado del arte para problemas con un set de datos grande, pero esto no significa que siempre será el mejor algoritmo a utilizar y es necesario probar con cada uno de estos para cerciorar un modelo óptimo [12].

2.3.4. Generalización de redes neuronales

Si bien lo anterior da una noción de lo esencial para poder generar un modelo basado en redes neuronales, dada la problemática presentada en este trabajo, un punto imprescindible al considerar es la propiedad de la red neuronal de poder ser generalizable a sets de datos fuera del entrenamiento. Para poder reforzar esta facultad, existen diversas técnicas que se utilizan al momento de la creación de un modelo. Para este trabajo se utilizarán tres y serán explicadas a continuación.

2.3.4.1. Dropouts

Un *dropout* es un proceso en el cual se desactivan ciertas neuronas de manera aleatoria en una capa de una red neuronal para así disminuir el sobreajuste de los datos. La diferencia con solo eliminar estas neuronas es que el dropout elimina aleatoriamente distintas neuronas en cada uno de los ciclos de ajuste de pesos, por lo que toda neurona es utilizada, pero con un nivel de ajuste menor. La proporción de neuronas desactivadas por ciclo se regula mediante el parámetro *dropout_rate*.

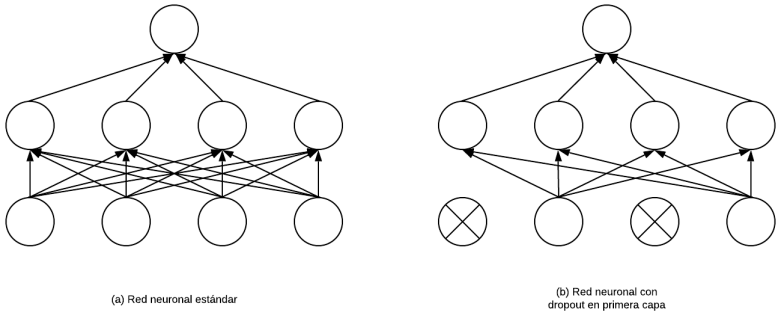

Figura 2.7: Ejemplo de uso de dropout con un dropout_rate de 0,5. Fuente: Elaboración propia.

2.3.4.2. Regularización de capas

La regularización de capas es un proceso en el cual se agrega una penalización dentro de los parámetros de la capa para reducir el número de parámetros dentro del modelo, lo cual ayuda a que el modelo se ajuste menos a los datos utilizados en el entrenamiento y encuentre reglas más generales para encontrar las clasificaciones, así ayudando a la generalización total del modelo [17]. A modo de simplicidad, si definimos una capa de una red neuronal como

$$y = Wx + b \tag{2.5}$$

con y el output de la capa, W la matriz de pesos de la capa, x los inputs y b un valor constante conocido como *bias*, se puede generar 3 tipos de regularizaciones en esta función, los cuales pueden funcionar en paralelo:

1. **Regularización de kernel**: se aplica una penalización en el kernel, es decir, trata de reducir los valores de la matriz W.
2. **Regularización de bias**: Se aplica una penalización en el sesgo, reduciendo el valor de b.
3. **Regularización de actividad**: Se trata de reducir de manera general el resultado de una capa, es decir, aplica una penalización para reducir el valor de y.

La lógica detrás de la regularización es agregar el término correspondiente a la regularización implementada a la función de pérdida. Así, al momento de buscar un mínimo para esta en la optimización tratará de disminuir los valores que afecten menos al valor sin penalización de la función de perdida. Para incluirlas dentro de esta, se considerarán tres técnicas:

1. **Regularización Lasso**: Más conocida como regularización L1. Se implementa al agregar la suma de los absolutos a la función de pérdida. Debido a esto, suele impactar fuertemente los parámetros, inhabilitando completamente parámetros irrelevantes al modelo y pudiendo llevar a matrices dispersas [18]. Dentro de la implementación en redes neuronales, esta penalización va acompañada por un hiperparámetro $l1$ para suavizar su efecto en la función de pérdida, el cual debe calibrarse.

2. **Regularización Ridge**: También conocida como regularización L2. Se implementa al agregar la suma del cuadrado de los parámetros a la función de pérdida. Al aplicar el cuadrado en vez del valor absoluto, esta regularización logra una penalización menor a los parámetros [18]. Similar a L1, va acompañada de un hiperparámetro $l2$ en la función de pérdida.
3. **Regularización L1 L2**: Aplica ambas regularizaciones mencionadas anteriormente al mismo tiempo.

2.3.4.3. Normalización de lotes

La normalización de lotes consiste en normalizar las activaciones en las capas intermedias de la red neuronal. Esto lo hace realizando una transformación del output de cada capa, acercando su media a 0 y su desviación estándar a 1. Así, cada *epoch*, o iteración de entrenamiento, presenta un peso inicial de las neuronas similar, por lo que en la práctica el modelo debería poder extraer información general del modelo y no presentar diferencias significativas cuando se enfrenta a datos anómalos al resto. Debido a esto, un modelo que utiliza normalización de lotes debería presentar un menor ajuste a los datos de entrenamiento y a su vez ajustarse mejor a datos desconocidos, ayudando a la generalización de la red neuronal [19].

2.3.5. Ejecución de redes neuronales: Tensorflow y Keras

Para ejecutar estos algoritmos de redes neuronales existen diversos lenguajes de programación que permiten el aprendizaje de estos modelos. Por requerimiento de la empresa, el lenguaje utilizado en este trabajo es python, y para su realización se utilizarán las librerías *Tensorflow* y *Keras*.

Tensorflow es un algoritmo para expresar algoritmos de aprendizaje de máquinas y para a su vez implementar su ejecución. Cuenta con un ecosistema integral y flexible de herramientas, bibliotecas y recursos de la comunidad que permite que los investigadores innoven con el aprendizaje automático y los desarrolladores creen e implementen aplicaciones con estas tecnologías fácilmente [20].

Una de estas tecnologías es *Keras*. *Keras* es una librería de aprendizaje profundo que funciona encima de *Tensorflow*. Fue creado con un enfoque en poder realizar experimentos de manera rápida y eficiente. Su principal ventaja es ser una herramienta que simplifica *Tensorflow* de cara al desarrollador mientras mantiene una flexibilidad para poder pasar de un desarrollo simple a uno más avanzado, siendo a su vez eficiente y escalable, logrando un rendimiento al nivel de la industria y siendo utilizado por empresas tales como *Youtube* y *Nasa* [17].

2.4. Métricas de evaluación

Las métricas de evaluación son el mecanismo por el cual se podrá comparar y mostrar resultados de los modelos realizados a lo largo de este trabajo. A priori, se definirán estas métricas para tener una noción general de cómo se evaluarán los modelos, pero es necesario saber que existen otros factores que se deberán tener en cuenta al momento de elegir un

método de evaluación de rendimiento del modelo, como lo son los montos recaudados con la predicción, por ejemplo.

2.4.1. Matriz de confusión

La matriz de confusión es una tabla que representa los valores esperados y predichos de la clasificación. Se utiliza esta tabla para ver el funcionamiento del modelo y si entrego las respuestas esperadas y para la creación de diversas métricas relevantes dependiendo del problema planteado [21]. Se puede ver su estructura en la figura 2.8.

	Positivo (Predicho)	**Negativo (Predicho)**
Positivo (Actual)	Verdadero Positivo	Falso Negativo
Negativo (Actual)	Falso Positivo	Verdadero Negativo

Figura 2.8: Estructura de matriz de confusión. Fuente: elaboración propia.

En esta tabla, se cuenta la cantidad de cada uno de estos casos:

- **Verdadero Positivo (TP)**: Predicción es **positiva** y el valor real es **positivo**.
- **Falso Positivo (FP)**: Predicción es **positiva** y valor real es **negativo**.
- **Verdadero Negativo (TN)**: Predicción es **negativa** y valor real es **negativo**
- **Falso Negativo (FN)**: Predicción es **negativa** y valor real es **positivo**

2.4.2. Precisión y Recall

Estas dos métricas nacen de la matriz de confusión y nos dan un insight de los casos positivos correctamente clasificados. La Precisión, también conocida como exactitud, se define como

$$Precision = \frac{TP}{TP + FP} \tag{2.6}$$

Así, la precisión expresa cuántos de los casos realmente positivos, fueron predichos como positivos. Por otra parte, la exhaustividad, también conocida como *Recall* se define como

$$Recall = \frac{TP}{TP + FN} \tag{2.7}$$

Por lo que la exhaustividad expresa la cantidad de casos clasificados positivos que fueron correctamente clasificados. El darle énfasis a una de estas métricas dependerá del problema a tratar [15].

2.4.3. Exactitud

También conocida como *Accuracy*. La exactitud se define como la cercanía del modelo a dar los valores reales de la data, y se expresa como

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.8}$$

Esta medida ayuda a dar nociones generales del modelo, pero presenta problemas con el desbalance de datos, ya que, por ejemplo, si los casos positivos fueran la clase dominante y el modelo nos diera solo predicciones positivas, la exactitud podría arrojar valores deseables [15].

2.4.4. F1

El score F1 es una función creada a partir de la precisión y la exhaustividad. Se define como

$$F1 = 2\frac{Precision * Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN} \tag{2.9}$$

Esta métrica es altamente utilizada como métrica de evaluación para modelos de clasificación, ya que con esta métrica se pueden evaluar de igual forma los casos positivos con los negativos, por lo que no sufre el mismo problema de la exactitud con el desbalance de datos. Mientras el valor de esta esté más cercano a 1, el modelo presentará mejores predicciones [15]. A priori, dentro de los modelos esta métrica será la base por la cual se definirá cómo estos se evaluarán.

2.5. Metodología

2.5.1. Proceso CRISP-DM

En la investigación "KDD, SEMMA and CRISP-DM: a parallel overview", Ana Azevedo y M.F. Santos, 2008 [22], se brindan tres diferentes maneras para abordar un proceso de extracción de conocimiento en base a la información. Para este trabajo, se utilizará un proceso CRISP-DM o *CRoss-Industry Standard Process for Data Mining*, al ser bastante comparable con los otros dos procesos y a su vez integrar una etapa de entendimiento del negocio, la cual es de vital importancia para la implementación del trabajo a realizar. Este cuenta con 6 pasos interactivos e iterativos, siendo cíclico y completo:

1. **Entendimiento del negocio** - En este paso inicial se busca obtener un entendimiento completo del negocio y de los objetivos que este busca con el proyecto. Busca entender el problema desde una perspectiva del negocio para así encontrar la mejor tarea de minería de datos para la realización de este. Con respecto a los objetivos específicos, se buscará en esta etapa poder definir las mejores métricas para la evaluación y tener un análisis empresarial de la data que se posee.
2. **Entendimiento de los datos** - En este paso, se buscará entender a cabalidad la estructura de datos y posibles variables que se pueden utilizar para la realización del

modelo. Se realizará un análisis exploratorio de los datos y se evaluarán diversas variables auxiliares que podrían ser utilizadas en el modelo.

3. **Preparación de los datos** - Esta fase cubre todas las actividades que ayuden a construir el data set final de datos. Aquí se utilizarán y analizarán diversas metodologías de preprocesamiento como análisis de factores, estandarización de datos y segmentación previa, entre otras.
4. **Modelamiento** - En esta fase, se seleccionarán y aplicarán diferentes técnicas de modelamiento y se buscará optimizar los parámetros de cada una de estas. Para esto, se utilizará una metodología que llamaremos "panda", que será descrita más adelante.
5. **Evaluación** – En esta etapa se evalúan los resultados de los diversos modelos y se analizan los pasos seguidos para poder cerciorar la credibilidad del modelo.
6. **Aplicaciones** - Si bien en un modelo CRISP-DM tradicional en esta etapa se busca la implementación del modelo en el negocio, dentro de este trabajo se buscará en esta etapa encontrar diversas aplicaciones que tendrá el modelo y una mirada general de cómo implementarlas. Así, con esta fase se completará el punto V de los objetivos específicos.

Como se mencionó anteriormente, este modelo es cíclico e iterativo, por lo que, dentro de su implementación, cualquier descubrimiento traído en cualquiera de sus fases puede llevar a un cambio que conlleve en revisitar otras de las fases, lo que lo hace efectivo y orientado a cumplir los objetivos de la manera más eficiente. Se realizarán iteraciones capaces de generar un modelo base y un modelo a comparar con esta metodología.

2.5.2. Metodología de optimización de modelo: PANDA

Con CRISP-DM tenemos una metodología para abarcar el trabajo a modo general, pero una vez entrenando los modelos planteados, existen dos enfoques para la optimización de estos: Panda y Caviar.

El primero de estos se basa en el hecho que la mayoría de los pandas adultos no tienen más de un crío a la vez para enfocarse en cuidar a un bebe a la vez. Traducido al entrenamiento de algoritmos, esta metodología se basa en enfocarse en entrenar un modelo a la vez, realizando cambios sucesivos a este mismo modelo para poder llegar hacia uno óptimo. Esta metodología es buena cuando no se tiene suficiente mano de obra para poder realizar varios modelos o no se tiene el poder de procesamiento en la compañía para realizar aprendizaje en paralelo.

Por otra parte, Caviar se refiere a la forma en que estos huevos de pescados son engendrados de a muchos al mismo tiempo, por lo que formula una metodología que busca un entrenamiento en paralelo de varios modelos para luego elegir el modelo con un aprendizaje óptimo. Esta metodología es útil cuando se tiene un ambiente de programación complejo y bien distribuido, con casos que utilizan cantidades de datos en el orden de Petabytes de información.

Debido a la índole del proyecto y a los recursos disponibles, se plantea una metodología panda para la optimización de los modelos por su optimización exhaustiva y su buena aplicabilidad con CRISP-DM.

Capítulo 3

Desarrollo del Modelo

Para realizar el modelo se tomará tan solo un pago mensual por cliente como índice de realización de pago. Así, se inicia con una base de 1.886.291 datos y 19 columnas. Tras eliminar nulos, se obtiene una base de 1.885.936 datos y con información de 484.327 clientes diferentes, la cual será la base inicial con la que se trabajará. La descripción de cada una de estas variables esta detallada en el anexo B.1.

La tabla 3.1 entrega un detalle por periodo de los datos.

Tabla 3.1: Resumen de datos por periodo y pagos correspondientes.

Periodo	**Total de datos**	**Pagos**	**Porcentaje pago**
202001	405.389	104.284	25,72 %
202002	388.684	170.207	43,79 %
202003	309.353	156.358	50,54 %
202004	299.774	160.433	53,52 %
202005	234.551	121.658	51,87 %
202006	248.185	113.402	45,69 %
Total	**1.885.936**	**826.342**	**43,82 %**

Fuente: Elaboración propia.

3.1. Análisis empresarial de los datos

3.1.1. Definición de base a utilizar

En primer lugar, se decide utilizar para este trabajo los datos de la empresa de retail seleccionada debido a la cantidad de datos que se poseen y además a la buena estructura que estos tienen. Una de las razones por lo que es un punto importante el poder generalizar los modelos es porque cada compañía tiene diferente formato de datos y con esto también diferente calidad en la recolección de estos. El partir con una base de datos con una buena estructura inicial ayudará a evaluar la eficacia del proyecto en general de mejor manera, o por lo menos dará un algoritmo útil para el mayor cliente de la empresa.

Por otra parte, es necesario entender por qué se decide utilizar los clientes de la cartera

de Flujo (definido en la base de datos por la variable *mora*) y los vencimientos de la primera quincena del mes (definido en la base de datos por la variable *vencimiento*). Como se mencionó anteriormente, la empresa retail divide sus deudores en clientes Flujo, Vigente y Castigo. Esto debido a la gran diferencia en los comportamientos de los deudores. Haciendo un análisis lógico, al momento de tener una deuda, el común de las personas suele sentir distinto el demorarse en pagar una nueva deuda dentro del mes que debe pagarla y el mes siguiente de esta, por lo que su comportamiento también suele cambiar. De manera más gráfica, el gráfico 3.1 muestra los rendimientos en la colección de deuda durante la primera mitad del año 2020 diferenciado por el tipo de cartera y con una división para los clientes Flujo dependiendo de su vencimiento.

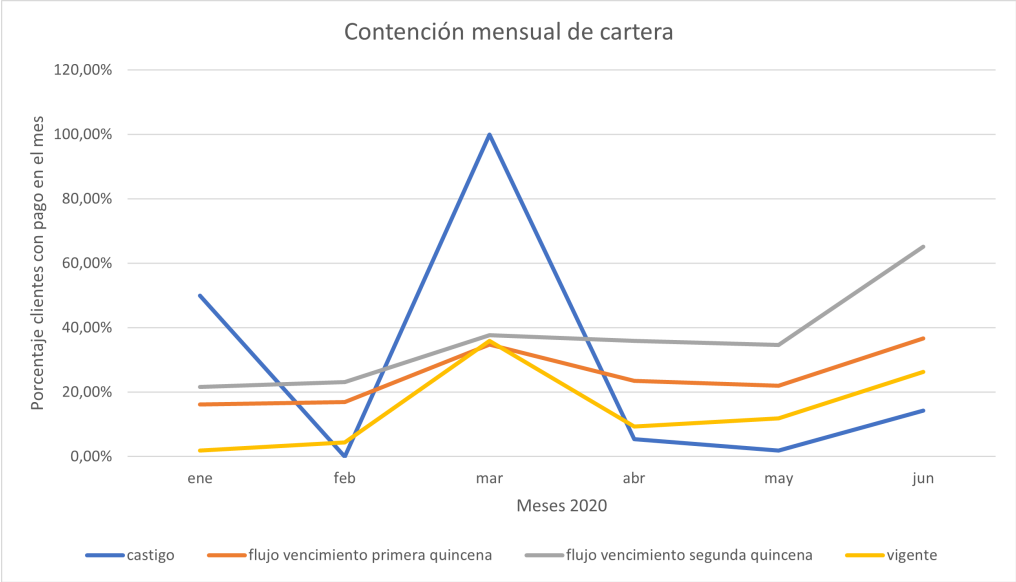

Figura 3.1: Porcentaje mensual de clientes que realizaron algún pago, dividido por carteras y por quincena de vencimiento para cartera flujo. Fuente: Elaboración propia.

Se puede notar una diferencia significativa entre cada cartera. Con respecto a la cartera de flujo, se puede ver que las tendencias no difieren fuertemente entre los vencimientos en la primera quincena y los que están en la segunda, pero se puede apreciar que las magnitudes varían entre si y que a medida que avanza el año es más fuerte la diferencia entre estos.

En términos de los vencimientos, por confidencialidad se tuvo que tomar una decisión para poder reducir el tamaño de la base de datos. Se tomó la decisión de tomar solo la primera quincena de vencimientos ya que al tener diferentes comportamientos los clientes, si la predicción se hace solo a estos deudores no se suele perder mucha información, ya que se puede ver que las tendencias son similares, aunque es algo que se debe analizar en trabajos futuros. Además, ya que la empresa trabaja por metas de cantidad recuperada, el tener una predicción específica para los recuperos que deben realizarse la primera quincena ayuda a plantearse las estrategias que se deberían tomar la segunda quincena para poder llegar a estas metas planteadas.

3.1.2. Insights del negocio en la data

Tras un entendimiento de los datos con una mira del negocio, existen varios puntos en los datos que son interesantes a analizar y ayudarán tanto a generar una limpieza para la base final como a generar nuevas columnas para nuestro modelo mediante ingeniería de variables.

En primer lugar, se puede apreciar en la tabla 3.2 que existen datos negativos dentro de los días de mora, el cupo disponible y la deuda facturada de cada cliente, lo cual va en contra de la lógica usual que tendrían estas variables. Los días morosos negativos se refieren a clientes que aún no tienen deuda, pero que se acerca su fecha de vencimiento, por lo que son casos a los cuales se les realizan gestiones de manera preventiva. En el caso de los cupos disponibles y la deuda facturada negativos, estos dentro de la lógica de negocio se toman como errores, por lo que no se utiliza como información para gestionar clientes y no aportaría información al modelo. Además, dentro de estos últimos, y en todas las variables correspondientes a montos, se suelen ver diversos outliers en los montos alto debido a que hay una gran variación en los montos, pero los cuales son imprescindibles al momento de generar el modelo ya que estos son los clientes con mayor importancia al momento de recuperar sus deudas.

3.1.3. Métricas de evaluación

Dado el análisis de negocios presentado, se deciden las métricas que se utilizarán. En primer lugar, se utilizarán como métricas para la evaluación de los modelos la exactitud (*accuracy* en la tabla), ROCAUC y F1 para tener una visión completa sobre el rendimiento del modelo y pueda ser fácilmente comparable con otros modelos similares. Luego de esto, se definieron métricas acordes a tanto el negocio general como con el proyecto. Estas se especifican en el anexo B.2.

3.1.4. Análisis de datos

Dentro de la tabla 3.1, se puede observar claramente una diferencia en la distribución de pagos en el mes de enero. Si bien esto solo ocurre de manera drástica para este mes, esto da un indicio de cómo las estacionalidades pueden ser un factor relevante al momento de realizar este modelo y lo cual debe ser indagado si se quiere llegar a un trabajo que pueda ser mejorado en un futuro con nueva data.

A modo general, la tabla 3.2 muestra los valores iniciales dentro de las variables numéricas utilizadas en el modelo.

En muchos de los casos se pueden ver valores igual a 0. En variables como *saldo total* y *por vencer* no hace sentido tener estos valores, y el hecho de que aproximadamente un 80 % de la data posee estos valores da un indicio de que no es una variable de confianza para utilizar. En términos al *total vencido* y *cuota más morosa*, aproximadamente un 80 % de los valores iguales a 0 corresponden a clientes en cartera preventiva (mora menor a 0) por lo que hace sentido que no tengan un monto asignado en este caso. Por último, con respecto al *total de deuda*, existen valores iguales a 1, pero estos corresponden a menos del 0,1 % de la data.

Tabla 3.2: Análisis descriptivo de data inicial.

	mean	min	max
cupo_disponible	34494,12	-4752453	3448828
mora	1,53	-10	31
saldo_total	10856,67	0	4862661
total_vencido	11492,27	0	4868294
por_vencer	377097,19	0	8952192
total_deuda	448069,64	1	13141748
promesas_cumplidas	0,39	0	203
promesas_rotas	1,02	0	209
monto_cuota_mas_morosa	11447,4	0	4868294
deuda_facturada	72464,72	-65592	4868294
edad	43,14	0	116
antiguedad_mora	194,44	86	313

Fuente: Elaboración propia.

Viendo variables demográficas y las demás variables categóricas, se puede apreciar errores claros en la *edad*, al presentar valores iguales a 0 o en general menores a 18 años. Además, el gráfico 3.2 muestra diversos outliers debidos a esta variable.

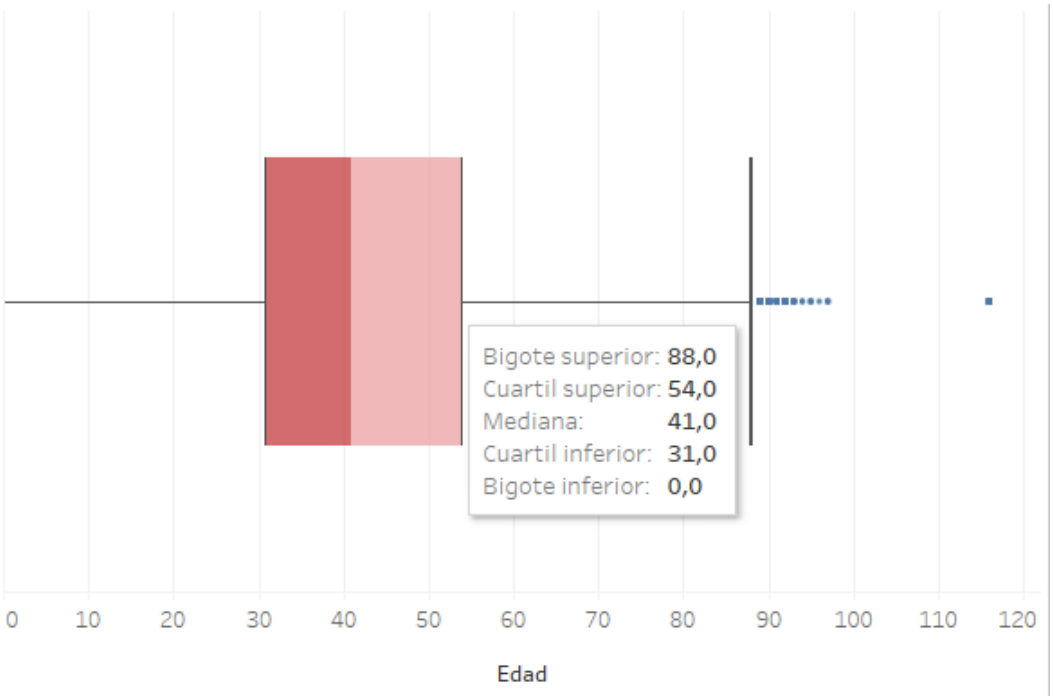

Figura 3.2: Gráfico de caja y bigotes de distribución de edades. Fuente: Elaboración propia.

Con respecto a *vencimiento* (la cual se tomará como variable categórica), se puede apreciar en la siguiente figura la distribución de estos dependiendo del pago.

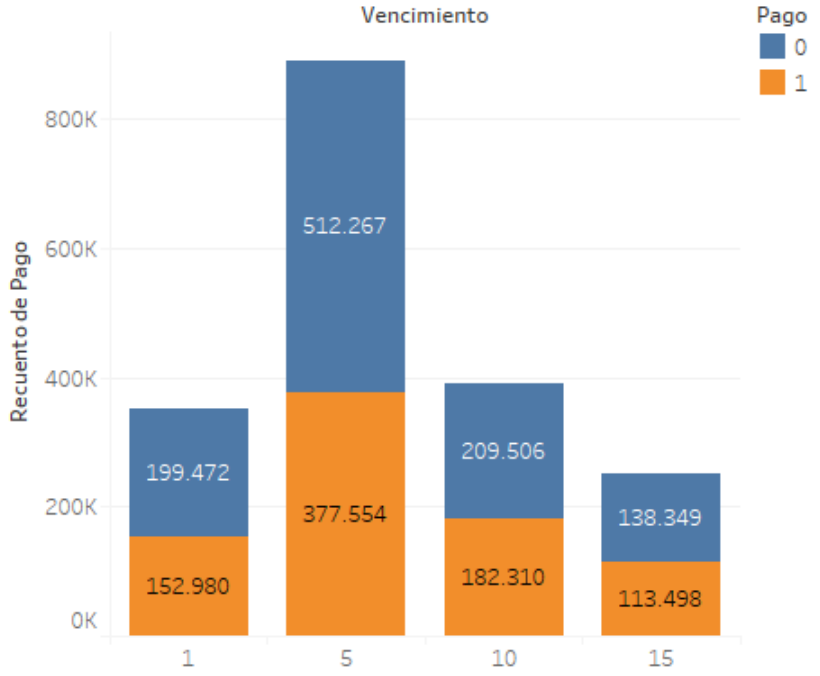

Figura 3.3: Cantidad de pagos por vencimiento. Fuente: Elaboración propia.

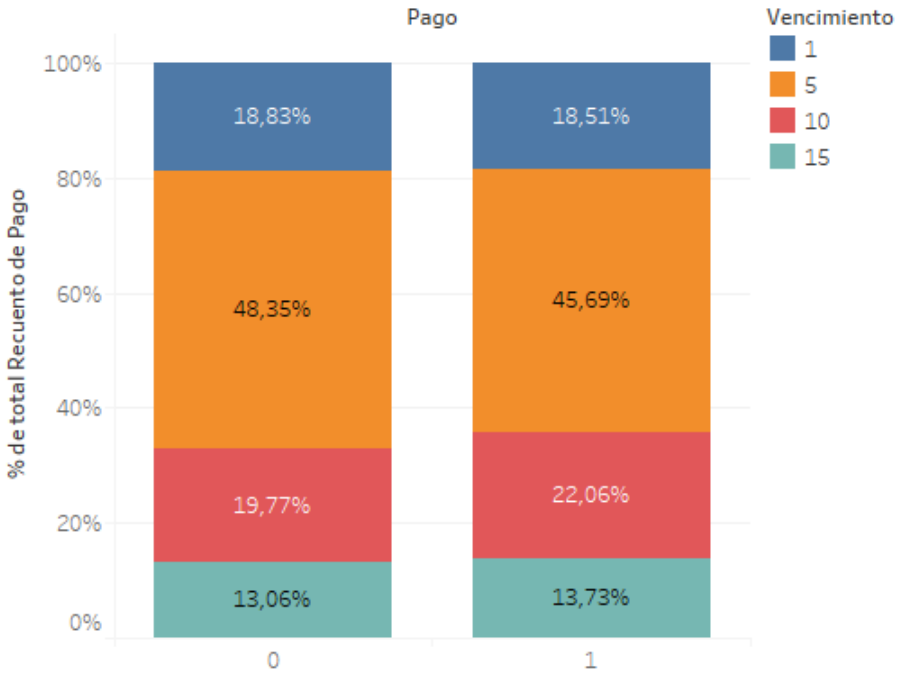

Figura 3.4: Distribución de valores de vencimiento según pago. Fuente: Elaboración propia.

En la variable sexo, las siguientes gráficas muestran la distribución de datos dependiendo del sexo y el pago.

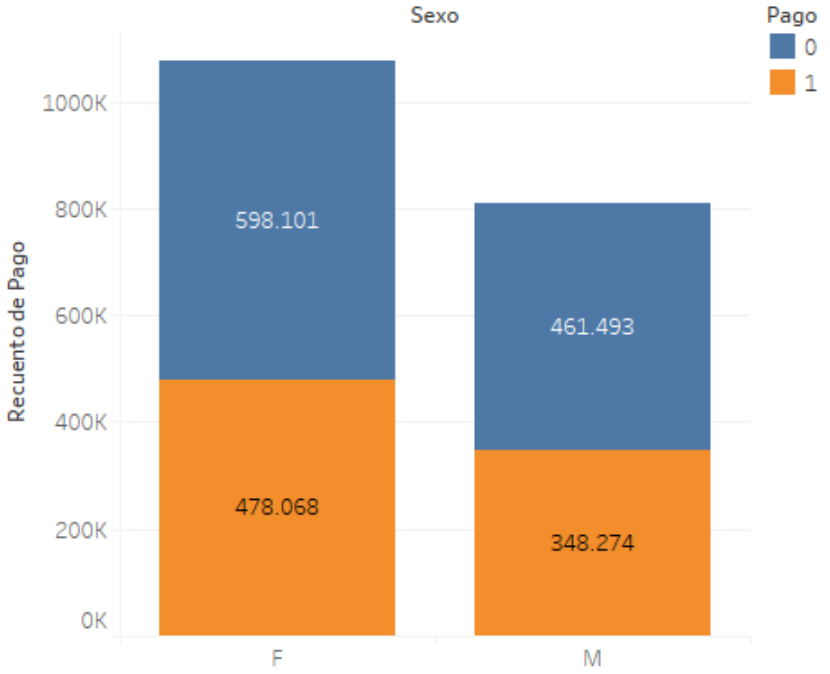

Figura 3.5: Cantidad de pagos según sexo. Fuente: Elaboración propia.

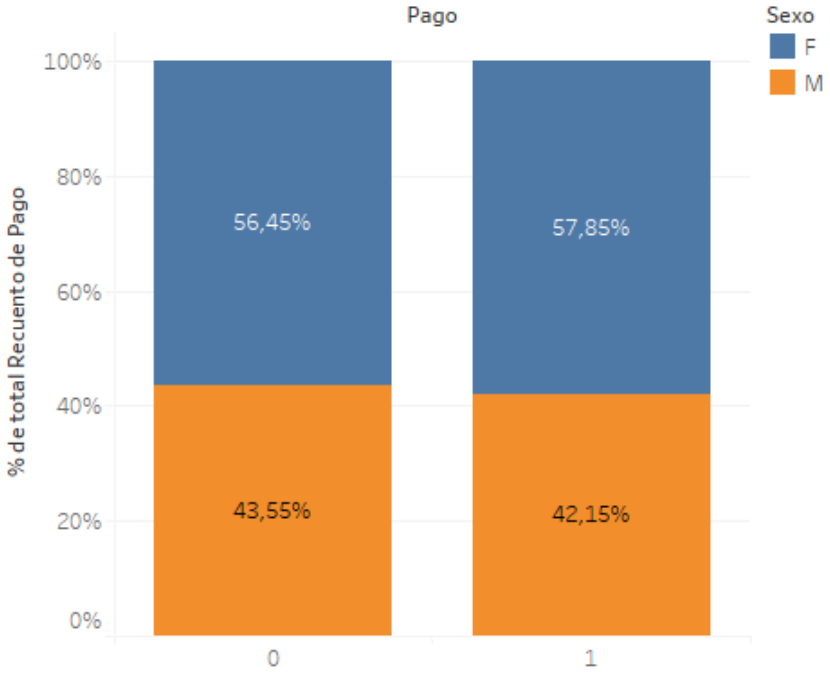

Figura 3.6: Distribución de valores de sexo según pago. Fuente: Elaboración propia.

Con respecto a las clasificaciones, solo cabe destacar que solo 11 de los 41 valores con mayor representatividad abarcan un 99 % de los datos, por lo que se deberá trabajar solo con estas.

Por último, para analizar la importancia de las variables, se realiza un análisis de correlaciones entre las variables del modelo y la variable objetivo de pago.

Tabla 3.3: Análisis de correlación entre variables independientes y variable pago.

Variable	Correlación
total_deuda	0,41 %
deuda_facturada	1,13 %
promesas_cumplidas	1,37 %
por_vencer	1,75 %
vencimiento	2,33 %
promesas_rotas	3,87 %
edad	4,70 %
cupo_disponible	9,29 %
preventivo	17,04 %
saldo_total	17,41 %
total_vencido	17,46 %
monto_cuota_mas_morosa	17,51 %
mora	23,74 %
antiguedad_mora	24,47 %
pago	100,00 %

Fuente: Elaboración propia.

3.1.5. Preprocesamiento de los datos

Considerando lo dicho anteriormente, se tomaron decisiones en base a la base para llegar a datos que optimizaran la inferencia de información para la realización del modelo. Dentro de estas decisiones, es necesario en primer lugar explicar las normalizaciones y estandarizaciones de datos realizadas para luego mostrar los demás cambios que llevaron a la base final a utilizar.

3.1.5.1. Estandarización de los datos

El primer trabajo de estandarización realizado se tuvo que hacer debido a la decisión de permitir los casos outliers de los valores de monto por las razones mencionadas anteriormente. Para que el sesgo que esto conlleva no afecte a las métricas de rendimiento y se pueda disminuir la varianza en los valores de estos montos, se aplicó la función logarítmica exponencial para llevar a los valores a una distribución más cercana a una distribución normal. La figura 3.7 da un ejemplo de este cambio en las distribuciones sesgadas

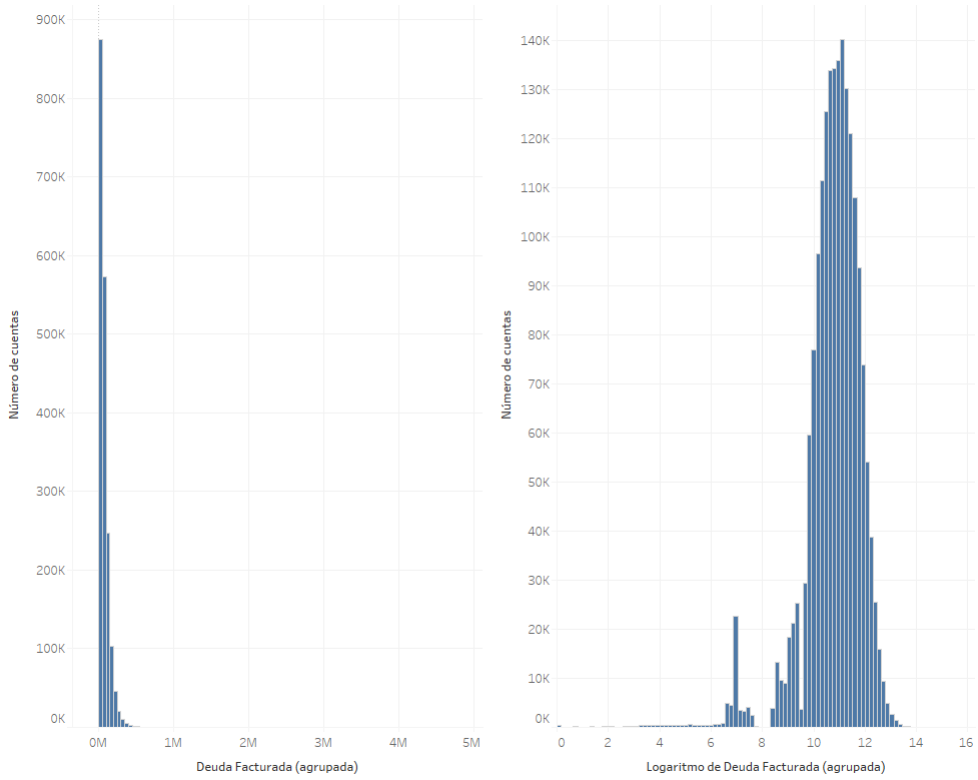

Figura 3.7: Distribución de Deuda Facturada vs distribución de logaritmo exponencial de Deuda Facturada. Fuente: Elaboración propia.

Luego de esto, debido a la naturaleza de los modelos utilizados, para tener resultados no sesgados por el tamaño de las variables, se realiza una estandarización general de los datos. En este caso, se realiza una estandarización *minmax*, la cual distribuye los valores de cada variable entre 0 y 1.

3.1.5.2. Base final

Tras el análisis y la transformación de valores previamente planteados se deciden hacer diferentes cambios:

- Eliminación de datos: Se eliminaron los datos con valores de *total de deuda* igual a uno, los valores de *cupo disponible* iguales a 0 y los valores de *edad* menores a 18 y mayores a 89. En términos de las *clasificaciones*, se eliminaron datos correspondientes a 30 de las 41 clasificaciones, dejando solo las 11 con mayor representatividad.
- Eliminación de variables: Se eliminó la columna de datos *saldo total* al presentar diversos errores y no aportar información relevante la modelo.
- Ingeniería de variables: Se creo la variable *mes* desde el periodo para trabajar en la estacionalidad. Cabe destacar que esta puede o no tener un impacto significativo en el modelo actualmente, pero su valor será visto en un trabajo futuro al tener más datos de diferentes meses para el modelo. Por otra parte, se creó la variable *preventivo*, cuyo valor es 1 si el deudor tiene una mora menor a cero y 0 si no. Además de esto, se generaron valores binarios en la base para cada uno de los valores categóricos en la base, de las variables *vencimiento*, *mes*, *sexo* y *clasificación*. Cabe destacar además que hubo un

trabajo previo en el cual se creó la variable *antigüedad mora* desde el valor en fecha de cuando se inició la deuda y la variable *edad* desde la fecha de nacimiento de los deudores.

Las siguientes tablas muestra nuevamente el resumen de datos por periodo y probabilidad de pago y un análisis final de las variables.

Tabla 3.4: Análisis descriptivo de data final previa estandarización *minmax*.

	mean	**min**	**max**
cupo_disponible	220298,52	1	3448828
mora	0,15	-10	31
total_vencido	7448,59	0	1849746
por_vencer	204940,38	0	6961673
total_deuda	254126,86	2	6969605
promesas_cumplidas	0,31	0	186
promesas_rotas	0,74	0	198
monto_cuota_mas_morosa	7412,74	0	1849746
deuda_facturada	67530,77	-18587	2576961
edad	44,32	19	88
antiguedad_mora	190,60	86	313
mes	3,22	1	6

Fuente: Elaboración propia.

Tabla 3.5: Resumen final de datos por periodo y pagos correspondientes.

Periodo	**Total de datos**	**Pagos**	**Porcentaje pago**
202001	360.071	157.697	43,80 %
202002	346.694	149.488	43,12 %
202003	291.459	148.107	50,82 %
202004	268.888	114.751	42,68 %
202005	217.437	96.351	44,31 %
202006	229.987	105.256	45,77 %
Total	**1.714.536**	**771.650**	**45,01 %**

Fuente: Elaboración propia.

Es interesante notar que la gran diferencia en el porcentaje del mes de enero se vio solucionada por la limpieza de los datos, por lo que se descarta que esta haya sido por estacionalidad.

Más allá de esto, no se realizan otras técnicas de preprocesamiento, ya que la mayoría de estas, como el análisis de componentes principales, requiere una interpretación lineal de los datos, lo que debido a la naturaleza no lineal de un modelo de redes neuronales lleva a una mala interpretación de los modelos [23]. Cualquier otra técnica de preprocesamiento a implementar queda propuesta para un trabajo futuro.

Tras el trabajo previamente explicado, se llega a una base final que cuenta con 1.859.538

filas y 32 columnas, con 1 de estas siendo nuestra variable objetivo. Esta se dividió en data de entrenamiento, correspondiente al 90 % de los datos de enero a mayo, data de validación correspondiente al 10 % restante y una data de testeo correspondiente al mes de junio. Además, se crearon bases correspondientes al top 10 % de clientes con mayor deuda facturada para el set de validación y de testeo.

3.2. Modelamiento

3.2.1. Modelo base

Tras reiteradas iteraciones del modelo utilizando la metodología establecida, se llega a un modelo base mediante el algoritmo LIGHTGBM. La tabla 3.6 muestra los hiperparámetros elegidos para este modelo, los cuales fueron seleccionados tras aproximadamente 50 iteraciones automáticas buscando maximizar el valor de la métrica F1 para luego ajustar manualmente los valores para llegar al modelo óptimo.

Tabla 3.6: Hiperparámetros utilizados para el modelo base.

Hiperparámetro	Valor utilizado
learning_rate	0,2
boosting_type	gbdt
sub_feature	0,75
num_leaves	867
min_data_in_leaf	40302
bagging_fraction	0,74
bagging_freq	9
max_depth	192

Fuente: Elaboración propia. Para el detalle de estos refiérase a la librería del modelo [14].

Los resultados del modelo se discutirán en el capítulo 4.

3.2.2. Modelo de redes neuronales

3.2.2.1. Modelo inicial

Al ser la creación de un algoritmo de redes neuronales un proceso más manual, no es directo el encontrar como partir con la estructura que esta tendrá. La gran cantidad de hiperparámetros y las diversas topologías que la red puede tomar pueden resultar desafiante para cualquier proyecto con base en este modelo.

Para sobrellevar la tarea de encontrar la mejor configuración para la red, se creo un algoritmo el cual puede generar redes neuronales, buscando encontrar la configuración que pudiera maximizar cierta métrica objetivo. Tras varias pruebas con diversas métricas, se encontró que la mejor manera de optimizar los parámetros del modelo se basa en buscar la configuración que maximice la métrica AUC de la data de validación.

Parámetros fijos

En primer lugar, los inputs quedan fijados por la cantidad de columnas que posee nuestra base de datos, 32. Además de esto, es necesario tener una capa final de 1 neurona, con una función de activación sigmoide, ya que esta nos dará un valor continuo entre 0 y 1 para poder clasificar entre pago y no pago. Para que ocurra esto, la función de perdida a utilizar deberá ser *binary_crossentropy* al estar optimizando una variable objetivo binaria.

Topología

Dentro del mundo de las redes neuronales, el problema abordado se puede considerar como un problema sencillo de abordar, por lo que se proponen 3 topologías basadas en redes *feed forward*:

- **Topología constante**: Topología que contiene un número constante de neuronas, independiente del número de capas. Para ajustar la cantidad de neuronas por capa se crea el parámetro *neurons* y para controlar la cantidad de capas se crea el parámetro *hidden_layers*.

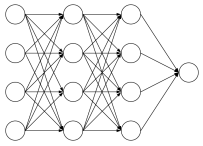

Figura 3.8: Ejemplo red constante. Fuente: Elaboración propia.

- **Topología de embudo**: Con esta topología, se intenta partir expandiendo las características de los datos al partir con un número de neuronas alto para luego ir reduciéndolo hasta llegar a la capa final con una neurona. Para ajustar la tasa en la cual la cantidad de neuronas se ve reducida por capa, se crea además el parámetro *neuron_rate* y se inicializa el proceso con una cantidad de neuronas calculada desde la cantidad de neuronas *neurons* y la cantidad de capas *hidden_layers*.

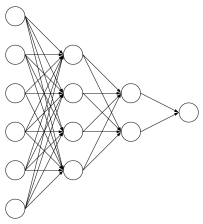

Figura 3.9: Ejemplo red embudo. Fuente: Elaboración propia.

- **Topología exponencial**: Esta última topología es similar a la topología de embudo pero inicializa mediante la apertura de la red al aumentar exponencialmente la cantidad de neuronas para luego generar el embudo en la segunda mitad de la red. Nuevamente el crecimiento y la disminución de neuronas por capa se ve suavizado por el parámetro *neuron_rate*.

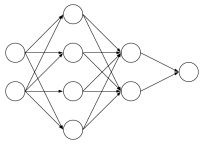

Figura 3.10: Ejemplo red exponencial. Fuente: Elaboración propia.

Hiperparámetros

Además de la topología, se optimizaron diversos hiperparámetros correspondientes a redes neuronales y específicos a la librería Keras [17] y Tensorflow [20]:

- **Epochs**: El valor de *epoch* define la cantidad de ciclos los cuales la red utiliza para entrenar distintos lotes de datos para llegar a los pesos que minimicen la pérdida del modelo.
- **Batches**: *batches* o lotes es la cantidad de datos que se entrega a la red en cada uno de sus ciclos o *epochs*.
- **Optimizer**: *Optimizer* nos indica que optimizador será utilizado en el modelo. En este caso se varió entre *SGD*, *RMSProp* y *Adam*, descritos previamente en la sección 2.3.3
- **Learning_rate**: La tasa de aprendizaje o *Learning_rate* es un parámetro que suaviza el trabajo del optimizador para encontrar un mínimo local más eficientemente. Dentro de la afinación de Hiperparámetros, se toma un valor base de esta variable, ya que se utiliza una metodología de actualización de la tasa en cada ciclo de la red, por lo que los valores tomados por la tasa de aprendizaje se ven realmente afectados por el ciclo en el que se encuentre la red y el *decay*.
- **Decay**: Como se mencionó anteriormente, el *decay* o decaimiento actualiza el valor de la tasa de aprendizaje en cada uno de los ciclos. Esto se hace tanto a nivel de valor de la tasa como dentro de cada uno de los optimizadores.
- **Momentum**: El *momentum* es una métrica similar a la tasa de aprendizaje. La función de esta es atenuar la oscilación del descenso en gradientes en caso que esta sea muy pequeña o muy grande.
- **Activation_f**: *Activation_f* indica la función de activación que tendrá una capa, explicado en la sección 2.3.1. Si bien este no es un parámetro general del modelo, a modo de simplicidad en el algoritmo se elige una función de activación general para todas las capas del modelo (exceptuando la última).

En el anexo A.1 se puede encontrar la función utilizada para la afinación de parámetros y creación de red de manera automática. Tras aproximadamente 200 iteraciones de red se llegó a la mejor red de esta función, con un AUC de 70,78 % y un valor F1 de 69,09 %. Sin embargo, en la data de prueba estos valores se vieron considerablemente disminuidos, con un AUC de 51,64 % y un F1 de 27,92 %. Esta discrepancia en las métricas de evaluación entre ambos sets de datos significa un gran sobreajuste en el modelo, por lo que el siguiente paso en el modelamiento es tratar este sobreajuste y ayudar a la generalización de la red.

3.2.2.2. Dropouts, regularizaciones y normalización de lotes

Para trabajar el sobreajuste se utilizaron las técnicas ya mencionadas en la sección 2.3.4. Estas se fueron integrando progresivamente en orden de impacto en un modelo, por lo que se inició con la prueba de *dropouts*, luego con las regularizaciones y finalmente con la normalización de lotes.

El proceso para agregar los *dropouts* fue similar a la creación del modelo inicial, con un algoritmo que iterara entre modelos con *dropouts* en diversas capas y con distinto *dropout_rate* en cada una. A diferencia del modelo inicial, en esta ocasión se buscó el modelo que maximizara el valor de F1 de la data de testeo. Al partir con un modelo ajustado a la data de validación, ahora se puede ajustar a la data de testeo ya que no presentará un infrajuste a los datos, y se utiliza ahora la métrica F1 al ser la más afectada por el sobreajuste.

Después de encontrar la mejor configuración de *dropouts*, se procedió a probar con diversas regularizaciones en las capas de la red, explicadas en la sección 2.3.4.2. Se probó aproximadamente con 30 configuraciones distintas que combinaban los distintos tipos de regularizaciones en los distintos niveles de la función de la capa, buscando nuevamente optimizar el F1 en el set de testeo. En este punto al estar ya cerca de métricas óptimas en el set de testeo, se buscó también manualmente priorizar las configuraciones que afectaran menos al set de validación.

Por último, se realizó un proceso de normalización de baches al inicio de la red neuronal como última medida contra el sobreajuste y para ayudar a la generalización del modelo, generando una mejora en las métricas tanto de monto del modelo como las variables F1 y AUC de la data de prueba, llegando al modelo final.

3.2.2.3. Modelo final

Tras el proceso mencionado anteriormente y diversos ajustes manuales, se logra llegar a un modelo de redes neuronales que compite fuertemente con el modelo base. La figura 3.11 y la tabla 3.7 muestran un resumen de la estructura de la red.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dropout_1 (Dropout)          (None, 25)                0
_________________________________________________________________
batch_normalization_1 (Batch (None, 25)                100
_________________________________________________________________
dense (Dense)                (None, 96)                2496
_________________________________________________________________
dense_1 (Dense)              (None, 576)               55872
_________________________________________________________________
dense_2 (Dense)              (None, 503)               290231
_________________________________________________________________
dense_3 (Dense)              (None, 383)               193032
_________________________________________________________________
dense_4 (Dense)              (None, 254)               97536
_________________________________________________________________
dense_5 (Dense)              (None, 147)               37485
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 148
=================================================================
Total params: 676,900
Trainable params: 676,850
Non-trainable params: 50
_________________________________________________________________
```

Figura 3.11: Summary() de la red neuronal final. Fuente: Elaboración propia utilizando Keras.

Tabla 3.7: Carácterísticas de las capas.

Capa	Neuronas	Función de activación	Regularizador	Dropout
Normalización	-	-	-	-
1	96	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0,2
2	576	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0
3	503	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0
4	383	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0
5	254	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0
6	147	ReLU	Regularización de sesgo L2 (ratio = 1e-6)	0
7	1	Sigmoide	-	0

Fuente: Elaboración propia.

El resto de los hiperparámetros elegidos se pueden ver en la tabla 3.8.

Tabla 3.8: Carácterísticas de las capas.

Hiperparámetro	valor
learning_rate	0,1
batches	512
epochs	46
neurons	96
momentum	0,65
decay	0,08
neuron_rate	0,14
optimizer	RMSProp
hidden_layers	5
activation_f	ReLU
topology	Embudo

Fuente: Elaboración propia.

Como se puede apreciar, la profundidad de la capa se puede considerar relativamente mediana con solo 7 capas, pero cada una de estas posee una gran cantidad de neuronas, lo que justifica el sobreajuste de los datos. Otro punto importante es que al incrementar drásticamente el número de neuronas en cada capa hace sentido que el *dropout* se incluya al inicio del modelo al disminuir la mayor cantidad la información que se entrega. En la siguiente sección se muestran los resultados del desarrollo.

Capítulo 4

Análisis de Resultados y Aplicabilidad

4.1. Resultados por modelo

Tras el trabajo previamente descrito, se comparan los resultados de los modelos desarrollados mediante las métricas de evaluación elegidas para esto. En la tabla 4.1 se muestran los resultados del modelo base para ambos sets de datos y para el top 10 % de clientes con mayor deuda facturada.

Tabla 4.1: Métricas de evaluación para modelo base en los distintos set de datos.

	LIGHTGBM			
	Validación	**Top 10 % Validación**	**Test**	**Top 10 % Test**
Corte para clasificación (Threshold)	0,58	0,58	0,58	0,58
Cantidad de datos	148.454	14.845	229.987	22.998
Exactitud	72,58 %	74,73 %	59,29 %	59,82 %
F1	64,93 %	63,92 %	60,82 %	68,94 %
AUC	71,09 %	71,60 %	60,05 %	58,72 %
Monto recuperado	$ 2.777.041.988	$ 395.191.723	$ 5.353.788.459	$ 1.673.520.633
Monto recuperado (%)	25,10 %	11,97 %	40,42 %	38,71 %
Monto recuperado en pagos (%)	57,11 %	30,51 %	78,90 %	77,09 %
Recuperos perdidos	11.836	1.160	61.029	7.529
Monto recuperos perdidos	$ 887.969.750	$ 602.192.654	$ 4.218.413.982	$ 1.672.327.821
Monto recuperos perdidos (%)	8,03 %	18,24 %	31,85 %	38,68 %
Número gastos innecesarios	28.865	2.592	32.597	1.712
Pagos predichos	37.679	3.324	72.659	10.257
No pagos predichos	70.074	7.769	63.702	3.500

Fuente: Elaboración propia.

El modelo presenta resultados significativos, pero no excepcionales, junto con un sobreajuste a la data incluso después de las diversas optimizaciones realizadas a este. En primer lugar, viendo las métricas tradicionales de evaluación, se puede apreciar una diferencia en la exactitud del modelo entre el set de validación y el de testeo significativa, lo cual se repite en la métrica AUC. Por otra parte, para la métrica F1 no se observan diferencias considerables entre ambos sets. Los valores de estas tres métricas se pueden considerar aceptables al ser mayores que una elección aleatoria de clasificación (en la cual estas métricas se encontrarían cercanas al 50 %), pero pese a esto existe un gran espacio de mejora para el modelo.

En segundo lugar, observando las métricas de recupero, se puede apreciar que dentro del monto recuperado es donde existe el mayor atractivo en este modelo. La métrica se ajusta precariamente al set de validación, pudiendo predecir el recupero de solo un 25,1 % del monto total y un 57,11 % del monto que fue verdaderamente recuperado. Sin embargo, en la data de testeo se puede apreciar que estos porcentajes suben en un 15 % y un 20 % aproximadamente, dando un monto recuperado drásticamente mayor (esta diferencia es aún mayor para los sets de top 10 %). Pese a esto, observando las métricas de recupero perdido, esta diferencia porcentual se ve nuevamente reflejada, pero en contra del set de testeo, con una diferencia porcentual de más de 20 % entre los sets de datos. La explicación lógica de estas métricas es que el modelo está dando como predicción valores más cercanos a una clasificación positiva (es más propenso a decir que un cliente pagará su deuda). Esto está creando una inflación en las métricas de evaluación y está disminuyendo la usabilidad del modelo, ya que, en el caso extremo, si un modelo sólo entrega un valor de clasificación para toda la data, este no está verdaderamente aportando información. Este caso extremo está lejos de lo que realmente sucede con el modelo, pero es un punto de mejora significativo a considerar en este algoritmo.

Para el resto de las métricas, no se ve una diferencia entre el número de gastos innecesario entre los sets de datos. Al ver los pagos y no pagos predichos, se puede concluir que esta diferencia viene dada por la misma tendencia de clasificación a pago del modelo, presentando una disminución en los no pagos similar al incremento en los pagos de las predicciones.

Para los sets pertenecientes al top 10 % de monto de deuda, lo más importante a destacar es que se ve una disminución porcentual significativa en el monto recuperado. Este también es un espacio de mejora relevante ya que el ajustarse bien a estos datos en particular aumenta considerablemente el rendimiento monetario del modelo.

En el modelo basado en redes neuronales, la tabla 4.2 muestra los valores de sus métricas.

Tabla 4.2: Métricas de evaluación para modelo final en los distintos set de datos.

	RED NEURONAL			
	Validación	**Top 10 % Validación**	**Test**	**Top 10 % Test**
Corte para clasificación (Threshold)	0,5	0,5	0,5	0,5
Cantidad de datos	148.454	14.845	229.987	22.998
Exactitud	56,10 %	65,74 %	54,35 %	60,20 %
F1	64,59 %	66,32 %	64,20 %	71,18 %
AUC	59,21 %	68,93 %	57,09 %	58,74 %
Monto recuperado	$4.285.363.908	$804.519.568	$6.232.636.410	$1.865.391.324
Monto recuperado (%)	38,74 %	24,37 %	47,05 %	43,15 %
Monto recuperado en pagos (%)	88,13 %	62,12 %	91,85 %	85,93 %
Recuperos perdidos	58.084	4.177	93.869	8.485
Monto recuperos perdidos	$3.799.710.446	$1.228.659.156	$5.081.485.262	$1.846.410.312
Monto recuperos perdidos (%)	34,35 %	37,22 %	38,36 %	42,71 %
Número gastos innecesarios	7.094	909	11.110	668
Pagos predichos	59.450	5.007	94.146	11.301
No pagos predichos	23.826	4.752	30.862	2.544

Fuente: Elaboración propia.

En este modelo se ven métricas las cuales tienen bastante espacio de mejora, pero con

un gran ajuste entre la data de validación y la de testeo. Los valores de las métricas de evaluación exactitud, F1 y AUC son de 56,5 %, 64,59 % y 59,21 % respectivamente, por lo que el poder predictivo del modelo es pequeñamente mejor a un modelo aleatorio. Sin embargo, se puede rescatar que la data se ajusta mucho mejor al top 10 % de datos en la validación, lo cual es un punto a favor para el modelo. Además, no se ve una diferencia significativa entre los valores en el set de validación y de testeo, por lo que se puede decir que se logró la tarea de generalizar la data, pero a un costo considerable en el poder del modelo. A pesar de esto, si queremos comparar los resultados con la literatura similar, podemos ver que el modelo realizado tuvo resultados parcialmente mejores a los obtenidos por Udani, Chinthaka y Thesath en su trabajo comparando solamente los modelos sin preprocesamientos, obteniendo una exactitud de 60,2 % en el top 10 % de testeo superando el 57,22 % de la literatura mencionada [8].

Con respecto a los montos, se pueden apreciar buenos resultados al poder predecir exitosamente un gran porcentaje del monto que se recuperará, con un 88,13 % en la data de validación y un 91,85 % en la data de testeo. Pese a esto, se puede justificar esto con el gran porcentaje de monto de recupero perdido. Esto lleva a pensar que el modelo tiene una tendencia a dar predicciones cercanas a 1, por lo que se ve fuertemente sesgado el poder predictivo del modelo. Además, en los sets del top 10 % se ve disminuido este recupero considerablemente, con una diferencia aproximada de un 26 % y un 16 % en los tops de validación y de testeo respectivamente.

Otro punto positivo del modelo es la pequeña cantidad de predicciones clasificadas gasto innecesario predichas por el modelo en todos los sets de datos con respecto al total de datos con los que se trabaja. Esto, aunque puede ser una oportunidad de aplicación para el modelo también se ve justificado por la gran cantidad de predicciones con tendencia a un pago realizadas.

4.2. Comparación de modelos

Como se mencionó anteriormente, ambos modelos cuentan con ciertas virtudes y falencias. El modelo base tiene un ajuste a la data de validación bastante mayor que las redes neuronales, pero el sobreajuste que este implica es alto por lo que realizar una implementación preliminar con este no presentaría resultados significativamente mejor que el modelo final. Por otra parte, este último presenta un poder de generalización bastante mayor al modelo base, pero para lograr esto sus métricas de evaluación en general se ven fuertemente disminuidas. A pesar de esto, cabe destacar que el modelo de redes neuronales inicial previo a los ajustes para mejorar el rendimiento en el set de testeo presentaba métricas bastante similares al modelo base, pero el sobreajuste que este conllevaba era significativamente mayor, por lo que existen indicios de que la red neuronal puede llegar al mismo poder predictivo del modelo LIGHTGBM si se encuentran nuevas maneras de atacar el sobreajuste.

4.2.1. Análisis monetario: Simulación

Al construir un set de dato de testeo correspondiente al siguiente mes de la data utilizada, el observar los resultados que este el modelo obtiene con esta información es equivalente a implementar el modelo hecho en el mes de junio de 2020 y evaluar su rendimiento. Cabe

destacar que este análisis se ve sesgado por el hecho de que no se tomaron decisiones de negocio con el modelo, por lo que no da una visión exacta del rendimiento de este. A pesar de esto, nos permite comparar de manera monetaria los modelos realizados con un enfoque de negocio, lo cual es necesario para poder justificar el uso de modelo.

Para esto, en la tabla 4.3 se muestran las ganancias correspondientes a la deuda facturada recuperada, las perdidas correspondientes al recupero perdido y los gastos que estos conllevaron debido al costo de las gestiones. Al no tenerse un costo por gestión explícito, se utilizó para el cálculo un costo promedio de gestión. Este se creó desde el monto de deuda facturada del cliente. Un estudio realizado por Mickle Williams en 2019 indica que, para empresas grandes de cobranza, los costos fijos correspondientes a servicios de cobranza preventiva varían entre un 0,5 % y un 5 % de la deuda [24]. Las carteras utilizadas en el modelo presentan tanto carteras preventivas como carteras con deudas más antiguas, lo que aumenta el costo de la gestión. Sin embargo, la empresa en la que se basa este trabajo se enfoca en gestiones digitales, las cuales reducen considerablemente el costo fijo asociadas a estas. Debido a esto, es razonable argumentar que un 3 % del monto de deuda facturada es un porcentaje justo para elegir como costo de gestión. Así, utilizando el promedio de deuda facturada el costo por gestión queda fijado en $2.173.

Tabla 4.3: Análisis monetario en simulación con datos de testeo.

	LIGHTGBM	**ANN**	**Δ(ANN - LIGHTGBM)**
Monto recupero	$5.353.788.459	$6.232.636.410	$878.847.951
Monto recupero perdido	$4.218.413.982	$5.081.485.262	$863.071.280
Δ (Recupero - Perdido)	$1.135.374.477	$1.151.151.148	$15.776.671
Gasto innecesario en gestiones	$62.723.645	$24.142.030	-$38.581.615

Fuente: Elaboración propia.

En un punto de vista para la empresa cliente, la diferencia entre el delta del monto recuperado y el perdido entre ambos modelos no es significativamente mejor para el modelo de redes neuronales con respecto al total de clientes evaluados (229.987). Sin embargo, esta sigue siendo positiva y da la noción que la red neuronal es un mejor modelo para implementar. Esto se ve fuertemente complementado con que los gastos que incurre la empresa de cobranza son considerablemente menores en el modelo de redes neuronales, con estos siendo un 61,5 % menores en esta última. Gracias a esto en un enfoque netamente económico la red neuronal se ve como clara ganadora entre los dos modelos.

4.3. Aplicabilidad

4.3.1. Elección de modelo

En el caso que se llegue a una etapa de aplicación del proyecto, se considera como mejor opción para esta el modelo de redes neuronales. Esto se debe a 2 principales motivos:

- **Análisis económico**: Cómo ya fue mencionado anteriormente, la red neuronal presenta mayor beneficio económico dentro del proyecto. Esto es específicamente importante para la empresa al necesitar reducir sus gastos de cobranza, y al estar en constante competencia con otras empresas de cobranza el poder aumentar minúsculamente la deuda

recuperada es un avance considerable dentro del negocio. De todos modos, cabe destacar que el análisis monetario se basó en diversos supuestos por lo que no refleja en su totalidad la realidad del negocio, por lo que por sí solo este análisis no justifica el uso del modelo.

- **Generalización**: Anteriormente se planteó este trabajo como un paso inicial a un proyecto mayor que involucra más cartera de deudores de la empresa cliente y en lo posible también otros clientes de la empresa. Para esto se hizo énfasis en lograr un modelo generalizable a datos externos al entrenamiento. La red neuronal logró un ajuste similar entre la data de validación y la de testeo, mientras que el modelo base presento un gran sobreajuste que muestra un mal ajuste a datos externos al set de entrenamiento y validación. Debido a esto, entre los dos modelos es evidente que para lograr un modelo generalizable es preferible utilizar el modelo de redes neuronales.

Otro punto importante a considerar es el tiempo de procesamiento de los modelos. Para toda implementación dentro de la empresa se busca una automatización del proceso. Debido a la cantidad de procesos que deben ocurrir mensual o diariamente para el funcionamiento del negocio, el tiempo que está destinado para la carga de estos es un recurso escaso y valioso, al tener un costo asociado a esto ya sea por procesamiento o por no permitir la realización de ciertas operaciones. Por esto, una métrica muy relevante es el tiempo que cualquier algoritmo que se use demora en ejecutarse. Como se desarrolló el trabajo en un equipo local y con diferentes herramientas para aumentar la velocidad de procesamiento que no necesariamente se encuentran en el ambiente en el que corren los procesos de la empresa, no se puede hacer una comparación actualmente de los algoritmos sin previa integración en las plataformas internas de la compañía, pero si nos encontráramos con modelos relativamente similares en rendimiento pero con gran diferencias de procesamiento, este sería un factor decisivo para la elección de estos.

4.3.2. Aplicaciones de modelo elegido

Para poder utilizar el modelo elegido, es necesario darle uso a las clasificaciones que este prediga. En una vista de negocio, un modelo no aportará a la empresa si este no mejora el rendimiento de esta en alguna área. Existen diversas aplicaciones del modelo creado dependiente del enfoque que se le dé. Dentro de estas, se encuentran 3 que logran atacar efectivamente problemáticas y necesidades actuales en la empresa:

4.3.2.1. Control de gasto en gestiones

Como se ha mencionado anteriormente, una problemática actual de la empresa son las problemáticas asociadas al flujo de caja debido a los sucesos económicos relacionados a sus clientes y el virus Covid-19. Para sopesar esto, una necesidad de la compañía es poder disminuir sus costos de cobranza. Una manera de lograr esto utilizando el modelo es el enfocar la fuerza de gestión en los deudores que presenten una predicción de no pago. Así, las gestiones que se realicen tendrán un impacto mayor, aumentando la intensidad o cantidad de gestiones por cliente a estos deudores mientras se disminuye la intensidad a los deudores con una predicción positiva de pago, optimizando el uso de los recursos de cobranza y aumentando el monto recuperado por gestión, lo que conlleva a, de ser necesario, poder mantener los mismos niveles de productividad anteriores al modelo con menos gestiones.

4.3.2.2. Reportería y utilización de predicción como variable

Para poder ordenar entre los diversos tipos de gestiones y canales para realizarlas, cada empresa cliente posee un árbol en el cual se dividen las carteras en diversos segmentos en los cuales a cada uno se le realizan acciones de contacto distintas. Para elegir estos segmentos y las acciones de estos, existe un jefe de cartera el cual monitorea las métricas de éxito de cada empresa cliente. Una aplicación del modelo útil para este jefe de cartera es el poder tener mensualmente un reporte dinámico en el cual pueda ver las predicciones promedio del modelo para diferentes segmentos de datos para complementar sus decisiones al momento de segmentar las carteras. Además de esto, dentro de los filtros de segmentación el agregar la predicción del modelo puede ayudar a sub-segmentar segmentos los cuales se tenga mal rendimiento para poder enfocarse en clientes con mayor probabilidad de pago, por ejemplo. Esto ayudaría a mejorar las métricas de rendimiento con cada compañía para mantener a la empresa competitiva y también ayudaría a encontrar los segmentos iniciales a utilizar para nuevas empresas clientes que se integren al negocio, lo que facilitaría el crecimiento acelerado y abrupto que ha tenido el negocio.

4.3.2.3. Segmentación de cartera

Los algoritmos de clasificaciones funcionan extrayendo reglas de asociación de los datos de entrenamiento para luego poder usar las mismas reglas para calcular un valor de probabilidad. Así, si se extraen estas reglas de asociación, se pueden agrupar las probabilidades de clasificación similares para generar segmentos definidos por la conjunción de los datos que sigan las reglas que lleven a estas probabilidades. El problema de esto es que una característica de las redes neuronales es que, debido a su complejidad interna, es complicado, si no imposible, poder extraer estas reglas. Sin embargo, en el artículo "*Using Neural Network Rule Extraction and Decision Tables for Credit-Risk Evaluation*", se encuentra la manera de extraer estas reglas en una red neuronal, especialmente en un algoritmo de clasificación de riesgo crediticio, el cual posee datos similares a los utilizados en este modelo [25].

Si se realiza una aplicación para el modelo que genere segmentos de clientes, existen dos posibles utilizaciones. En primer lugar, Esto puede generar automáticamente segmentos para los jefes de cartera y, similar al punto anterior, apoyar las decisiones que ellos tomen y mayor aún poder encontrar segmentos relevantes para nuevos clientes que se integren. En segundo lugar, esta herramienta puede ser ofrecida para los clientes como herramienta de reportería para ellos, lo cual podría fomentar la llegada de nuevos clientes y aumentar los ingresos que cada cliente entrega a la empresa.

Capítulo 5

Conclusiones

5.1. Conclusiones generales

La industria de la cobranza es un mercado altamente competitivo. Un pequeño aumento en el rendimiento de las gestiones puede significar que una empresa cliente asigne más clientes para el servicio de cobro, y a su vez una disminución en el rendimiento puede llevar a una asignación de menos deudores, e incluso a la perdida completa del cliente. Gracias a esto, todo proyecto que logré mejorar la productividad o a mantener el nivel de operaciones en un tiempo de recesión y crecimiento acelerado es un modelo que debería ser probado dentro de la empresa.

Para llegar al trabajo realizado y a sus correspondientes resultados, se investigó exhaustivamente sobre la industria de la cobranza. Sin embargo, esto mismo llevó a entender lo recatada que es en un nivel de negocio para mostrar indicadores y avances en estas, lo que se puede explicar por la competitividad mencionada anteriormente. Si bien esto está fundamentado fuertemente por esto, también ralentiza los avances que se puedan ocasionar en la industria.

Con respecto al proyecto, resulta evidente la eficacia de la predicción del pago de deuda y su vasto número de aplicaciones, pero específicamente este trabajo da a ver la importancia de la información utilizada para crear estos modelos. Debido a los resultados tanto del modelo base como el modelo final se puede concluir que existe un gran espacio de mejora para estas métricas y un factor imprescindible a revisar para esto es la selección de datos utilizados para el entrenamiento. Ya sea por cantidad o calidad, la revisión de la información utilizada sería el primer punto recomendado para revisar este trabajo, esto será ahondado en la próxima sección.

5.2. Trabajo futuro

Debido a que este trabajo se centra en la primera etapa de un proyecto de mayor escala, es imprescindible planificar como poder darle continuación a lo realizado. Existen diversas aristas por las cuales se puede abordar en un futuro el trabajo realizado para mejorar tanto los modelos realizados como para apoyar la implementación e integración de nuevas empresas al proyecto en general. A continuación, se detallan las más importantes.

5.2.1. Mejora de base utilizada

Ahondando en lo comentado en el punto anterior, una mejor base de datos puede llevar a mejoras significativas en los resultados del modelo. La manera más directa de lograr esto es agregando información de más meses al modelo, lo cual requiere la espera de el paso de los meses para añadir la información. Al momento de finalizar el trabajo explicado, ya existe una gran cantidad de información utilizable para reentrenar el modelo, por lo que sería este el primer paso para verificar si esta nueva información puede mejorar las métricas presentadas.

Además de poblar con más datos el modelo, también se puede trabajar con la cantidad de información que se entrega en cada uno de estos datos. Si bien se descartó en un principio la realización de una ingeniería de variables entre las columnas que ya se tienen en el modelo, esto puede ser revisitado a modo experimental para ver si llevan a una mejora en el modelo planteado, además de otras técnicas de preprocesamiento mencionadas, como segmentación de datos y análisis de componentes. También, existen diversas bases externas a la utilizadas que pueden ser integradas al modelo ya existente para complementar la información que se tiene por cliente, como informes comerciales sobre deudas del cliente externas a la empresa cliente e información de inmueble, entre otras. Esto en primer lugar no se agregó debido a que existe un costo asociado a la obtención de estos datos.

5.2.2. Integración de historial de cliente

La siguiente opción que se tiene, también ligada a los datos, es integrar el historial de deuda de un cliente dentro del modelo, como fue mencionado anteriormente [11]. Actualmente, cada mes de datos que tiene el modelo posee toda la cartera mensual de la empresa cliente, y no existe ningún indicador que ayude a ligar la información que se está ingresando sobre un deudor con su información pasada. Por ejemplo, si existe un deudor A que debía pagar tanto en junio como en julio, dentro del modelo no existe forma de que este sepa que la información de julio pertenece al mismo deudor A que ya poseía información en junio. El comportamiento de pago histórico es intuitivamente una variable que aporte gran información para poder predecir el comportamiento que tendrá un deudor, por lo que para un trabajo futuro se deberá probar con esto.

Existen diversas maneras de lograr esta asociación. La manera que se sugiere para hacer esto es la utilización de topologías de redes neuronales que puedan aprender dependencias de orden dentro de la información brindada, para así poder ingresar información temporal de los clientes junto con la información estática que tengan estos. Los ejemplos más clásicos y utilizados son **LSTM** o *Long Shot-Term Memory* [26], y **GRU**, también conocida como *Gated Recurrent Unit* [27].

5.2.3. Resto de cartera y otros clientes

Para lograr el objetivo final del trabajo, se tendrá que considerar el proyecto a modo general y se deberá buscar maneras de implementar el modelo para más bases de datos. En primer lugar, es necesario probar con la data de la empresa cliente que no fue utilizada dentro del entrenamiento (deudores con mora mayor a 30 y vencimiento en la segunda quincena).

Esto a priori no presenta mayor complicación al tener las mismas variables en los datos y solo deberá tener en cuenta si el mismo modelo sigue siendo el óptimo para la realización del proyecto.

Ya en segundo lugar y a modo más avanzado, es necesario evaluar la opción de que el modelo sea utilizado para diversas compañías. Para esto podrán surgir diversas problemáticas en el rendimiento del modelo como tal. A modo inicial, los puntos a considerar más importantes son la data disponible en las otras compañías y el nivel de agregación del modelo (servirá para todas las compañías o solo para las compañías dentro del rubro del retail financiero)

5.2.4. Implementación

Por último, es necesario tener en mente el cómo se implementará el modelo en la empresa, de ser implementado. La empresa cuenta con un flujo de carga de datos y de procesamiento de datos muy estricto que corre de manera diaria para poder tener la información disponible para la gestión de deudores día a día, por lo que el tiempo que el modelo demore en procesar es un factor imprescindible a considerar para la implementación del proyecto, como fue mencionado anteriormente, y lo cual es un factor relevante tanto para elegir como esquematizar el proyecto dentro del flujo (temporalidad del reentrenamiento y de la realización de predicciones), como para elegir qué modelo es el más adecuado para la empresa.

Bibliografía

[1] D. I. Molina, *Fintech: Lo que la tecnología hace por las finanzas*. Profit Editorial, 2016.

[2] *Fintech: América Latina 2018: Crecimiento y consolidación*. Inter-American Development Bank (IDB), nov 2018.

[3] Fintech Chile, "Primer estudio Fintech en Chile: Desafío y oportunidades," tech. rep., CB Insights, 2019.

[4] M. Loukides, *What Is Data Science?* O'Reilly Media, 2011.

[5] N. M. Flores Vera, "Rediseño del proceso de cobranza para una empresa de telecomunicaciones," 2012.

[6] R. Farber, *CUDA Application Design and Development*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1st ed., 2011.

[7] K. L. M. Figuerola Duthurburu, "Identificación de mejoras en el proceso de cobranza preventiva y temprana de productos de banca personal del Banco Interamericano de Finanzas (BanBif)," 2019.

[8] U. A. Wijewardhana, "A Mathematical Model for Predicting Debt Repayment: A Technical Note," *Australasian Accounting, Business and Finance Journal*, vol. 12, no. 3, pp. 127–135, 2018.

[9] L. M. Healy, "Logistic regression: An overview," *Eastern Michighan College of Technology*, 2006.

[10] J. Zou, Y. Han, and S.-S. So, "Overview of artificial neural networks," in *Artificial Neural Networks*, pp. 14–22, Springer, 2008.

[11] R. van de Geer, Q. Wang, and S. Bhulai, "Data-driven consumer debt collection via machine learning and approximate dynamic programming," *Available at SSRN 3250755*, 2018.

[12] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[13] S. Neelamegam and E. Ramaraj, "Classification algorithm in data mining: An overview," *International Journal of P2P Network Trends and Technology (IJPTT)*, vol. 4, no. 8, pp. 369–374, 2013.

[14] Microsoft, "LightGBM's documentation," 2020.

[15] M. Buscema, "A brief overview and introduction to artificial neural networks," *Substance use & misuse*, vol. 37, no. 8-10, pp. 1093–1148, 2002.

[16] N. Buduma, *Fundamentals of Deep Learning*. O'Reilly, junio 2017: ed., 2017.

[17] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[18] O. Demir-Kavuk, M. Kamada, T. Akutsu, and E.-W. Knapp, "Prediction using stepwise L1, L2 regularization and feature selection for small data sets with large number of features," tech. rep., 2011.

[19] J. Bjorck, C. Gomes, B. Selman, and K. Q. Weinberger, "Understanding Batch Normalization," tech. rep.

[20] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software disponible en tensorflow.org.

[21] J. Patterson and A. Gibson, *Deep learning: A practitioner's approach*. .ºReilly Media, Inc.", 2017.

[22] A. I. R. L. Azevedo and M. F. Santos, "KDD, SEMMA and CRISP-DM: a parallel overview," *IADS-DM*, 2008.

[23] G. P. Zhang, "Avoiding pitfalls in neural network research," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 1, pp. 3–16, 2006.

[24] M. W. Foretic González, "Plan de negocios para un emprendimiento de un sistema en línea de cobranza y recaudación en chile," 2019.

[25] B. Baesens, R. Setiono, C. Mues, and J. Vanthienen, "Using neural network rule extraction and decision tables for credit-risk evaluation," *Management science*, vol. 49, no. 3, pp. 312–329, 2003.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (gru) neural networks," in *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pp. 1597–1600, IEEE, 2017.

Anexo A

Código

Código A.1: Función de Creación de modelo inicial y afinación de hiperparámetros.

```
import pandas as pd
import numpy as np
import tensorflow as tf
import keras
from keras import layers
from keras import models
from keras import utils
from keras.layers import Dense
from keras.models import Sequential
from keras.optimizers import Adam
from keras.optimizers import SGD
from keras.optimizers import RMSprop
from keras.callbacks import LearningRateScheduler
from keras.callbacks import History
from sklearn.metrics import f1_score
from sklearn.metrics import roc_auc_score
def run_nn_tuning(train_X, train_y, val_X, val_y, test_X, test_y, iterations = 50):
    best_auc = -1
    test_best_auc = -1
    count = 0

    for i in range(iterations):
        print('iteration number', count)
        count += 1
        try:

            params = {}
            params['learning_rate'] = 0.1
            params['batches'] = np.random.choice([64, 128, 256, 512])
            params['epochs'] = np.random.randint(10, 100)
            params['neurons'] = np.random.randint(25, 200)
            params['momentum'] = np.random.uniform(0, 1)
            params['decay'] = np.random.uniform(0.01, 0.1)
            params['neuron_rate'] = np.random.uniform(0.1, 0.2)
```

```
        params['optimizer'] = np.random.choice(['rmsprop', 'adam', 'sgd'])
        params['hidden_layers'] = np.random.randint(3, 8)
        params['activation_f'] = np.random.choice(['relu','sigmoid'])
        params['topology'] = np.random.choice(['exp','funnel', 'constant'])
        print(params)
        if params['optimizer'] == 'rmsprop':
            optimizer = RMSprop(lr=params['learning_rate'], momentum= params['
    ↪ momentum'], decay=params['decay'])
        elif params['optimizer'] == 'adam':
            optimizer = Adam(lr=params['learning_rate'], decay=params['decay'])
        elif params['optimizer'] == 'sgd':
            optimizer = SGD(lr=params['learning_rate'], momentum= params['momentum'
    ↪ ], decay=params['decay'])

        if params['topology'] == 'exp':
            model = Sequential()
            model.add(Dense(params['neurons'], activation='relu', input_dim =25))
            current_neurons = params['neurons']
            for i in range(params['hidden_layers']):

                if i <= params['hidden_layers']/2:
                    current_neurons = int(current_neurons*np.exp((i+1)*params['
    ↪ neuron_rate']))
                    model.add(Dense(current_neurons, activation=params['activation_f']))
                else:
                    current_neurons = int(current_neurons*np.exp((-(i+1))*params['
    ↪ neuron_rate']))+1
                    model.add(Dense(current_neurons, activation=params['activation_f']))

            model.add(Dense(1, activation='sigmoid'))
            model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=
            ['AUC', 'Precision', 'Recall'])
        elif params['topology'] == 'funnel':
            model = Sequential()
            model.add(Dense(params['neurons'], activation='relu', input_dim =25))
            current_neurons = params['neurons']*(params['hidden_layers']+1)
            for i in range(params['hidden_layers']):
                model.add(Dense(current_neurons, activation=params['activation_f']))
                current_neurons = int(current_neurons*np.exp((-(i+1))*params['
    ↪ neuron_rate']))+1

            model.add(Dense(1, activation='sigmoid'))
            model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=
            ['AUC', 'Precision', 'Recall'])
        elif params['topology'] == 'constant':
            model = Sequential()
            model.add(Dense(params['neurons'], activation='relu', input_dim =25))

            for i in range(params['hidden_layers']):
                model.add(Dense(params['neurons'], activation=params['activation_f']))

            model.add(Dense(1, activation='sigmoid'))
```

```
82              model.compile(loss='binary_crossentropy', optimizer=optimizer)
83
84
85          print(model.summary())
86          def exp_decay(epoch):
87              lrate = params['learning_rate'] * np.exp(-params['decay']*params['epochs'])
88              return lrate
89
90          # Actualización de tasa de aprendizaje
91          loss_history = History()
92          lr_rate = LearningRateScheduler(exp_decay)
93          callbacks_list = [loss_history, lr_rate]
94
95          model.fit(X_train.values, y_train.values, validation_data=(X_val.values,y_val.
    ↪ values),  epochs= params['epochs'], callbacks=callbacks_list, batch_size= params['
    ↪ batches'])
96
97          val_y_pred = model.predict(val_X.values)
98          test_y_pred = model.predict(test_X.values)
99
100         auc_val = roc_auc_score(val_y, val_y_pred)
101         print('auc en val:', auc_val)
102         y_pred = (test_y_pred > threshold)
103         auc = roc_auc_score(test_y, test_y_pred)
104         print('auc en test:', auc)
105         if best_auc < auc_val:
106             best_auc = auc_val
107             test_auc = auc
108             pp = params
109         if test_best_auc < auc:
110             test_best_auc = auc
111             val_auc_auc = auc_val
112             test_pp = params
113
114     except:
115         print('failed with')
116         print(params)
117     print("*" * 50)
118     print("FOR VAL")
119     print('Maximum is: ', best_auc)
120     print('test: ', test_auc)
121     print('Used params', pp)
122     print("FOR TEST")
123     print('Maximum is: ', test_best_auc)
124     print('val: ', val_auc_auc)
125     print('Used params', test_pp)
126 return pp, test_pp
```

Anexo B

Tablas

Tabla B.1: Descripción de las variables de base inicial

Nombre	**Descripción**
cuenta	identificador de la cuenta.
sexo	sexo del deudor.
clasificacion	clasificación de deudor y deuda interna a la empresa cliente.
cupo_disponible	cupo disponible en tarjeta de deudor.
mora	dias de mora del deudor (días desde su fecha de vencimiento a la fecha sin pagar).
vencimiento	día de vencimiento del deudor (día del mes donde tiene que pagar su deuda).
saldo_total	cupo total.
total_vencido	total de deuda vencida (sin pagar pasada su vencimiento).
por_vencer	total de deuda por vencer (que debe pagar en su próximo vencimiento).
total_deuda	total que debe el cliente.
promesas_cumplidas	veces que pago su deuda luego de un compromiso de pago anteriormente.
promesas_rotas	veces que no pago su deuda luego de un compromiso de pago anteriormente.
monto_cuota_mas_morosa	monto cuota más antigua que esta en mora.
deuda_facturada	total de deuda que debió haberse pagado a la fecha.

Fuente: Elaboración propia.

Tabla B.2: Descripción de métricas de evaluación utilizadas.

Nombre	Descripción	Nombre en tabla
Monto recuperado	Monto total que el modelo predice que será pagado y es pagado (asumiendo que los pagos de deuda son del total de esta). Esta métrica lleva a un marco monetario la eficacia del modelo, pudiendo justificar más fácilmente la aplicación del modelo en el negocio.	monto_recuperado
Porcentaje monto recuperado	Porcentaje del total de monto de deuda que el modelo predijo correctamente que se recupera. Ayuda a normalizar los resultados monetarios para así poder comparar entre bases de datos más fácilmente.	%_monto_recuperado
Porcentaje monto recuperado de pagos efectivos	Este porcentaje nos indica el porcentaje del monto total que se recupero fue bien predicho por el modelo. Se diferencia del Porcentaje monto recuperado ya que no toma en consideración las deudas que no se pagan, por lo que brinda una visión más ligada al rendimiento del modelo.	%_monto_recuperado_pagos
Número de recuperos perdidos	Número de casos en los cuales hubo una predicción positiva en casos que realmente eran negativos (se pensó que pagaría, pero no pagó). Estos casos son relevantes ya que, si dentro de la aplicación se busca generar estrategias más potentes a los casos predichos negativos, estos casos no tendrían dicho trato por lo que al disminuir este número se puede argumentar fácilmente que el modelo mejora.	n_recuperos_perdidos
Monto de recuperos perdidos	Suma de deuda facturada de los recuperos perdidos. El recuperar más monto de menos clientes es un escenario mejor para la mayoría de los clientes, por lo que con esta métrica se puede complamentar la comparación monetaria de los modelos.	monto_recupero_perdido
Porcentaje monto recuperos perdidos	Ratio entre el monto de recupero perdido y el total de la deuda facturada en el modelo. Permite nuevamente una comparación monetaria entre modelos con distintas bases de prueba la cual estará normalizada dependiendo cuanta deuda hay en cada base.	%_monto_recupero_perdido
Número de gastos innecesarios	Número de casos en los cuales hubo una predicción negativa en casos que realmente eran positivos (se pensó que no pagaría, pero pagó). Similar a los recuperos perdidos, si se desean fijar distintas estrategias dependiendo de la predicción obtenida, estos casos implicarían un costo gastado en una gestión más fuerte la cual no sería necesaria para concretar el pago (según la predicción, pagaría de igual manera). Se ve el número de casos ya que se puede asumir un costo promedio igual para toda gestión, por lo que solo con esta métrica se puede tener una comparación monetaria entre los modelos.	n_gasto_innecesario
Número de pagos predichos	Cantidad de predicciones positivas que resultaron pertenecer a clientes que pagaron. Esta variable ayuda a tener una visión más integral del modelo y puede en un futuro ser utilizada para realizar otras métricas relevantes como la exhaustividad del modelo, entre otras.	pagos_predichos
Número de no pagos predichos	Cantidad de predicciones negativas que resultaron pertenecer a clientes que no pagaron. Tiene un uso similar a los pagos predichos.	no_pagos_predichos

Fuente: Elaboración propia.