PROPUESTA DE IMPLEMENTACIÓN DE UN PILOTO DE MLOPS SIGUIENDO LOS LINEAMIENTOS DEFINIDOS POR LA EMPRESA WIZIT MIND BLOWING SOLUTIONS S.A.S.

Proyecto de trabajo de grado para optar por el título de:
INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

DANIEL SANTIAGO VÁSQUEZ ASTAIZA

Director: PhD. GUSTAVO ADOLFO RAMÍREZ GONZÁLEZ
Asesor: Ing. RAFAEL ESTEBAN CERÓN ESPINOSA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE TELEMÁTICA
SERVICIOS AVANZADOS SOBRE INTERNET
POPAYÁN, 2022

AGRADECIMIENTOS

A William Felipe, por ser el hombre que con amor y entrega abnegada, ha provisto por encima de su propia humanidad desde el primer momento y por ser mi ejemplo para convertirme en la persona que soy hoy.

A Luz Marina, la causa de mis lágrimas y objeto de mis oraciones más sinceras y profundas. La mujer de mi vida que con su entrega desinteresada me ha enseñado el verdadero significado del amor.

A Catalina, por ser ese apoyo y ese soporte tan valioso en los momentos de flaqueza que me permite mantenerme de pie aunque los vientos soplen con toda violencia. Quien con amor y ternura me enseña constantemente el valor del perdón. Y que junto a Andrés David me dio el regalo y el privilegio de ser tío de dos hermosos muchachos.

A Jero y a Josu, por ser mis grandes maestros con su nobleza, curiosidad y su ternura porque a pesar de su corta edad es mucho lo que han sembrado en mí. No me alcanzarán la vida y las palabras para expresarles todo mi amor. Ustedes lo son todo en mi vida.

A Anna María, a quien adoro profundamente y que junto a mi madre y a Cata es una de las mujeres de mi vida, quien alegra mi vida, y constantemente me enseña de paciencia, de amor y perdón. Quien se ganó un lugar de privilegio en mi vida, en mi familia y en mi corazón, y a quien agradezco toda la vida por abrirme un espacio lleno de amor entre Arturo, Mireya, Sofi, Kihara y Tari.

A Ihann Miguel, mi mejor amigo, por casi dos décadas de amistad y por ser esa persona incondicional y ese amigo que todos deberíamos tener en nuestras vidas, por seguirme la cuerda en todos mis arranques, por ser compañero fiel y sabio consejero siempre acertado en cada momento.

A Ivania, por demostrarme que el tiempo y la distancia no pueden separar a dos corazones que se quieren de verdad. Por su incondicional apoyo, por prestarme su hombro y su oído de manera sincera. Por abrirme las puertas de su hogar y de su corazón; y por darme el honor de ser su mejor amiko.

A Mía, por ese amor tan bello e incondicional que a pesar del tiempo sigue en mi corazón y en mis pensamientos. Espero que la vida te sea benevolente y que el viento sople siempre a tu favor, donde quiera que estés. Siempre te voy a amar, perrita.

A Gustavo, a Esteban y a todo el equipo de TBBC, hoy Wizit, por todo el apoyo, por confiar en mis capacidades y aportar de forma tan valiosa con tantas enseñanzas, a nivel personal y profesional; y sobre todo por toda la paciencia en la realización de este proyecto.

Finalmente, me gustaría agradecer a Dios por darme la resiliencia y la obstinación necesarias para poder llegar hasta este punto y por permitirme edificar cada uno de mis logros a nivel académico, personal y profesional.

Dedicado a mi viejo querido, Don Eliécer Astaíza Ordóñez, te amo donde sea que estés.

TABLA DE CONTENIDO

LISTA DE ACRÓNIMOS

AI	*Artificial Intelligence:* Inteligencia artificial.
ML	*Machine Learning:* Aprendizaje automático.
ANN	*Artificial Neural Network:* Red neuronal artificial.
EDA	*Exploratory Data Analysis:* Análisis exploratorio de datos.
MLE	*Machine Learning Engineering:* Ingeniería de Machine Learning.
MLOps	*Machine Learning Operations:* Operaciones de Machine Learning.
AWS	*Amazon Web Services.*
DNN	*Deep Neural Network:* Red neuronal profunda.
DT	*Decision Tree:* Árbol de decisión.
SVM	*Support Vector Machine:* Máquina de vector de soporte.
KNN	*K-Nearest Neighbors:* K-vecinos más cercanos.
RF	*Random Forest:* Bosque aleatorio.
LR	*Logistic Regression:* Regresión logística.
XGBoost	*eXtreme Gradient Boost.*
AUC	*Area Under the Curve:* Área bajo la curva.
ROC	*Receiver Operating Characteristic:* Característica Operativa del Receptor.
CI	*Continuous Integration:* Integración continua.
CD	*Continuous Delivery:* Entrega continua.
LASSO	*Least Absolute Shrinkage and Selection Operator.*

1. INTRODUCCIÓN

1.1. Planteamiento del problema

La inteligencia artificial juega un rol importante en el mundo debido al creciente valor que sus aplicaciones representan para gobiernos, empresas y organizaciones. Grandes empresas como Tesla, están construyendo vehículos que pueden ser conducidos por una red neuronal [1], otras empresas como Siemens están desarrollando aplicaciones basadas en imagenología para aportar información relevante sobre el estado de los pacientes con cáncer y contribuir en la toma de decisiones a nivel clínico [2], solo por mencionar algunos casos de aplicación. A pesar de su gran popularidad, aún existen algunos escenarios en los que dicha tecnología podría apoyar algunos procesos en las empresas, especialmente del sector financiero.

En Colombia, la banca tradicional otorga el acceso a sus productos y servicios basándose en un indicador conocido como *credit score* o puntaje crediticio. Dicho puntaje es individual y se estima utilizando un algoritmo que considera cinco importantes factores a saber: El historial de pagos, la relación de la deuda, la extensión del historial de crédito, monto del crédito que se está solicitando y la variedad de productos y servicios de crédito que posea una persona [3].

Según Datacrédito-Experian, el ***60%*** de la calificación se basa en elementos relacionados al historial de crédito [3]. Así mismo, según cálculos realizados por el Banco de la República de Colombia, a junio del 2021 los créditos otorgados por los bancos a las personas naturales es del ***60%***, el cual es menor con relación a los años anteriores, adicional a ello, solo el ***20%*** de los créditos solicitados tienen la finalidad de consumo [4]. Teniendo en cuenta lo expuesto, es válido afirmar que aquellas personas que no cuentan con ningún tipo de historial de crédito o cuentan con baja calificación tienen menores posibilidades de acceder a productos y servicios financieros, esto se reafirma si se tiene en cuenta que el ***76%*** de los créditos otorgados a personas con puntajes de crédito considerados como bajos (Menores a ***694***) son otorgados por las *Fintech* [5].

En consecuencia con lo anterior el equipo de Wizit Mind Blowing Solutions S.A.S (de ahora en adelante Wizit), que es una empresa colombiana con seis años de experiencia desempeñándose en las áreas de consultoría y analítica de datos, se plantea la posibilidad de utilizar su experticia en dichos campos para construir una solución que permita a las personas que no cuenten con historial crediticio acceder a productos y servicios financieros.

Siguiendo lo definido en el trabajo complementario *“PROPUESTA DE UN CONJUNTO DE LINEAMIENTOS BASADO EN MLOPS PARA EL SOPORTE DE LA GESTIÓN EN PROYECTOS DE CIENCIA DE DATOS EN LA EMPRESA WIZIT MIND BLOWING SOLUTIONS S.A.S.”* para la implementación de soluciones en *Machine Learning* utilizando el kit de herramientas para *MLOps* de Amazon Web Services; y teniendo en

cuenta la necesidad de la empresa por desarrollar un algoritmo que permita predecir si una persona, con o sin historial de crédito, es idónea para acceder a productos o servicios financieros, surge la siguiente pregunta de investigación:

¿Cómo determinar el perfil de riesgo de una persona, sin historial de crédito, para recibir un producto o servicio financiero haciendo uso de MLOps?

1.2. Objetivos

Para dar respuesta a la pregunta de investigación planteada, se establece un objetivo general, apoyado sobre tres objetivos específicos:

- **Objetivo general:** Desarrollar un piloto de una aplicación web que integre herramientas tecnológicas de Amazon Web Services orientadas a procesos de *MLOps*.

- **Objetivos específicos:**

    - Definir la fuente de datos para la construcción y entrenamiento del modelo de datos.
    - Implementar el modelo de datos utilizando los módulos de *MLOps* recomendados por el proveedor *Cloud* (AWS).
    - Probar el desempeño de la implementación.

1.3. Partes de la memoria

A continuación se plantean los capítulos donde se detalla la realización de este trabajo:

- **Capítulo 1 — Introducción:** Presenta el contexto, describe el problema de investigación y define la estructura del trabajo realizado.

- **Capítulo 2 — Estado del arte:** Presenta los conceptos, técnicas y tecnologías que soportan este trabajo, adicional a las experiencias y resultados obtenidos por las diferentes investigaciones en el campo de la predicción de impago utilizando *Machine Learning*, y los lineamientos de *MLOps* que serán referente para el desarrollo del trabajo.

- **Capítulo 3 — Implementación:** Presenta el proceso de construcción de un *pipeline* de *Machine Learning* basado en *MLOps* utilizando las herramientas y servicios del proveedor *Cloud* AWS. Se describen las etapas de selección de la fuente de datos, preprocesamiento, construcción, entrenamiento y despliegue del modelo.

- **Capítulo 4 — Resultados:** Presenta los resultados de la investigación a través de los diferentes escenarios de prueba, describe de manera general el

funcionamiento del prototipo desarrollado y se comenta el desempeño del mismo desde el punto de vista del negocio.

- **Capítulo 5 — Conclusiones:** Presenta las conclusiones obtenidas a partir de este trabajo, y las consideraciones para el desarrollo de trabajos futuros.

2. ESTADO DEL ARTE

En este capítulo se recogen los conceptos, técnicas y tecnologías que soportan este trabajo. De igual forma, se describen las investigaciones relevantes para este trabajo desarrolladas en el marco de *Machine Learning* y de predicción de riesgo de impago.

Este capítulo se encuentra dividido en las siguientes secciones:

- **Conceptos y definiciones:** Contiene conceptos y definiciones que son relevantes para esta investigación en los niveles técnico, de negocio y metodológico. Además de presentar los servicios del proveedor AWS que son más relevantes para el desarrollo e implementación de este proyecto.

- **Entorno de servicios web:** Contiene información sobre los servicios web de AWS que serán utilizados durante el desarrollo de este proyecto, además de presentar la definición y la utilidad que poseen.

- **Trabajos relacionados:** Esta investigación preliminar se apoyó en un análisis cienciométrico para presentar un panorama del estado actual de las investigaciones y trabajos relacionados, con el fin de determinar un punto de partida para este trabajo a nivel técnico.

2.1. Conceptos y definiciones

Con el fin de llevar a cabo este piloto y resolver la pregunta de investigación planteada anteriormente, es necesario tener en cuenta algunos conceptos teóricos relacionados con la realización de este proyecto.

- ***Machine Learning*:** Es una rama de la inteligencia artificial y de las ciencias de la computación que se enfoca en el uso de algoritmos y datos para imitar la forma en la que los seres humanos aprenden, mejorando gradualmente su precisión [6].

  Algunos de los algoritmos de *Machine Learning* más relevantes para el desarrollo de este trabajo son los siguientes:

  - ***Decision Tree:*** Un árbol de decisión es un algoritmo que se puede utilizar para tareas de clasificación y regresión. Posee una estructura jerárquica, similar a un árbol, que consiste de un nodo raíz, ramas derivadas del mismo, nodos internos y nodos de hoja [7].

  - ***Random Forest:*** Es un algoritmo compuesto por un arreglo de *Decision Trees*. Obtiene su nombre de la aleatoriedad con la que se generan subconjuntos de características para asegurar una baja correlación entre los *Decision Trees* [8]**.**

- ***Artificial Neural Network:*** Es un modelo computacional que consiste de varios elementos de procesamiento que reciben entradas y entregan salidas basadas on sus funciones de activación previamente definidas. Este tipo de algoritmos se asemejan al funcionamiento de un cerebro humano [9].

- ***Logistic Regression:*** Es un tipo de modelo estadístico que es comúnmente utilizado para clasificación y análisis predictivo. Permite estimar la ppribabilidad de ocurrencia de un evento a partir de un conjunto de datos de variables independientes. Este tipo de algoritmo se utiliza para problemas de regresión y clasificación [10].

- ***XGBoost:*** *XGBoost* o *Extreme Gradient Boosting*, es una librería de *Machine Learning* de código abierto escalable, distribuída y potencializada por gradientes que hacen uso de *Decision Trees*. Este algoritmo se utiliza para tareas de regresión, clasificación y calificación [11].

- ***Exploratory Data Analysis*****:** El análisis exploratorio de datos o EDA se utiliza para analizar e investigar *datasets* y hacer un compilado de sus principales características, usualmente se utilizan métodos para la visualización de los datos. El EDA también contribuye a determinar el mejor curso de acción para la manipulación de las fuentes de datos para obtener las respuestas deseadas, esto permite encontrar patrones, detectar anomalías, probar una hipótesis o verificar un supuesto.

  Su propósito fundamental consiste en permitir la identificación de errores, reconocimiento de patrones, detecciœn de valores atípicos y eventos anómalos, además de contribuir a determinar qué pueden revelar los datos, más allá del modelado y la prueba de hipótesis, esto con el fin de mejorar el entendimiento que se tiene sobre las variables del *dataset* y las relaciones entre las mismas [12].

- ***Feature Selection*****:** La selección de características o *features* es un proceso que ayuda a descartar *features* que no son útiles para el análisis con el fin de disminuir la complejidad del modelo resultante. El resultado final es un modelo con poca o ninguna degradación en su poder predictivo. Existen tres clases de técnicas de selección de características [13]:

  - ***Filtering*****:** Las técnicas de filtrado realizan el preprocesado de las características para eliminar aquellas que tienen menores probabilidades de ser útiles para el modelo. Por ejemplo, se puede calcular la correlación de todas las variables independientes frente a la variable objetivo y se descartan aquellas que queden por debajo de un umbral definido. Esta técnica es la menos costosa en términos de potencia de cómputo y tiempo, pero no toman en cuenta el modelo utilizado, por tanto pueden no

seleccionar las características más adecuadas [13].

- ***Embedded*:** Esta técnica es costosa en tiempo y capacidad de cómputo, pero permiten probar distintos subconjuntos de características evitando eliminar aquellas características que por sí solas no aportan al análisis pero en combinación con otras pueden ofrecer mayor poder de predicción. Este método funciona generando un puntaje de calidad para cada subconjunto de datos [13].

- ***Wrapper*:** Esta técnica realiza la selección de características como parte del entrenamiento del modelo, como es el caso de un *Decision Tree* que de forma inherente escoge una característica de la cual se desprende para crear el árbol en cada paso del proceso de entrenamiento. Otro ejemplo es *LASSO* [14] que implementa un tipo de regularización L1 que penaliza severamente características no esenciales o altamente correlacionadas [13].

- ***Pipeline*:** Es el proceso de almacenamiento y encolado de tareas e instrucciones que son ejecutadas simultáneamente y de manera organizada por un procesador [15]. Para el caso de *Machine Learning c*onsta de las siguientes etapas:

  - **Preprocesamiento:** Es el proceso de manipulación y transformación de los datos y comprende todas las técnicas de análisis que contribuyan a mejorar la calidad de un *dataset*, permitiendo un mayor desempeño en la predicción del modelo que lo consuma [16], [17].

  - **Entrenamiento:** Es el proceso en el que se proporcionan datos de entrenamiento a un modelo de *Machine Learning*. Estos datos contienen la respuesta correcta permitiendo que el modelo identifique los patrones necesarios para realizar una predicción [18].

  - **Evaluación:** El fin de un modelo es generalizar patrones que le permitan realizar predicciones, es por ello que es necesario comprobar el rendimiento del modelo sobre muestras de datos sin analizar. Usualmente se divide el *dataset* en dos muestras donde se utiliza una para entrenamiento y la otra para validación. El desempeño del modelo sobre el *dataset* de validación determina las métricas que miden la precisión y rendimiento del modelo [19].

  - **Despliegue:** Es el proceso en el que un modelo de *Machine Learning* es puesto en el entorno de producción, con el fin de hacer sus predicciones disponibles para la toma de decisiones a nivel de negocio, la interacción con sus usuarios, entre otros usos [20].

- ***Resampling*:** Comprende métodos que resultan en el cambio de la proporción de las clases dentro de un *dataset*, con el propósito de obtener información adicional sobre el modelo entrenado [21].

- **DevOps:** Es una combinación de prácticas y herramientas que incrementan la capacidad para proveer aplicaciones y servicios de manera rápida en comparación con los procesos tradicionales [22]**.**

- **CI/CD:** Hace parte de DevOps y es una parte importante del proceso de *MLOps*. Combina las prácticas de integración continua o *Continuous Integration* y entrega continua o *Continuous Delivery*. CI/CD minimiza la intervención manual necesaria para realizar el paso del código a producción y a su vez abarca la compilación, prueba y despliegue del mismo. A través de un *pipeline* de CI/CD, se pueden realizar cambios de código que serán probados y enviados para ser entregados y desplegados [23].

    - ***Continuous Integration*:** Es la práctica de integrar todos los cambios de código en la rama principal de un repositorio compartido. La integración continua permite detectar errores y fallas de seguridad con mayor facilidad y en una etapa más temprana del desarrollo. Al unir cambios frecuentemente y ejecutar las pruebas automáticas y los procesos de validación, se minimizan las probabilidades de generar conflictos en el código, incluso cuando existen múltiples desarrolladores en el mismo proyecto. Adicionalmente, se reduce el tiempo para obtener *feedback*, corregir *bugs* y problemas de seguridad. La última parte de este proceso comprende probar la calidad del código [23].

    - ***Continuous Delivery*:** Es una práctica que trabaja en conjunto con la integración continua para automatizar el aprovisionamiento de la infraestructura y el proceso de despliegue [23].

- ***Thresholding*:** Es un criterio basado en un escalar que es comparable a la calificación de la capacidad predictiva de un modelo. Se utiliza para separar las clases positivas de las clases negativas [24].

- ***Confusion Matrix*:** Se trata de una matriz que muestra las predicciones y los resultados reales de clasificación. Consta de $n \times n$ elementos, donde n es el número de clases de la variable objetivo [25]. Para un problema de clasificación binaria, la matriz de confusión tiene la siguiente forma:

	Clases reales	
	Positivas	**Negativas**

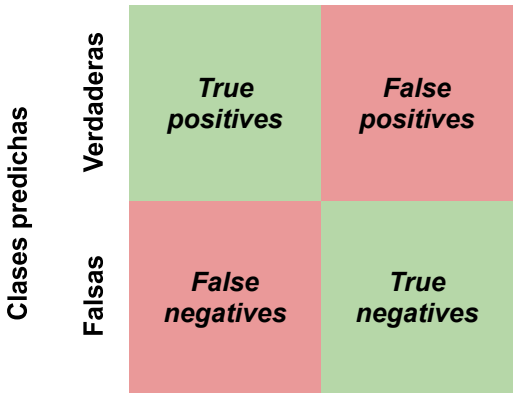


Figura 1: Matriz de confusión. Fuente propia.

Un verdadero positivo o *true positive (TP)* es un resultado en el que el modelo predice correctamente la clase positiva. De forma análoga, un verdadero negativo o *true negative (TN)* es la predicción correcta de la clase negativa. Por otro lado, un falso positivo o *false positive (FP)* es el resultado donde el modelo predice la clase positiva de forma incorrecta. De forma análoga, un falso negativo o *false negative (FN)* es la predicción incorrecta de la clase negativa. A partir de estos valores, es posible conocer las siguientes métricas:

- ***Precision*:** La precisión busca identificar qué proporción de las predicciones positivas realizadas es correcta [26]. La fórmula que la define es la siguiente:

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

Formula 1: Precision. Tomado de [27].

- ***Recall*:** El *Recall*, también conocido como *True Positive Rate*, busca identificar qué proporción de las predicciones positivas realizadas es identificada correctamente [26]. Su fórmula es la siguiente:

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

Formula 2: Recall o True Positive Rate. Tomado de [27].

- ***ROC curve*:** La curva ROC o Característica Operativa del Receptor es una gráfica de diagnóstico que examina una colección de predicciones probables realizadas por un modelo. Se observan las tasas de verdaderos positivos y de falsos positivos utilizando un conjunto de diferentes umbrales [28].

- **AUC:** Es el área bajo la curva ROC y provee una métrica para determinar el desempeño del modelo a través de los distintos umbrales de

clasificación. Puede ser interpretada como la probabilidad de que el modelo clasifique una muestra positiva aleatoria de manera correcta [28]. Se define mediante la siguiente fórmula:

$$AUC = 1 - \frac{1}{2}\left(\frac{FP}{FP+TN} + \frac{FN}{FN+TP}\right)$$

Fórmula 3: Área bajo la curva. Tomado de [27].

- ***F1 score*:** El puntaje F1 o medida F1 puede ser interpretada como la media armónica de las métricas *precision* y *recall*, ya que pretende igualar la contribución de cada una. Se encuentra acotada en el rango entre *0* (peor valor) y *1* (mejor valor) [29]. La fórmula que la define es la siguiente:

$$F1 = \frac{2 \times (precision \times recall)}{precision + recall}$$

Fórmula 4: F1 score. Tomado de [27].

- ***Accuracy*:** Es la tasa de predicciones correctas realizadas por el modelo sobre el *dataset*. Se estima utilizando un *dataset* independiente para pruebas, y que no fue usado en ningún momento durante el entrenamiento [25]. La fórmula que la define es la siguiente:

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

Fórmula 5: Accuracy. Tomado de [30].

- ***Kruskal-Wallis H-test*:** Se trata de una prueba de dos hipótesis, llamadas nula y alternativa que busca determinar si la mediana de los grupos observados es igual, se puede realizar sobre dos o más muestras independientes [31]. Se encuentra dada por la siguiente fórmula:

$$H = \left(\frac{12}{N \times (N+1)} \times \sum_{j=1}^{k} \frac{R_j^2}{n_j}\right) - 3 \times (N+1)$$

Fórmula 6: Kruskal-Wallis H-test. Tomado de [31].

Donde:

- k: El número de grupos comparados.
- N: El tamaño total de la muestra.
- n_j: El tamaño de la muestra en el jotaésimo grupo.
- R_j: Es la suma de los rangos en el jotaésimo grupo.

- ***Chi-square test*:** Es una prueba estadística para datos categóricos no paramétricos que permite determinar si estos son significativamente diferentes a lo esperado, para ello plantea dos hipótesis, llamadas nula y alternativa que busca determinar si la mediana de los grupos observados es igual, se puede realizar sobre dos o más muestras independientes [32, p. 2]. Se encuentra dada por la siguiente fórmula:

$$X^2 = \sum \frac{(O-E)^2}{E}$$

Fórmula 7: Chi-square test. Tomado de [32].

Donde:

  - O: La frecuencia observada.
  - E: La frecuencia esperada.

- ***Credit score*:** Se trata de un número que resume el historial de crédito en un reporte, que indica la probabilidad de incumplir una obligación crediticia.

- ***MLOps*:** *Machine Learning Ops* o *Machine Learning Engineering* se define como el uso de principios científicos, herramientas y técnicas de Machine Learning, y de la ingeniería de software tradicional que permite diseñar y construir sistemas de computación complejos, abarcando todas sus etapas, desde la recolección de datos, pasando por la construcción y entrenamiento del modelo, hasta desplegar el modelo para ser usado por un producto o consumidores finales [33]. Este proceso se divide en las siguientes etapas:

  - **Comprensión del negocio:** Tiene como finalidad conocer todas las reglas de negocio que van a determinar el desarrollo del proyecto. Consta de las siguientes subetapas:

    - **Entendimiento inicial:** En esta subetapa se busca comprender la perspectiva del cliente con respecto al negocio. El cliente debe brindar la información necesaria y suficiente para capturar los requerimientos y resultados esperados.

    - **Viabilidad del proyecto:** En esta subetapa se determinan los criterios de aceptación a nivel técnico y a nivel comercial. Se consolidan los indicadores clave de rendimiento o KPIs que permitan observar el estado actual del proyecto con el fin de agilizar la toma de decisiones y el diseño de estrategia.

    - **Estimar costos:** Teniendo en cuenta la información recolectada en las subetapas anteriores se procede a estimar el costo del proyecto. La estimación incluye impuestos, costo de adquisición,

almacenamiento, procesamiento de datos, recursos humanos y demás valores que repercutan en el costo final.

- **Propuesta de valor y de negocio:** Se perfilan los componentes que van a definir el factor diferencial del producto que se quiere construir. Se define a nivel técnico el tipo de aproximación que se va a tomar para construir la solución requerida.

- ***Kick-Off*:** Es una presentación donde se detallan los siguientes puntos: Objetivo del proyecto, fases del proyecto, entregables y solución propuesta, arquitectura de despliegue, análisis de datos, modelo de *Machine Learning*, equipo de trabajo, canales de comunicación, reuniones de seguimiento y consultoría.

- ***Pipeline*:** Es importante la implementación de un *Pipeline* para facilitar la creación, automatización y administración de flujos de trabajo de *Machine Learning* con el fin de agilizar el desarrollo, la iteración y despliegue de la solución.

- **Preparación de los datos:** Permite realizar el tratamiento de los datos para asegurarse que el *dataset* cumple con las normas de negocio y tiene Consta de las siguientes subetapas:

  - **Verificación de la calidad de los datos:** Permite determinar la validez del *dataset* que se va a utilizar. El objetivo principal es asegurar que los datos sean los adecuados para este proyecto y que representen la lógica del negocio. Por lo cual, se hace la verificación de datos nulos, fuera de rango, desbalance de datos, entre otras verificaciones, efectuadas normalmente usando las herramientas y módulos de cada lenguaje de programación.

  - **Ingeniería de características:** Es un proceso que permite seleccionar y transformar columnas para alimentar un modelo de *Machine Learning*. Por lo general, la ingeniería de características incluye la creación, la transformación, y la extracción de características.

    La extracción de características ayuda a identificar las columnas del *dataset* que son relevantes para el dominio del problema. Este proceso se apoya en las siguientes técnicas:

    - ***Filtering:*** Los métodos de filtrado de características mediante filtrado permiten seleccionar las variables basándose en estadísticas y modelos matemáticos. Puede ser univariado y multivariado, evaluando la relevancia de

cada variable para la predicción [34].

- ***Wrapper:*** Este método considera un subconjunto de características, donde se evalúa la calidad de cada una a través de la preparación, evaluación y combinación de variables contra otras variables. De esta forma se facilita la detección de posibles interacciones entre dichas variables, potencializando la calidad de las predicciones [34].

- ***Embedded:*** En este método, la selección de características se lleva a cabo en el proceso de entrenamiento del modelo de forma simultánea para determinar aquellas características que potencian el poder predictivo del modelo. Algunos algoritmos como *Random Forest*, incluyen este tipo de funcionalidades [34].

La transformación de características permite mitigar el impacto de datos faltantes o que no son válidos, a través de estrategias como la imputación.

La creación de características permite reducir las dimensiones del *dataset* a partir de la obtención de nuevas características a partir de las ya existentes.

- ***Foundations*:** Es un proceso que incorpora prácticas de *DevOps*. Permite tener control sobre las versiones del código, mantener trazabilidad sobre el rendimiento del sistema y hacer integraciones y entregas de código de manera continua.

- **Control de versiones de datos:** Es una forma de mantener trazabilidad sobre los cambios que sufren los datos y los modelos entrenados a partir de estos.

- **Análisis de datos:** Busca comprender la tendencia que tienen los datos para extraer de ellos la información de valor para el negocio. Consta de las siguientes subetapas:

  - **Observación:** Se analiza el *dataset* para conocer su tendencia estadística.

  - **Hipótesis:** En este proceso se plantean diferentes hipótesis que permitirán comprobar si las variables en el análisis son lo suficientemente dicientes para hacer la predicción.

  - ***Insights*:** A partir de la recolección de datos anterior, se extrae la información que resulta útil para la toma de decisiones

correspondientes sobre los datos que van a alimentar el modelo.

- **Modelado:** En esta etapa, se realiza el entrenamiento y la validación de los modelos candidatos, con el fin de escoger el más idóneo para manejar el problema presentado. Se compone de las siguientes subetapas:

  - **Entrenamiento:** Para este proceso se divide el *dataset* en dos partes para entrenar el modelo y para validarlo, en algunos casos la proporción suele ser del ***70%*** para el entrenamiento y del ***30%*** para validación. Inicialmente se busca seleccionar un algoritmo de *Machine Learning* que represente de mejor manera las aspiraciones del negocio respecto al problema a resolver [35].

  - **Validación:** Una vez entrenado, el modelo se valida contra el *dataset* de validación el cual permite determinar el desempeño del modelo sobre datos desconocidos, que no fueron visibles durante el entrenamiento. Como fruto del proceso de validación es usual obtener métricas de desempeño como *precision* y *recall* para el caso de los clasificadores. Estas métricas son algunos de los criterios que son tenidos en cuenta para escoger el modelo adecuado para el problema en cuestión [36].

  - **Pruebas:** Finalmente, se realizan pruebas sobre distintas variantes del modelo o modelos entrenados para determinar la mejor aproximación posible para dicho problema [37].

- **Despliegue:** Es un proceso que involucra llevar un modelo de *Machine Learning* a un entorno de producción con la finalidad de hacerlo consumible por los usuarios finales, desarrolladores, sistemas o aplicaciones para facilitar la toma de decisiones a nivel de negocio tomando en cuenta unos datos de entrada. Esta es una de las etapas más críticas de todo el *pipeline* de *MLOps* y de *Machine Learning* en general debido a que entre el ***60%*** y el ***90%*** de los modelos no consiguen ser desplegados [20].

2.2. Entorno de servicios web

Adicionalmente, se desglosan algunos de los servicios web ofrecidos por el proveedor AWS, que serán utilizados para el desarrollo de este proyecto:

- ***AWS CodeCommit*:** Es el servicio de control de versiones de AWS, que se puede utilizar para almacenar y gestionar documentos, código fuente y archivos binarios en la nube. Para el caso de proyectos que hacen uso de *MLOps* se utiliza para mantener trazabilidad sobre los cambios en un *pipeline*.

- ***AWS CodePipeline*:** Es un servicio que se usa para modelar, visualizar y automatizar los pasos requeridos para el lanzamiento de un *software*. Permite realizar un modelado rápido y configurar los diferentes ambientes del proceso de lanzamiento. Para el caso de proyectos que hacen uso de *MLOps* se utiliza para automatizar el proceso de preprocesamiento, entrenamiento y despliegue.

- ***Amazon S3*:** Se ofrece como un servicio escalable, seguro, rentable y de alta disponibilidad que está equipado para organizar y administrar grandes volúmenes de datos. Para casos de uso de *Machine Learning*, se utiliza para almacenar *datasets* y artefactos.

- ***Amazon SageMaker*:** Es un servicio completamente gestionado para *Machine Learning* que cuenta con herramientas para construir, entrenar y desplegar modelos al entorno de producción. Cuenta con una funcionalidad para trabajar *notebooks* de Jupyter, con módulos precargados, para tareas como exploración y análisis [38].

  Algunas de las herramientas de este servicio que fueron utilizadas para el desarrollo de este proyecto son:

  - ***Amazon SageMaker Data Wrangler*:** Es una herramienta que contribuye a disminuir el tiempo de recolección y preparación de datos para proyectos de *Machine Learning*. Cuenta con diversos módulos que simplifican los procesos de preparación de datos, selección de características, limpieza, exploración y visualización de datos a nivel estadístico. Adicionalmente, permite importar y exportar *datasets* modificados a través de flujos de datos y ejecutar análisis sobre el *dataset* [39].

- ***AWS CloudFormation*:** Es un servicio que permite el modelado y la configuración de recursos de AWS para reducir el tiempo de gestión. A través de plantillas se puede realizar una descripción de los recursos de AWS requeridos para ser automáticamente provisionados a través de *Cloudformation* [40].

2.3. Trabajos relacionados

Para tener acceso a los trabajos relacionados con este proyecto, se realizó un análisis cienciométrico utilizando *ScientoPy* [41], sobre los resultados de una búsqueda en las bases de datos de Scopus y Web of Science, aplicando la siguiente cadena de búsqueda:

"credit score" AND ("machine learning" OR "deep learning")

Una vez realizado el preprocesado de los resultados obtenidos usando *ScientoPy*, se buscó identificar las técnicas para el procesamiento de datos y los criterios de

evaluación de riesgo más relevantes dentro del contexto de implementación de este piloto. Para ello, se describirán los resultados obtenidos con la consulta y posteriormente se buscará responder las siguientes preguntas clave:

- ¿Cuáles son los tipos de algoritmos de clasificación aplicados en este contexto?
- ¿Cómo medir el riesgo de un préstamo?

Las respuestas a las preguntas planteadas en el análisis previamente mencionado son las siguientes:

- **¿Cuáles son los tipos de algoritmos de clasificación aplicados en este contexto?:**

  Utilizando *ScientoPy*, se obtiene la lista de las palabras clave por autor dentro de los resultados de búsqueda, como se muestra en la siguiente gráfica:

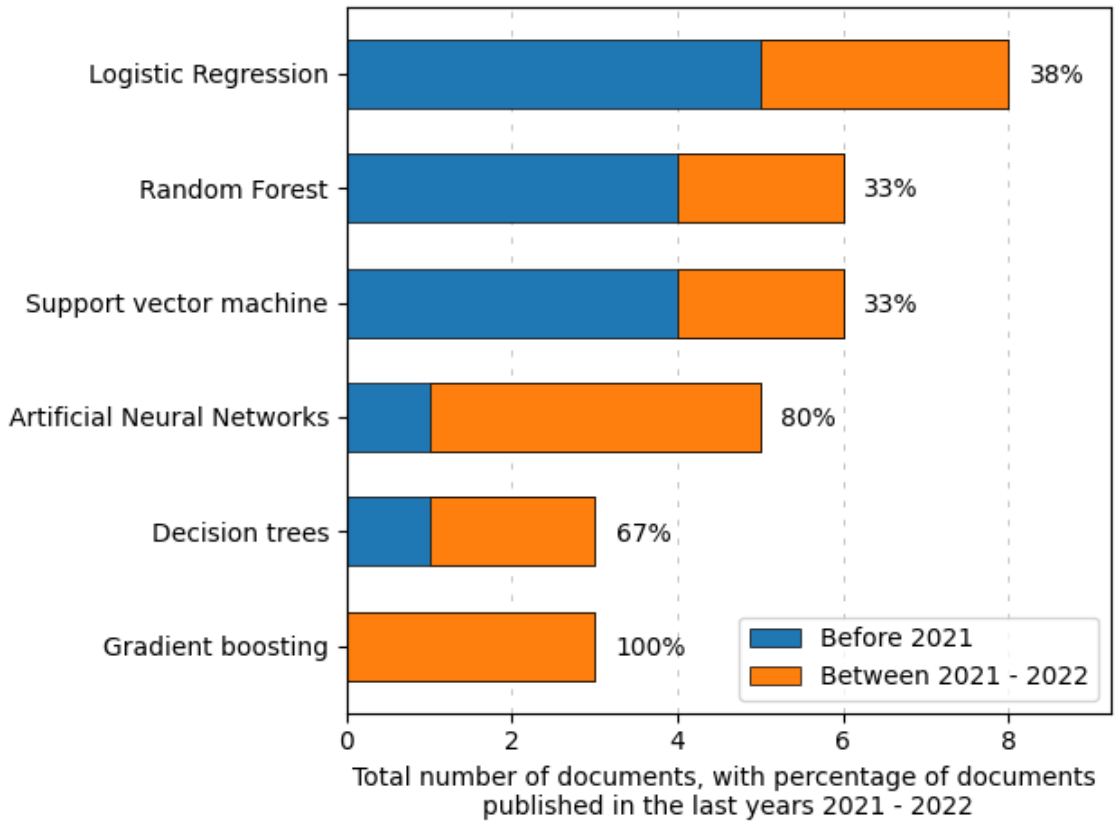


Figura 2: Resultados por tema del análisis cienciométrico de la búsqueda realizada. Fuente propia.

A partir de la figura 2, se puede concluir que las investigaciones en esta área se han incrementado en años recientes. En la figura 3, se observa cual ha sido su crecimiento:

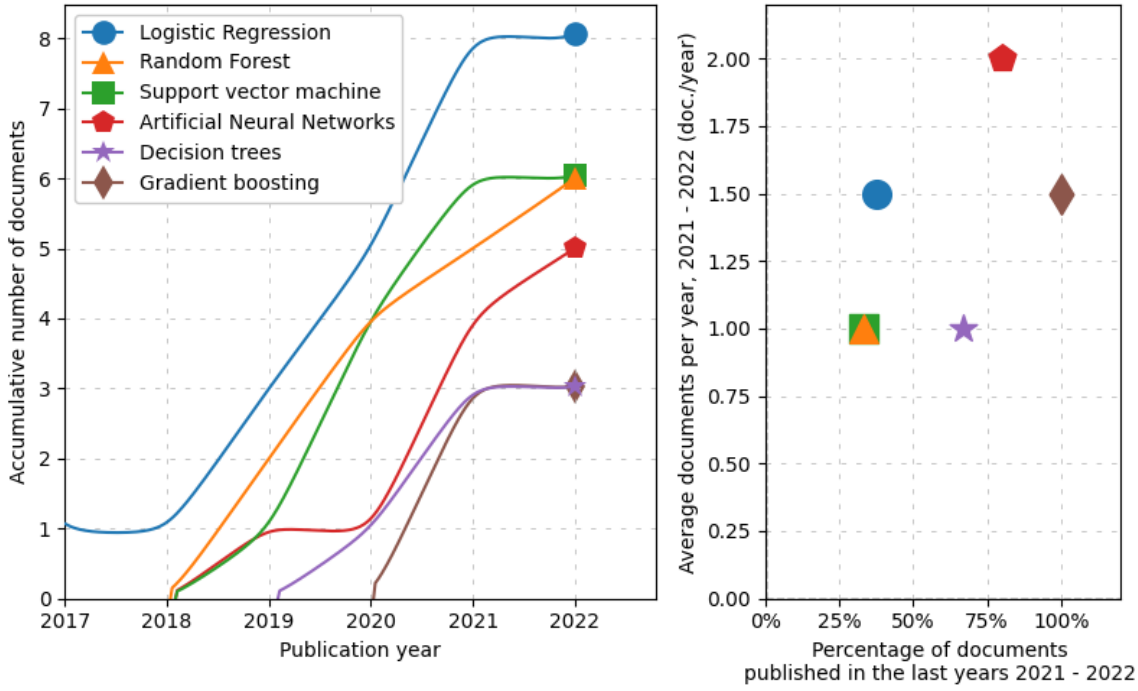


Figura 3: Tendencia de las investigaciones de la búsqueda realizada. Fuente propia.

Sobre los tópicos clave que están en tendencia de crecimiento según la figura 3, es posible listar conceptos y palabras clave que serán importantes para el desarrollo de este trabajo: *Artificial Neural Networks*, *Logistic Regression*, *Random Forest*, *Support Vector Machine*, *Decision Trees* y *Gradient Boosting*.

- **¿Cómo medir el riesgo de un préstamo?:**

  Refiriéndose de nuevo al gráfico de la figura 1, se identifica que *Credit score*, *Credit risk* y *Credit scoring* son los términos que, independientemente de los criterios utilizados, buscan evaluar el riesgo de incumplimiento de los pagos. Finalmente, dichos términos fueron agrupados bajo *Credit scoring*.

Teniendo en cuenta la agrupación de términos realizada previamente, se procedió a utilizar la funcionalidad de análisis extendido de *ScientoPy*, con el fin de acotar los trabajos relacionados más relevantes para la implementación de este piloto.

Los autores de [42] construyeron un modelo, conocido como CPLE-LightGBM, alrededor del *dataset* de transacciones rechazadas de Lending Club, esta propuesta busca reducir el sesgo que presentan los modelos construídos alrededor del *dataset* de transacciones aceptadas. Este modelo es un modelo clasificador compuesto por el *framework* CPLE, para aprendizaje semi-supervisado y LightGBM, que es un *framework* inspirado en *Gradient Based DT*.

Los resultados obtenidos en el entrenamiento y en producción demostraron una reducción del sesgo y un aumento significativo en el desempeño del modelo, cuando la tasa de rechazo de préstamos es baja. No obstante, la falta de información de costo en el modelo significó una dificultad a la hora de escoger un modelo óptimo desde el punto de vista operativo. Cabe resaltar que el *dataset* en torno al que se construyó el modelo, a diferencia del *dataset* de transacciones aprobadas, no hace referencia al historial de crédito de las personas. En cuanto al trabajo futuro, los autores proponen estimar el costo para los distintos tipos de error, la factibilidad de utilizar técnicas de filtrado de datos sobre los *datasets* de préstamos rechazados. Además, proponen validar el comportamiento de este modelo en otros *datasets* y finalmente evaluar otros métodos de aprendizaje semi-supervisado para la inferencia de rechazo.

Los autores de [43] buscaron determinar la aproximación más precisa para predecir la puntuación crediticia de una persona, inicialmente a través de la identificación y selección de las *features* más relevantes en dicha evaluación. Adicionalmente, los autores compararon tres técnicas de selección de *features* conocidas como *Chi-square*, *Information gain* y *Gain-ratio*, complementandose con cinco tipos de clasificadores: *Bayesian Optimization*, *Naïve Bayes*, SVM, DT, y RF.

Los resultados de esta investigación indicaron que *Chi-square* y RF son los dos tipos de técnicas más apropiadas para reducir incidencias de tipo falso positivo y falso negativo. El uso complementario de estas técnicas promete ser una alternativa robusta para construir un algoritmo de predicción de puntuación de crédito (con un ***93%*** de precisión), a pesar de que los tiempos de entrenamiento para este tipo de clasificador sean relativamente altos en comparación a las demás alternativas presentadas. Cabe resaltar que los autores realizaron su implementación utilizando el *dataset* de crédito alemán. Finalmente los autores proponen a modo de trabajo futuro utilizar otros *datasets*, en conjunto con otros métodos de clasificación.

El autor de [44] propone un análisis comparativo de modelos compuestos para la predicción de la puntuación crediticia, utilizando el *dataset* de crédito australiano. A partir de su revisión preliminar, el autor destaca un conjunto de modelos clasificadores base, que son parte de los modelos compuestos, que serán comparados, entre los que se encuentran: *Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbour, Multilayer Perceptron y Naïve Bayes*. Por otra parte, los modelos compuestos son los siguientes: *Random Forest, Bagged Decision Tree, ExtraTreesClassifier, Adaboosting, Gradient Boosting* y *Voting Ensemble* (Compuesto por LR, DT y SVM).

Los resultados obtenidos muestran que SVM es el modelo clasificador base con el mejor desempeño obteniendo una precisión del ***87.68%***, mientras que *Random Forest* y *Extra Trees Classifier*, exhibieron una precisión del ***88.41%*** al momento de determinar la puntuación crediticia de un determinado usuario. Finalmente, el autor propone como trabajo futuro estudiar y optimizar los tiempos y costos computacionales requeridos por

los modelos.

[45] hace uso de tres *datasets* provistos por entidades bancarias Brasileñas, para entrenar algoritmos de *Machine Learning* que permiten determinar la elegibilidad de un determinado usuario para recibir un préstamo.

Los autores construyeron 5 modelos de *Machine Learning*, catalogados en dos grupos, según la manera en la que los datos son adquiridos, para este caso se trata de *batch learning* y *stream learning*. Dichos métodos buscan combinarse con el fin de aumentar la robustez de las predicciones.

Para *batch learning*, se construyeron tres algoritmos: LR, RF y XGBoost (esquema de aprendizaje secuencial que busca minimizar una función de pérdida, evitando el *overfitting* o sobreajuste). En cuanto al *stream learning*, se construyeron dos algoritmos: *Leveraging Bagging* y *Adaptive Random Forest* (RF entrenado usando *Bagging*, que posee mecanismos para adaptarse a los cambios en los datos).

Los resultados demostraron que la combinación de *batch* y *stream learning* permite aumentar la confiabilidad de las predicciones y la capacidad de respuesta al permitir que el entrenamiento se pueda realizar en la medida en que los datos son recolectados. Los autores no proponen trabajos futuros.

[46] propone un modelo conocido como GCSSE (*GMDH-based Cost-sensitive Semi-supervised Selective Ensemble*), diseñado para abarcar problemas específicos como el número reducido de muestras etiquetadas y distribuciones de clases desbalanceadas.

El funcionamiento de este modelo está compuesto por dos etapas: (1) las muestras de datos no etiquetadas, se etiquetan selectivamente y se entrenan una serie de clasificadores base, y (2) una red neuronal GMDH obtiene un arreglo de clasificaciones. Este modelo se probó con 5 *datasets* de dominio público (*uk-thomas*, *german*, *australian*, PAKDD y *give-credit*).

Los resultados demuestran que este modelo tiene un mayor desempeño para evaluar la calificación crediticia de una persona, comparado con otros modelos similares, aunque el uso de aprendizaje semi-supervisado corre el riesgo de introducir ruido debido al etiquetado incorrecto de los datos. Los autores proponen el uso de aprendizaje semi supervisado activo con el fin de mitigar el efecto del etiquetado erróneo de los datos.

[47] buscó establecer cómo distintos factores afectan la capacidad de pago de los usuarios, a través de un modelo construído alrededor del *dataset* Home Credit, una organización no bancaria de República Checa.

Utilizando regresión logística, el autor logró establecer que existe un fuerte vínculo entre la información demográfica de los usuarios (por ejemplo la edad, el sexo y el nivel

educativo) y la probabilidad de pagar los préstamos por completo. El modelo obtuvo una precisión del ***82%***. El autor no propone trabajos futuros.

[48] indica que los procesos tradicionales de diseño de redes neuronales suelen llevarse a cabo empíricamente teniendo en cuenta que los valores de los hiper parámetros y las arquitecturas de dichas redes suelen ser determinados mediante una búsqueda exhaustiva por todos los posibles valores siendo ineficiente y costoso a nivel técnico y de *hardware*. Para resolver dichas limitaciones, los autores proponen la implementación de un *framework* que hace uso de los siguientes algoritmos de optimización: *Gradient Boosted Regression Trees*, *Uniform Sampling*, *Gradient Descent*, *Bayesian Optimization* y *Decision Trees*. Dicho *framework* además busca resolver problemas de desbalance entre las clases de datos en el *dataset* de crédito alemán con el fin de favorecer la precisión y el desempeño de la red neuronal. La implementación resultante está compuesta por 21 capas. Los resultados concluyen que utilizar cualquiera de los ya mencionados algoritmos de optimización tiene un impacto positivo en la predicción, no obstante, la optimización Bayesiana es la que posee el menor margen de error. Pese a los excelentes resultados obtenidos por la investigación, se limita al estudio de un solo conjunto de datos de unas determinadas características mientras que no se exploran fuentes de datos alternativas.

Los autores de [49] manifiestan que hoy en día, la estimación del puntaje crediticio de las personas es una de las tareas más importantes para las entidades financieras, no obstante, la implementación de los modelos que permiten realizar estas predicciones puede tomar de 3 a 18 meses, por lo cual, muchas de estas entidades utilizan los mismos modelos a través de los años, desconociendo la naturaleza dinámica de la economía y de la población, comprometiendo la precisión y la confiabilidad de los sistemas.

Con el fin de determinar la mejor alternativa para la implementación de estos modelos, los autores presentan dos grandes grupos de técnicas de aprendizaje: Aprendizaje por lotes de datos y aprendizaje por corrientes de datos.

Para el caso del aprendizaje por lotes de datos, se contemplan 4 tipos de modelos de datos: *Logistic Regression, J48* (Un tipo de *Decision Tree*), *Naïve Bayes* y *Random Forest*. En cuanto al aprendizaje por corrientes de datos se tienen: *Hoeffding Tree* (Un tipo de *Decision Tree* que limita la división de hojas al límite de Hoeffding), *Hoeffding Adaptive Tree* (Es una extensión de *Hoeffding Tree*, donde la división de hojas también está sujeta a las tasas de error), *Leveraging Bagging* y *Adaptive Random Forest*. Estos modelos fueron construídos alrededor de tres *datasets* provistos por entidades financieras brasileñas.

Las pruebas sobre dichos modelos demostraron que la puntuación crediticia debe ser asumida como un problema de clasificación por corriente de datos, ya que las variables utilizadas en el análisis pueden desfasarse con el tiempo.

[50] explora diversos algoritmos de clasificación (LR, KNN, SVM y RF), basándose en el *dataset* de Lending Club.

Los autores utilizan las siguientes *features* meta normalizadas, basadas en los datos del usuario con el fin de aumentar la confiabilidad y la robustez de las predicciones:

- ***DTI (Debt-to-Income Ratio):*** La relación del total de la deuda mensual de una persona con respecto a sus ingresos.

- ***Income-to-Payment Ratio*****:** Los pagos mensuales del préstamo con relación a los ingresos totales.

- ***Revolving Utilization Rate:*** La cantidad de créditos de una persona, relativo a todos los créditos rotativos disponibles.

- ***Revolving to Income Ratio:*** Relación de balance de crédito rotativo respecto al ingreso mensual de una persona.

Según las pruebas realizadas a cada modelo, se concluye que RF es el método más indicado para este tipo de aplicaciones debido a su robustez para identificar a los potenciales buenos usuarios sin embargo, cabe la posibilidad de que RF realice una evaluación errónea de los buenos clientes. Los autores proponen utilizar datos sociales de los usuarios para contribuir en la evaluación del puntaje crediticio, adicionalmente, proponen utilizar SVM con hiper parámetros afinados como una alternativa a RF.

[51] propone un *framework* dinámico de aprendizaje reforzado que corrige las anomalías de los datos recolectados para adaptarse al entorno real, basándose en la retroalimentación en tiempo real en lugar de hacer supuestos teóricos.

Este modelo de *Reinforcement Learning* está compuesto de dos partes: el entorno y el agente, que interactúan entre sí compartiendo estados, acciones y objetos de recompensa con el fin de optimizar el umbral de aceptación de las solicitudes de préstamo.

Los resultados demuestran que la aproximación utilizada tiene un alto desempeño en entornos simulados y del mundo real. Los autores señalan que el trabajo futuro se puede encaminar a la experimentación con técnicas de inferencia de rechazo y hacia algoritmos que utilicen, por ejemplo, métodos basados en gradiente. Adicionalmente, indican que la adaptabilidad y confiabilidad del modelo puede ser incrementada a través de la detección de fraudes.

Por su lado los autores de [52] proponen un sistema de clasificación compuesto denominado *One-class Classification Driven Dynamical Ensemble Learning* (OCDDEL) el cual hace uso de un clasificador de una sola clase que ayuda a conocer aquellos subconjuntos de datos de pruebas que tienen distribuciones similares al conjunto de datos de pruebas. Cuando exista coincidencia, es decir, cuando las distribuciones sean

similares, el valor de prueba será evaluado por un clasificador con mayor capacidad de ajuste ya que está enfocado sólo en aquellas muestras con distribuciones similares. En contraste, cuando no existe una coincidencia, el valor de prueba pasa a ser evaluado por un clasificador con mayor capacidad de generalización. Cabe mencionar que este proyecto hace uso del *dataset* de *Lending Club*.

Aunque se evidencia un comportamiento dinámico por parte del clasificador, según el caso de prueba, y un desempeño superior en comparación con sus competidores, esta aproximación no toma en cuenta posibles cambios en cuanto al negocio y los datos.

Los autores de [53] proponen una aproximación interesante, haciendo uso de datos sociales de redes como *Twitter* y *LinkedIn*, y de servicios de pago como *Google Pay* y *Apple Pay* con el fin de predecir la probabilidad de impago. Realizan pruebas sobre distintos modelos de *Machine Learning*, entre los cuales están *Logistic Regression*, *SVM*, *Naïve Bayes*, *Decision Tree*, *Random Forest* y *KNN*. Los resultados arrojan un *Accuracy* de hasta ***70.73%***. No obstante, debido a las políticas de plataformas como *Apple Pay*, se requeriría de un tiempo extenso para la captura de datos.

Los autores de [54] presentan una aproximación que no depende del historial de crédito del aplicante, enfocándose en su lugar en otros datos como las redes sociales del solicitante, el valor de su ganado y los ingresos derivados de esa actividad, la edad del solicitante entre otros factores. Hacen uso de *LASSO* y *Random Forest* para predecir la posibilidad de impago a partir de los elementos mencionados anteriormente. Aunque los resultados ofrecen un *Accuracy* cercano al ***70%***, el enfoque de esta aproximación está pensado para un sector rural en Bangladesh, distanciándose del contexto de aplicación de este trabajo de grado.

Los autores de [55], plantean una evaluación comparativa en función del desempeño y evalúa las métricas *AUC*, *Precision* y *Recall* entre tres técnicas de *ML*: *Logistic Regression*, *Random Forest* y *MultiLayer Perceptron*. Estos modelos fueron alimentados por el *dataset* de *Lending Club* en cuatro escenarios diferentes: Sin *resampling*, *undersampling*, *oversampling* y con un método híbrido. Los autores evalúan el desempeño de las diferentes combinaciones entre técnicas de *resampling* y modelos de *ML* haciendo uso de unas herramientas denominadas *eXplainable Artificial Intelligence (XAI)* lo que demostró que *Random Forest con undersampling* es la combinación con la mayor precisión (***90.7%***).

Este trabajo es particularmente valioso ya que explora técnicas de transformación del *dataset* y del procesamiento de los datos que presenta de manera clara el impacto que estas tienen sobre el desempeño final, particularmente en aplicaciones de *ML* en el área financiera. Por otro lado, no explora elementos de monitoreo y mantenimiento del modelo una vez este se encuentre desplegado por lo que se desconoce el nivel de adaptación y resiliencia ante el cambio que tengan dichas aproximaciones en producción.

Los autores de [56] plantean una aproximación en la que se hace uso de la técnica *Kruskal-Wallis* como método para determinar el peso que aporta cada una de las características del *dataset* de *Lending Club* y del *dataset* Alemán dentro del proceso de predicción. A partir de estos resultados, una nueva columna es generada bajo el nombre de *Credit Risk Index* o CRI, de esta forma se consigue mitigar el efecto del cambio de los datos y de la lógica del negocio al momento de predecir el riesgo de impago. Es importante recalcar que a pesar de proveer una aproximación resiliente ante los cambios, esta aún es susceptible de considerar elementos que son parte del historial crediticio de los aplicantes.

Los autores de [57] proponen una implementación que hace uso de un modelo compuesto de dos etapas conocido como *Multi-Grained Cascade Forest* o *gcForest*, que consume tres *datasets*: Alemán, Australiano y Japonés. La primera etapa consiste en determinar un conjunto óptimo de características a través de dos criterios: La interpretabilidad de las características y la tasa más baja de error tipo II además de los valores *AUC* y *Accuracy* más altos, respectivamente. Esto se logra a través de la evaluación de los resultados de cinco métodos distintos: *Full-variable Logistic Regression*, *Stepwise Regression* (proceso iterativo y paso a paso de selección de características) basado en criterio *AIC* [58], *Stepwise Regression* basado en criterio *BIC* [59], *Lasso-Logistic Regression* y *Elastic Net Logistic Regression*. La segunda etapa de la implementación consiste en realizar una validación cruzada con K pliegues. Este proceso se realiza de forma iterativa hasta que se consiguen los mejores resultados de acuerdo a los criterios anteriormente mencionados. Este método ofrece una implementación robusta puesto que centra su atención en métricas que permiten fácilmente determinar la calidad del modelo adicionalmente, presenta resiliencia al cambio al aplicar un proceso iterativo sobre el modelo y los datos que lo alimentan que le permitirían adaptarse a cambios de negocio o de los datos. No obstante, la complejidad de esta implementación podría representar un reto en la escalabilidad y mantenibilidad del sistema, ya que no se define un *pipeline* automatizado para preprocesamiento, modelado y despliegue que permita minimizar los tiempos que requieren dichas tareas de manera manual, y que contribuyen a mitigar los errores humanos.

Los autores de [60], proponen la implementación conocida como *LGBBO-RuleMiner*, un algoritmo de clasificación basado en reglas que utiliza optimización basada en biogeografía. La naturaleza de este algoritmo implica la obtención de dos operadores evolucionarios que son migración y mutación. Dichos operadores, permiten la obtención de un conjunto de reglas potenciales para realizar la predicción no obstante, a partir de éstas sólo se emplea aquella que tenga el mayor poder predictivo. Los resultados por su parte, muestran que la aproximación ofrece un gran desempeño en comparación con otras implementaciones que hacen uso de los *datasets* Alemán y Australiano. Sin embargo, no se aplican técnicas para la escalabilidad y resiliencia del algoritmo para adaptarse a los cambios, además dicha aproximación no evita hacer uso de datos históricos de crédito.

Los autores de [61] plantean la construcción de un modelo de clasificación que utiliza *K-Nearest Neighbors* o *KNN*. Este modelo es alimentado por el *dataset* Australiano, dividido en dos partes para entrenamiento y pruebas y es ajustado por cuatro valores de *k*. Los resultados de los ejercicios de pruebas muestran un buen desempeño por parte del modelo. Sin embargo, este trabajo no presenta alternativas de modelos distintas a la ya mencionada, ni exhibe un tratamiento preliminar sobre el *dataset* que permita aumentar la capacidad predictiva como por ejemplo selección o ingeniería de características. Este trabajo se apoya sobre los datos del historial de crédito de las personas.

Los autores de [62] proponen una aproximación interesante para la predicción del riesgo de impago basándose en datos demográficos, de la frecuencia de uso de correo electrónico y psicométricos, en contextos de economías en desarrollo como es el caso de México y Nigeria. Teniendo en cuenta que estos datos provienen de fuentes diversas, los autores determinaron necesario dividir dichos datos en dos arreglos denominados A y B. En el caso de A se incluyen tres *datasets* los cuales contienen información de 12 variables demográficas, 350 de carácter psicométrico y 53 variables alternativas, consistente en la información sobre el uso de correo electrónico. Para el caso de B, a diferencia del caso A, se puede encontrar la variable objetivo, indicando el impago del crédito. Estos conjuntos de datos van a alimentar distintos tipos de modelos listados a continuación para el caso A: *Logistic Regression*, *LASSO*, *Ridge Regression*, *XGBoost*, *PCA + Logistic Regression*, *PCA + XGBoost*, *PCA + Ridge Regression* y *PCA + LASSO*, siendo este último el que exhibe mejores resultados. Por parte de B se tienen: *Logistic Regression*, *LASSO*, *Ridge Regression*, *XGBoost*, *Oversampling + XGBoost*, *Oversampling + XGBoost*, *Neural Networks*, *PCA + Logistic Regression*, *PCA + LASSO*, *PCA + Ridge Regression*, *PCA + XGBoost* y *PCA + Neural Networks*. A pesar de que los resultados exhibidos en el caso A y en el caso B demuestran un gran desempeño y que es verdaderamente factible realizar una predicción confiable a partir de los datos alternativos, la consecución de estos datos es un bloqueante significativo que no se puede obviar ya que estos encuentran barreras de privacidad, tiempo de recolección y disponibilidad, lo cual no permite que se configure como una alternativa viable.

Los autores de [63] plantean un análisis experimental de varias aproximaciones que utilizan *Machine Learning*, comprendiendo alrededor de 9 métodos de selección de características (*ILFS* o *Infinite Latent Feature Selection, ECFS* o *Eigenvector Centrality, Relief, FSV* o *Feature Selection via Concave Minimization, LS* o *Laplacian Score, MFCS* o *Multi-Cluster-based Feature Selection, UDFS* o *Unsupervised Discriminative Feature Selection, LLCFS* o *Local Learning-based Clustering* y *CFS* o *Coefficients-based Feature Ranking*) y 16 algoritmos de clasificación (*LDA* o *Linear Discriminant Analysis, NB* o Naive Bayes*, TDNN* o *Time Delay Neural Network, KNN* o *K-Nearest Neighbours, DT* o *Decision Tree, ELM* o *Extreme Learning Machine, ELM-1, ELM-2, RF* o *Random Forest, PART* o *Partial Decision Tree, MLP* o *Multi-layer Perceptron, RBFN* o *Radial Basis Function Neural Network, SVM-R* o *Support Vector Machine, SVM-P, LRA* y *SMO*), además de los *datasets* Australiano, Alemán, Japonés, Taiwanés, *Bankruptcy* y *Bank Direct Marketing*. Los resultados de este trabajo se

obtienen a partir de diferentes combinaciones entre métodos de selección de características, modelos y fuentes de datos. De acuerdo con lo hallado por los autores, la aproximación con *UDFS* y *TDNN* posee los mejores resultados, a pesar de esto, los *datasets* estudiados contienen información del historial de crédito.

Los autores de [64] proponen un algoritmo de clasificación que hace uso de redes neuronales artificiales o *ANNs* en combinación con algoritmos genéticos para la selección de características sobre los *datasets* Alemán y Australiano. Esta aproximación hace uso de un criterio conocido como *Information Complexity Criterion* o *ICOMP* [65]–[68] el cual penaliza la complejidad de la covarianza de un modelo. La metodología se puede sintetizar de la siguiente manera:

1. Se genera una primera generación de datos a partir de un subconjunto de vectores.

2. Se entrenan las redes neuronales para cada subconjunto de población usando un algoritmo basado en gradientes y se asignan los puntajes estimados usando *ICOMP*.

3. Se escoge el subconjunto de datos más favorable a través de métodos heurísticos o estocásticos.

4. Se aplican métodos propios de los algoritmos genéticos como la mutación o la migración para obtener nuevos subconjuntos de datos conocidos como siguiente generación.

5. En caso de no cumplirse el criterio de detención del algoritmo, se vuelve al paso 3, de lo contrario se finaliza el proceso.

A pesar de que los resultados obtenidos por este trabajo demuestran su potencial, el uso de algoritmos genéticos para la selección de características, además del uso de redes neuronales añade un componente de complejidad técnica al momento de realizar ajustes como la inclusión de conocimiento proveniente del dominio del problema sobre el algoritmo, como lo proponen los mismos autores.

Los autores de [69] proponen un esquema basado en *Deep Learning* para facilitar el proceso de préstamos en una red *Blockchain* conocido como *KiRTi*, apoyado por una red neuronal tipo *Long Short Memory Term* que permite obtener los datos a partir de la red *Blockchain*. Este tipo de red neuronal es alimentado por el *dataset* Alemán. El mayor aporte de este proyecto es la implementación de la red *Blockchain* para incrementar la fluidez y la transparencia en el proceso de la realización de préstamos. No obstante, existe aún alta dependencia de los datos de historial de crédito dada la fuente utilizada.

Los autores de [70] proponen un análisis comparativo usando un *dataset* conocido como *VietCredit*, recolectado en Vietnam, que alimenta diversos modelos como *Logistic*

Regression, *Linear Discriminant Analysis*, *K-Nearest Neighbors*, *Decision Tree*, *Naïve Bayes* y *Support Vector Machine;* y algunos algoritmos compuestos como *AdaBoost*, *Gradient Boosting*, *LGBMClassifier*, *Random Forests* y *Extra Trees*. Finalmente, se comprueba que el método que ofrece el mejor desempeño es *LGBMClassifier*. Este trabajo es particularmente interesante por cuanto el *dataset* de *VietCredit* ofrece diferentes variedades de datos sobre los aplicantes, como lo son los datos demográficos, industria en la que se desempeña, ocupación, entre otros. A pesar de hacer uso de información distinta a la del historial de crédito y de ser de alto interés, este *dataset* no se encuentra disponible para descarga.

Los autores de [71] proponen un análisis comparativo entre dos algoritmos clasificadores que son *Decision Tree* y *Random Forest*, para ello se realiza un *EDA* sobre el *dataset* de *Lending Club*, a partir del cual son transformadas o eliminadas algunas columnas. Finalmente, los autores comprueban que el modelo con el mejor desempeño fue *Random Forest*. Teniendo en cuenta la dimensionalidad del *dataset* utilizado no se aplica ninguna técnica de selección de características, ni se realizan pruebas o despliegues posteriores.

Los autores de [72] proponen el uso de diversos algoritmos para analizar un *dataset* conocido como *Kalapa Credit Score*, propuesto por una entidad financiera de Vietnam, como parte de una competencia. Los datos presentan una variedad de columnas que incluyen información demográfica, sobre el préstamo y sobre el historial de crédito. Algunos de los algoritmos empleados para clasificar las muestras son *LightGBM*, *CatBoost* y *Random Forest*, como modelos compuestos, y *Support Vector Machine* y *Logistic Regression* como modelos únicos. Para nutrir estos algoritmos, los datos son tratados usando una técnica de selección de características conocida como *Target Permutation* [73]. De acuerdo a los resultados, el mejor desempeño lo obtuvo *Random Forest*. A pesar de poseer una variedad interesante de datos, este *dataset* no se encuentra disponible.

Los autores de [74] proponen el uso del algoritmo *CatBoost* para analizar un *dataset* conocido como *Indian loan default dataset*, el cual debe ser abordado utilizando técnicas de reconocimiento de caracteres para obtener los datos del préstamo. Los datos, a pesar de no estar disponibles para descarga, presentan una variedad de columnas que incluyen información demográfica, sobre el préstamo y sobre el historial de crédito. Los resultados demuestran un desempeño mayor de este algoritmo frente a *Gradient Boost* o *Random Forest*. A pesar de contener información valiosa para el análisis, no se encuentra disponible la fuente de datos.

Finalmente, en la tabla 1 se presenta lo expuesto anteriormente en relación con las brechas de conocimiento y los aportes de cada trabajo:

Ref.	Brechas	Aportes	Observaciones
[42]	El proceso de *resampling* realizado sobre el *dataset*, que	Propone un modelo basado en aprendizaje semi-supervisado	Utiliza los *datasets* de *Lending Club* y de

	ayuda a realizar la inferencia de rechazo, se hace de manera aleatoria; lo cual puede contribuir a la introducción de sesgo sobre el modelo entrenado.	conocido como *CPLE-LightGBM*, que hace uso de registros de préstamos aprobados y rechazados para inferir el rechazo y obtener predicciones más precisas, con un mayor rendimiento además de ser transversales al caso de uso por haber sido probado con dos *datasets* distintos.	*we.com*. Este último no está disponible para descarga.
[43]	El modelo fue construído en torno a un solo *dataset* y no se explora su desempeño con las diversas alternativas disponibles.	Utiliza el algoritmo de selección de atributos *Chi-square* en combinación con el algoritmo de clasificación *Random Forest*, para ofrecer una precisión hasta del ***93%*** y una notable reducción de falsos positivos y falsos negativos.	Utiliza el *dataset* de crédito alemán. Ofrece uno de los resultados más notables entre los trabajos estudiados.
[44]	Se limita al estudio de un solo *dataset*, que contiene datos históricos. El modelo puede ser optimizado mediante algoritmos de selección de atributos que incrementen el desempeño y la precisión.	Se realiza un análisis comparativo de algoritmos de clasificación base y compuestos. SVM resultó ser el mejor clasificador base en términos de precisión, para el caso de los clasificadores compuestos se obtuvo un gran desempeño en *Random Forest*, *Extra Tree Classifier* y *Bagged Decision Tree*.	Utiliza el *dataset* de crédito australiano.
[45]	Existen aún limitaciones a nivel técnico para la implementación de la aproximación propuesta, como la disponibilidad de etiquetas para algunos datos necesarios para la predicción.	Combina el aprendizaje por lotes de datos con el aprendizaje por corrientes de datos con el fin de ofrecer predicciones más precisas sobre el comportamiento de los clientes.	Utiliza dos *datasets* obtenidos de instituciones financieras de Brasil, cuyo uso está restringido al sector académico.
[46]	Existe una probabilidad de introducir ruido en el etiquetado de algunas muestras, llevando a predicciones erróneas.	Implementa un modelo llamado GSSE para realizar predicciones más precisas en casos donde los datos etiquetados son menores en relación al *dataset*, basándose en el costo para obtener dicha predicción.	Los resultados obtenidos sobre cinco *datasets* muestran un alto desempeño en cada uno.
[47]	Hace uso de la regresión logística para el análisis de los datos. Se puede extender sobre otros tipos de algoritmos de clasificación y de selección de atributos.	Señala que existen atributos relevantes para determinar si un cliente cumplirá con el pago de sus obligaciones, entre los que se encuentran el número de cuotas, la edad, los días que lleva	El dataset usado es de *Home Credit*.

		trabajando y su patrimonio entre otros.	
[48]	El modelo resultante podría ser extrapolado a otras fuentes de datos para evaluar su desempeño de manera transversal.	Los autores implementan un *framework* para la optimización de modelos de *Deep Learning* y logran obtener un *Miss Alarm Rate* o MAR de ***3%***.	Utiliza el *dataset* de crédito alemán.
[49]	La selección de atributos puede ser depurada porque, como señalan los autores, no existe garantía de que todos los atributos utilizados sean relevantes. Emplea atributos basados en el historial del solicitante del crédito.	Mitiga el *drift* de datos utilizando aprendizaje por corriente de datos, lo que permite que el modelo se adapte rápidamente a los cambios en el contexto de negocio.	Utiliza dos *datasets* obtenidos de instituciones financieras de Brasil, cuyo uso está restringido al sector académico.
[50]	Aunque la implementación es bastante completa en términos de exploración de métodos de selección de características y modelos, no explora *datasets* alternativos ni la forma de llevar esta implementación a un entorno de pruebas y producción.	Este estudio presenta que *Random Forest* es el algoritmo de clasificación con la precisión y el desempeño más altos entre los clasificadores analizados. Adicionalmente, hace uso de ciertos atributos de los clientes como la propiedad de la vivienda, el ingreso anual, entre otros para realizar la predicción.	Utiliza el *dataset* de *Lending Club*. Entrega una perspectiva interesante respecto a los atributos importantes al momento de entrenar modelos.
[51]	La fuente de datos utilizada para la creación del modelo no se encuentra disponible.	Utiliza el aprendizaje reforzado para optimizar constantemente el umbral de aceptación de solicitudes de crédito.	El *dataset* hacen uso de historial de crédito para sus estimaciones
[52]	La aproximación no toma en cuenta la adaptabilidad al cambio de los datos o de la lógica del negocio, lo cual hace probable que se experimente *Model Drift* o degradación del modelo.	Utiliza clasificadores especializados según los datos ingresados bajo el escenario de prueba, obteniendo mejor desempeño comparado con otras aproximaciones.	Hace uso del *dataset* de *Lending Club*.
[53]	Aunque esta aproximación es bastante interesante y además valiosa para este trabajo, el proceso de conseguir los datos a partir de las fuentes citadas requeriría de un trabajo de meses, ya que pasarelas como *Apple Pay* no exponen el historial de transacciones de un individuo, haciendo que deba ser capturada en este caso.	Se apoya en la conducta de los individuos en las redes sociales para determinar patrones de comportamiento que den indicios sobre la probabilidad de impago, desestimando así la necesidad de datos de historial crediticio.	Hace uso de datos provenientes de *LinkedIn*, *Twitter*, *Apple Pay* y *Google Pay* para predecir la probabilidad de impago.

[54]	Posee un enfoque interesante, ya que opta por no evaluar criterios de historial de crédito. Sin embargo, se enfoca a un sector de Bangladesh que es mayormente agrónomo y rural, y evalúa criterios como el valor de su ganado y los ingresos que recibe de él, entre otras variables que escapan al contexto de aplicación de este proyecto.	Se encuentra que es posible determinar el riesgo de impago a partir de la edad del aplicante, su capacidad de producir ingresos, sus ahorros, sus deudas y sus redes sociales.	Los datos provienen de un programa llamado *Targeting the Ultra-Poor* (*TUP*), administrado por *BRAC*, una organización dedicada a otorgar microcréditos. El *dataset* no se encuentra disponible.
[55]	Las aproximaciones utilizadas no consideran elementos de monitoreo del modelo en producción.	Demuestra la importancia de hacer uso de técnicas de *resampling* sobre el *dataset* y el impacto que estas tienen sobre las métricas de desempeño de la implementación, para este caso siendo *undersampling* y *Random Forest*.	Hace uso del *dataset* de *Lending Club*.
[56]	Aunque esta aproximación es resiliente ante cambios de negocio y de los datos, no obstante, aún puede considerar datos del historial de crédito.	Hace uso de la *Kruskal-Wallis* para determinar el conjunto de columnas del *dataset* que tienen más peso dentro de la predicción.	Hace uso del *dataset* de *Lending Club* y del *dataset* Alemán.
[57]	La complejidad de la implementación la hace difícil de mantener y escalar en el tiempo.	Propone una aproximación robusta frente a los cambios de datos y del negocio, gracias a una implementación de dos etapas que se enfoca en métricas que permiten establecer el nivel de calidad del modelo.	Hace uso de los *datasets* Alemán, Australiano y Japonés.
[60]	Propone una solución que no mitiga cambios en los datos y adicionalmente, hace uso del historial de crédito para realizar sus predicciones.	Aproximaciones basadas en biogeografía, como el algoritmo *LGBBO-RuleMiner*, exhiben resultados con alta calidad del modelo.	Hace uso de los *datasets* Alemán y Australiano.
[61]	Esta aproximación es muy limitada en términos del preprocesamiento realizado y de los modelos evaluados, a pesar de mostrar buenos resultados con *KNN*. Hace uso del historial de crédito.	Presenta un buen rendimiento del modelo *KNN*.	Hace uso del *dataset* Australiano.
[62]	La obtención de los datos	Este estudio confirma que es	Hace uso del *dataset*

	necesarios para la predicción representa un obstáculo en términos de tiempo de consecución, privacidad y disponibilidad lo que no configura una alternativa viable.	completamente factible realizar predicciones de impago basándose en datos que no dependen del historial de crédito de las personas.	de *Lenddo* y de instituciones financieras de México y Nigeria. Los datos no son accesibles.
[63]	No considera el despliegue a pruebas y producción del modelo implementado.	Realiza un barrido por diversas técnicas de selección de características y modelado de datos. Para las fuentes estudiadas, se demuestra que la mejor aproximación es *UDFS* + *TDNN*.	Hace uso de los *datasets* Australiano, Alemán, Japonés, Taiwanés, *Bankruptcy* y *Bank Direct Marketing*.
[64]	La complejidad del sistema propuesto hace que la integración de nuevos planteamientos o técnicas a nivel de datos o de negocio esté en cierta medida determinada por la capacidad del equipo de realizar integraciones y entregas.	La integración de métodos de selección de características usando algoritmos genéticos con redes neuronales significativamente incrementa el desempeño del modelo y la calidad de las predicciones.	Hace uso de los *datasets* Australiano y Alemán.
[69]	El alcance de la investigación se limita a la observación de un solo conjunto de datos además, no se explora el proceso de despliegue a pruebas y producción.	La implementación usando redes neuronales basadas en *LSTM* ofrecen un alto rendimiento en comparación con otros tipos de clasificadores.	Hace uso del *dataset* Alemán.
[70]	El *dataset* utilizado, a pesar de contener datos diferentes al historial de crédito no se encuentra disponible.	Esta aproximación demuestra que algoritmos de clasificación compuestos como el *LGBMClassifier* se desempeñan de manera notable en este tipo de problemas de dominio del negocio.	Hace uso del *dataset* de *VietCredit*.
[71]	No se aplican técnicas de selección de características, ni se hacen pruebas sobre el modelo desplegado.	Se comprueba que el modelo *Random Forest* ofrece un desempeño mayor a comparación del *Decision Tree*.	Hace uso del *dataset* de *Lending Club*.
[72]	El *dataset* utilizado, a pesar de contener datos diferentes al historial de crédito no se encuentra disponible.	Esta aproximación demuestra que algoritmos de clasificación compuestos como *Random Forest* se desempeñan de manera notable en este tipo de problemas de dominio del negocio.	Hace uso del *dataset* de *Kalapa Credit Score*.

[74]	A pesar de sus buenos resultados, no es posible hacer uso del *dataset* para este análisis ya que no se encuentra disponible.	Demuestra que el uso del algoritmo *CatBoost* mejora el desempeño frente a otras opciones de modelo como *Random Forest* o *Gradient Boost*.	Hace uso del *dataset* denominado *Indian loan default dataset*.

Tabla 1: Resumen de brechas, aportes y observaciones de los trabajos relacionados. Fuente propia.

3. IMPLEMENTACIÓN

Para el desarrollo de este trabajo, Wizit cuenta con un conjunto de lineamientos para la implementación de proyectos de *Machine Learning* usando la metodología *MLOps*. El propósito de este trabajo es llevar a cabo la primera aproximación práctica bajo dicha metodología.

De acuerdo con [33], *MLOps* busca proveer un proceso de desarrollo de extremo a extremo para diseñar, construir y gestionar software de *Machine Learning*, favoreciendo que este sea reproducible, que se pueda probar para garantizar su calidad y que se piense para ser adaptable al cambio.

A nivel del proveedor, AWS ofrece una serie de lineamientos y utilidades para la construcción de *pipelines* de *MLOps*, incluyendo plantillas. Para el caso de este proyecto se utiliza la plantilla "*MLOps template for model building, training, and deployment"* [75] que permite provisionar los recursos web necesarios para el proceso y automatizar todo el ciclo de vida del modelo, incluyendo las etapas de construcción, entrenamiento y despliegue. A continuación se detalla el *pipeline* en la siguiente figura:

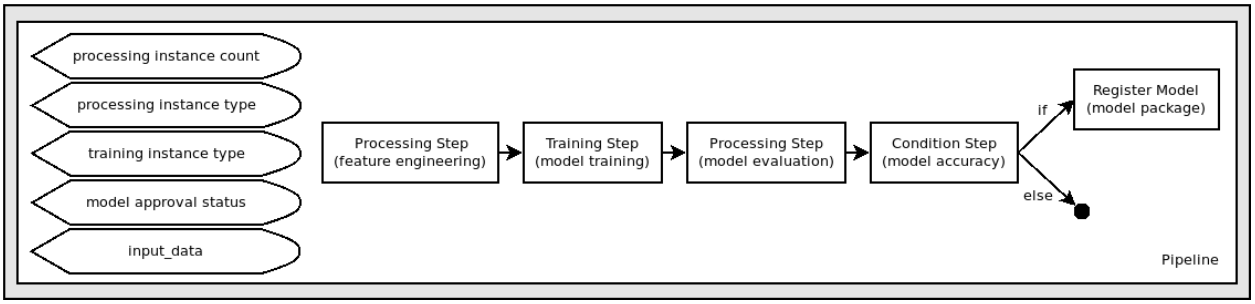


Figura 4. Diagrama de flujo de la plantilla del Pipeline seleccionado. Tomado de [76].

Nota: *El paso a paso para tener acceso a esta plantilla en AWS se desarrolla en el Anexo A de este trabajo.*

De acuerdo con la figura 4, la plantilla provista por AWS cuenta con cinco pasos que soportan la línea de ensamblaje del proceso de *MLOps*, dichas etapas se comportan de la siguiente manera:

1. **Procesamiento:** El paso de procesamiento o *processing step* se encarga de la adecuación de la fuente de datos que será utilizada para nutrir el modelo. Aquí tiene lugar el proceso conocido como ingeniería de características donde se crean, transforman y se seleccionan aquellas columnas del *dataset* que revisten mayor peso o valor al momento de realizar una predicción. Este paso es de vital importancia ya que de este depende en gran medida la calidad final de la implementación.

2. **Entrenamiento:** El paso de entrenamiento o *training step* recibe como insumo principal los datos procesados en el paso inmediatamente anterior. Estos datos pasan a nutrir el modelo escogido para la implementación con el fin de que este pueda identificar patrones dentro de los mismos que le permitan realizar predicciones sobre futuras entradas.

3. **Evaluación:** En el paso de evaluación o *evaluation step* se recolectan todas las métricas relevantes para determinar más adelante si el modelo entrenado cumple con los objetivos para los cuales fue creado según la lógica de negocio establecida.

4. **Evaluación condicional:** Una vez han sido recolectadas las métricas, se procede a determinar si éstas cumplen con los umbrales preestablecidos para conocer si la calidad de la implementación es aceptable según las normas establecidas por el negocio.

5. **Registro:** Si el modelo entrenado cumple con los requisitos a nivel de negocio, éste es registrado para posteriormente ser desplegado a los ambientes de pruebas y producción de manera respectiva.

Después de la creación se va a aprovisionar automáticamente toda la infraestructura necesaria en AWS para construir, entrenar y desplegar el modelo en los ambientes de desarrollo y producción.

A continuación se detalla un diagrama que describe las etapas de construcción y entrenamiento:

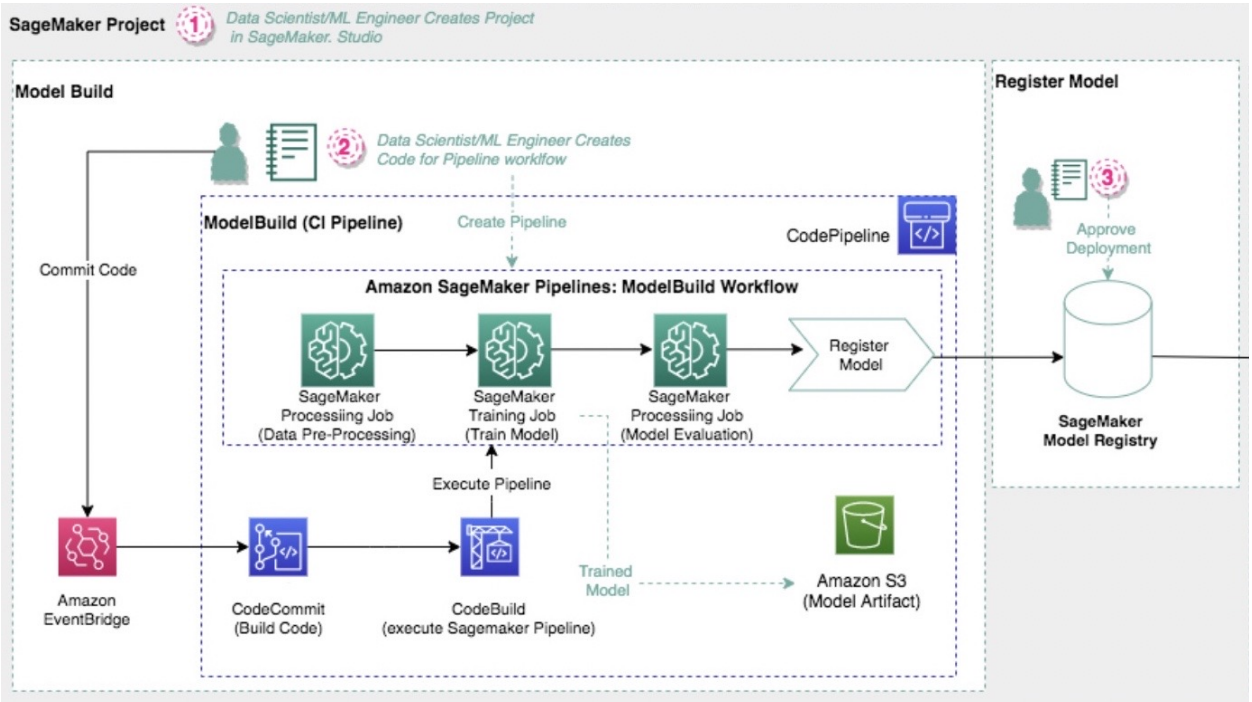


Figura 5. Diagrama de flujo para la construcción del pipeline de la plantilla. Tomado de [75].

En la figura 5 se pueden apreciar tres etapas para la construcción del modelo. El proceso realizado se describe a continuación:

1. Un científico de datos o ingeniero de *Machine Learning* crea el proyecto a través de *SageMaker Studio*.

2. Un científico de datos o ingeniero de *Machine Learning* crea el código que va a soportar los requerimientos esperados y envía un *commit* de éste al repositorio de *CodeCommit*, el cual despacha un evento de *Amazon EventBridge* que inicializa la ejecución del *pipeline* a través de *CodePipeline*, orquestando los recursos necesarios como:

    a. ***CodeCommit*:** Para la gestión del repositorio remoto usando Git. Permite asegurar y encriptar el código fuente proyecto, apoyándose en *AWS KMS* para aumentar la privacidad y seguridad. También compila el código fuente.

       En este escenario *CodeCommit* será el servicio utilizado para gestionar el repositorio del proyecto. Se ha incluído además una integración con *EventBridge*, el cual detecta los cambios en el repositorio y se encarga de compilar y ejecutar el código fuente del *pipeline* de *SageMaker* incluyendo parámetros como:

        i. ***AWS_REGION*:** La región de AWS hace referencia a una ubicación física alrededor del mundo donde se ubican sus *data centers*. Las regiones cuentan con grupos de *data centers* que proveen seguridad, baja latencia y redundancia con el fin de garantizar la mayor disponibilidad posible del servicio. Para este proyecto, la región escogida fue "*us-east-1"* ubicada en el Norte de Virginia, al este de Estados Unidos [77].

        ii. ***SAGEMAKER_PROJECT_ARN:*** Los nombres de recursos de Amazon o *ARN*, son identificadores únicos asignados a recursos de AWS que se utilizan para referenciar un recurso de manera inequívoca [78]. El valor de este parámetro se asigna de manera automática.

        iii. ***SAGEMAKER_PIPELINE_ROLE_ARN:*** Se refiere al *ARN* del rol de *IAM* que permite gestionar y manejar todos los recursos asociados con la construcción, entrenamiento y despliegue del *pipeline*. El valor de este parámetro se asigna de manera

automática.

iv. ***ARTIFACT_BUCKET:*** Es un *bucket* en *S3* que almacena las salidas del proceso de entrenamiento del modelo.

v. ***SAGEMAKER_PROJECT_NAME_ID:*** Identificador único del nombre del proyecto.

b. ***CodeBuild*****:** Para ejecutar pruebas automáticas y producir paquetes de código listos para desplegar en la infraestructura.

c. SageMaker**:** Permite realizar las tareas de preprocesamiento, entrenamiento y evaluación del modelo. Cuenta con dos salidas, una que almacena el modelo entrenado en un *bucket* de *Amazon S3*, y la otra que despliega el modelo a *staging* y *production* si el usuario lo aprueba. Dentro de este servicio es importante resaltar dos apartes importantes del mismo:

i. ***Amazon SageMaker Pipelines ModelBuild Workflow*****:** Permite construir un flujo para seleccionar, manipular y transformar el *dataset*. De igual manera permite efectuar validaciones y pruebas de los diferentes algoritmos para *Machine Learning*, y registrar el modelo de nuestra preferencia en *SageMaker Model Registry*.

ii. ***SageMaker Model Registry*****:** Genera un artefacto del modelo entrenado el cual quedará disponible para publicación si las métricas del modelo satisfacen el umbral deseado y si el usuario lo aprueba. Esta herramienta también permite [79]:

1. Catalogar modelos para producción.
2. Administrar versiones de los modelos.
3. Administrar la aprobación de un modelo para ser candidato a producción.
4. Automatizar el despliegue del modelo utilizando CI/CD.

Nota: *Se debe tener en cuenta que AWS CodePipeline es un componente completamente independiente y diferente al pipeline creado por SageMaker.*

3. Un científico de datos o ingeniero de *Machine Learning* debe aprobar manualmente el despliegue del modelo entrenado según la evaluación de sus métricas.

Una vez aprobado el despliegue, se ejecuta todo el proceso requerido para exponer el modelo. A continuación se muestra un diagrama que describe dicha etapa:

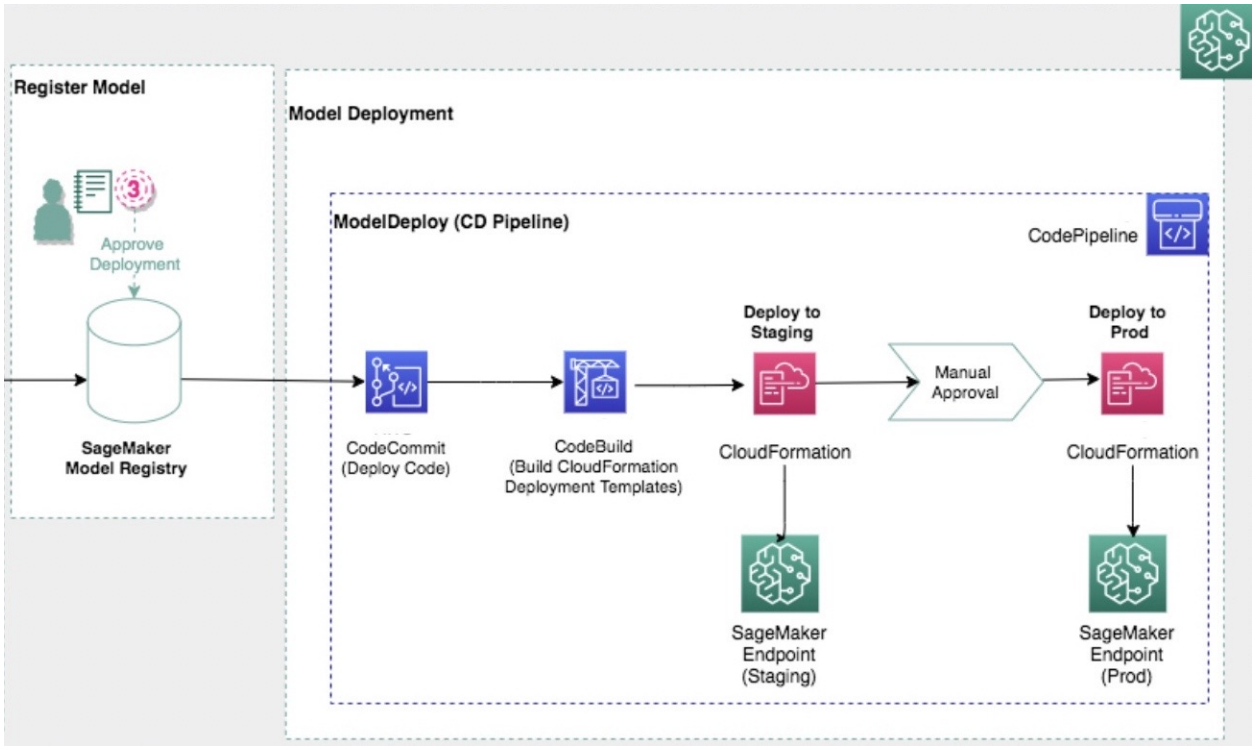


Figura 6. Diagrama de flujo del despliegue de la plantilla. Tomado de [75].

Una vez el modelo entrenado es registrado en *SageMaker Model Registry* mediante aprobación manual, se inicia un proceso cuyo resultado final es el despliegue de dos *endpoints* para los ambientes de *staging* y *production*, el diagrama de arquitectura contempla los siguientes componentes:

- ***CodeCommit*:** Repositorio de código para el despliegue del artefacto (paquete o recopilación de datos del modelo) almacenado en *SageMaker Model Registry* y la exposición de los *endpoints* tipo REST en los diferentes ambientes.

- ***CodeBuild*:** Se encarga de construir los templates CloudFormation para el aprovisionamiento de los recursos de la infraestructura que va a soportar la ejecución del modelo.

- ***CloudFormation*:** Se encarga de configurar y aprovisionar toda la infraestructura de nuestro modelo de tal forma que pueda ser invocado por medio de los *endpoints* expuestos en *staging* y en *production* (después de ser aprobado de forma manual), *CloudFormation* gestiona recursos como:

    - Tipo de instancias para la ejecución del modelo.
    - Cantidad de instancias para la ejecución del modelo.
    - Despliegue del paquete en las instancias.
    - Configuración del *endpoint SageMaker*.

- ***CodeDeploy*:** Automatización de despliegue de código en elementos de infraestructura como máquinas virtuales *EC2*, funciones *Lambda* o servidores *on-premise.*

Una vez se ha ejecutado todo el proceso de manera satisfactoria, el modelo queda expuesto para pruebas y producción a través de dos puntos de consulta que son llamados desde las aplicaciones web o móviles que se apoyan sobre estas funcionalidades.

3.1. *Foundations*

Aunque ya han sido descritos los servicios web que van a representar mayor protagonismo en el desarrollo de este proyecto, se procede a mencionar herramientas adicionales que facilitaron la realización de ciertas tareas:

- **Git:** Es un sistema de control de versiones abierto y distribuido que ha sido diseñado para manejar proyectos grandes y pequeños con velocidad y eficiencia [80].

- ***Google Colab*:** Es un producto de Google que permite escribir y ejecutar código arbitrario hecho en Python a través del navegador y que está especialmente adecuado para tareas de *Machine Learning*, análisis de datos y educación. Esta herramienta alberga un servicio de gestión de *notebooks* de *Jupyter* sin requerir de configuraciones para ser utilizado [81].

- ***Visual Studio Code:*** Es una herramienta para edición de código fuente que puede correr en Windows, macOS y Linux. Incluye soporte para lenguajes como *JavaScript* y *Python* [82].

Adicional a las herramientas expuestas anteriormente y como parte de las bases de este proyecto, se procede a describir la metodología *Cross-Industry Standard Process for the development of Machine Learning applications with Quality assurance* o *CRISP-ML(Q)*, la cual es una de las bases para *MLOps*. En la siguiente figura se contemplan las fases principales de dicha metodología:

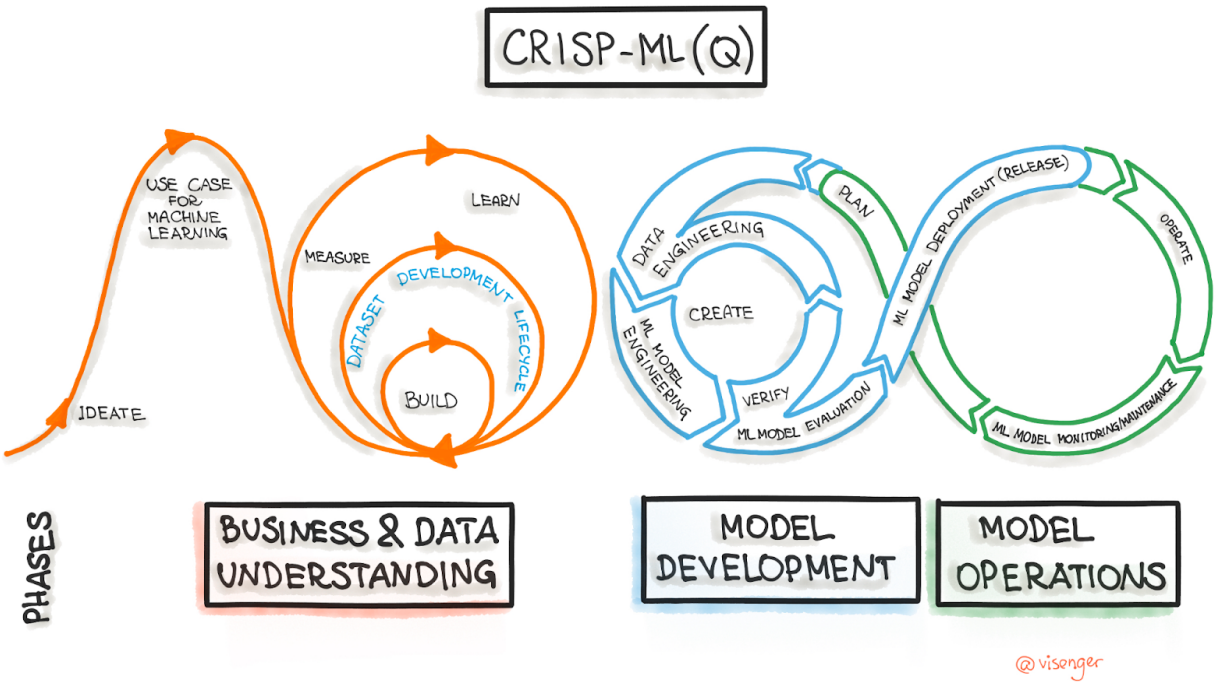


Figura 7. Ciclo de vida de CRISP-ML(Q). Tomado de [33].

A continuación, en la tabla 2, se relacionan algunas de las etapas de dicha metodología, con algunas de las tareas que se incluyeron en la realizacion de este proyecto:

Fase *CRISP-ML(Q)*	Tareas
Business and Data Understanding	• Definición de los objetivos del negocio. • Traducción de los objetivos de negocio a objetivos de *Machine Learning*. • Recolección y verificación de los datos. • Determinación de la viabilidad del proyecto.
Data Engineering	• Selección de características. • Balanceo del *dataset* • Limpieza de los datos. • Ingeniería de características. • Estandarización de los datos.
ML Model Engineering	• Definir la métrica de calidad del modelo. • Selección del algoritmo de *Machine Learning*. • Entrenamiento del modelo • Documentación del modelo de *Machine Learning* y pruebas.
ML Model Evaluation	• Validación del desempeño del modelo. • Desplegar el modelo según las métricas. • Documentación de la fase de evaluación.

Model Deployment	• Evaluación del modelo en producción.

Tabla 2: Etapas de CRISP-ML (Q) con las tareas que aplican para este proyecto. Adaptado de [33].

3.2. Preparación de los datos

3.2.1. Definición de la fuente de datos

Una de las consideraciones más importantes a tener en cuenta es que la empresa no cuenta con datos propios ni de terceros para llevar a cabo este proyecto, por ello, es necesario investigar en fuentes como *https://www.kaggle.com* y *https://datasetsearch.research.google.com* para conocer los *datasets* que información relacionada con préstamos bancarios realizados a las personas dentro del contexto Colombiano.

En la tabla 3 se relacionan los *datasets* encontrados usando las mencionadas plataformas:

Dataset	**Observaciones**	**URL**
Loans management from KIVA	• Contiene ***21995*** registros de préstamos en Colombia. • El *dataset* no presenta información que permita determinar si un cliente pagará o no su préstamo.	[83]
Datatón Bancolombia 2019	• El uso del *dataset* está restringido a la competencia Datatón Bancolombia 2019.	[84]

Tabla 3: Datasets encontrados en la búsqueda preliminar. Fuente propia.

Debido a la falta de datos en el contexto colombiano que pudieran utilizarse para este proyecto, se optó por estudiar los *datasets* obtenidos a partir de la revisión bibliográfica.

A continuación, en la tabla 4, se detallan las fuentes resultantes de este ejercicio:

Dataset	**Observaciones**	**URL**
Statlog (German Credit Data) Data Set	• Tiene ***20*** columnas y ***1000*** registros. • Contiene datos demográficos sobre los clientes (Propiedad del hogar, personas a su cargo, edad, sexo entre otros). • Contiene una columna que indica si el cliente cumple o no con el pago de sus créditos. • Los datos presentes pertenecen al contexto Alemán y fueron donados a la Universidad de California — Irvine en 1994.	[85]

Statlog (Australian Credit Approval) Data Set	• Tiene ***15*** columnas y ***690*** registros. • Los datos han sido anonimizados y las columnas han sido renombradas para no revelar su contenido. • Contiene una columna que indica si el crédito fue aprobado para el cliente. • Los datos presentes pertenecen al contexto Australiano y fueron donados a la Universidad de California — Irvine.	[86]
Credit Scoring DataSet (CSDS)	• Se compone de tres archivos; CSDS 1, CSDS 2 y CSDS 3. • Los datos han sido anonimizados y las columnas han sido renombradas para no revelar su contenido. • El *dataset* está restringido para uso académico. • Los datos presentes pertenecen al contexto Brasileño y fueron recolectados por instituciones financieras de ese país.	[87]
Qualitative_Bankruptcy Data Set	• Tiene ***7*** columnas y ***250*** registros. • Sus columnas son de tipo categórico y se denominan *Industrial Risk, Management Risk, Financial Flexibility, Credibility, Competitiveness, Operating Risk* y *Class*. • Los datos obedecen a un contexto en donde se evalúan atributos de las empresas y se determina si están o no en bancarrota [88]. • Los datos fueron recogidos en India y donados a la Universidad de California — Irvine en 2014.	[89]
Bank Marketing Data Set	• Tiene ***17*** columnas y ***45211*** registros. • Los datos fueron recolectados en campañas de marketing de una institucion financiera de Portugal, y se perfila a los clientes que van o no a tomar un producto de dicha entidad. • Los datos fueron donados a la Universidad de California — Irvine en 2012.	[90]
default of credit card clients Data Set	• Tiene ***24*** columnas y ***30000*** registros. • Contiene datos demográficos como la edad, estado civil, género y nivel educativo; el resto de las columnas corresponden a la variable objetivo y a una versión ampliada del historial de crédito, como los pagos realizados en fechas anteriores por los aplicantes. • Los datos presentes se recolectaron en 2016, pertenecen al contexto Taiwanés y fueron donados a la Universidad de California — Irvine.	[91]
Japanese Credit Screening Data Set	• Tiene ***125*** registros. • Contiene datos demográficos como la edad, estado civil, género y nivel educativo; también se contempla el propósito del préstamo. El resto de los datos pertenecen al historial de crédito. • Los datos presentes se recolectaron en 1992, pertenecen al contexto Japonés y fueron donados a la	[92]

	Universidad de California — Irvine.	
All Lending Club loan data	• Se compone de dos archivos, que contienen datos sobre los préstamos aprobados y rechazados. • El archivo de préstamos aprobados tiene ***27*** columnas que incluyen información demográfica como propiedad de la casa, relación de deuda a ingresos, ingresos anuales, entre otros, además de una columna que indica el estado del préstamo del cliente. Contiene información sobre el historial crediticio (*N*úmero de cuentas o créditos abiertos en el tiempo, entre otros). Consta de aproximadamente ***396030*** registros. • El archivo de préstamos rechazados tiene ***10*** columnas que incluyen información como cantidad solicitada, el propósito del préstamo, relación de deuda a ingresos, ingresos anuales, entre otros, además de una columna que indica el estado del préstamo del cliente. Contiene información sobre el historial crediticio. Consta de aproximadamente ***27648742*** registros. • Recolectado por *Lending Club*, una *fintech* que ofrece servicios financieros como préstamos.	[93]

Tabla 4: Datasets encontrados a partir del análisis cienciométrico. Fuente propia.

La escogencia de la fuente de datos se llevó a cabo considerando factores como; disponibilidad, cantidad de registros y presencia de datos no relacionados con el historial de crédito.

Pese a que *CSDS* [87] reúne datos recolectados en Brasil, que podrían ser más cercanos al contexto de aplicación de este proyecto, su uso está restringido al ámbito académico.

Por otro lado, a partir del *dataset* Australiano [86] no es posible conocer las *features* que permitan establecer si un cliente cumplirá o no con el pago de su préstamo, debido al carácter anonimizado de sus columnas, adicional a la relativamente reducida cantidad de registros con la que cuenta.

El *dataset* denominado *Qualitative_Bankruptcy Data Set* [89] incluye datos categóricos que permiten determinar el riesgo de un negocio de caer en bancarrota, a pesar de poseer información interesante, se escapa del contexto que está siendo estudiado, por lo que no se considera una fuente de datos viable para ser utilizada.

Por su parte, el *dataset* conocido como *Bank Marketing Data Set* [90] contiene una variedad de datos que tienen un componente demográfico como profesional respecto a los usuario, sin embargo, este *dataset* pertenece a un contexto en el que se pretende determinar si un usuario va a tomar o no un préstamo, no si lo va a pagar

completamente, por lo que no se considera una fuente de datos viable para este proyecto.

El *dataset* denominado *default of credit card clients Data Set* [91] pertenece a un contexto donde se busca evaluar la probabilidad de impago de tarjetas de crédito, no obstante, aunque tiene datos no ligados a la historia de crédito, su dependencia de ellos no deja de ser extensa ya que solo 5 de sus 24 columnas representan datos demográficos y del préstamo, por lo cual no se considera una fuente de datos viable.

El *dataset* Japonés contiene información demográfica y sobre el nivel educativo del aplicante, este se da en un contexto demasiado distante al de aplicación de este proyecto, además de tener una dimensionalidad reducida con tan solo 125 registros en comparación con el resto de fuentes de datos consultadas, por lo cual queda descartado.

La fuente de datos Alemana [85] presenta una variedad interesante de columnas ya que contiene información demográfica y socioeconómica de las personas. Pese al valor que dichos datos puedan representar, la cantidad de registros y la fecha en que fueron recolectados componen un impedimento para utilizar dicho *dataset*.

Finalmente, el *dataset* de *Lending Club* [93] presenta un buen balance entre los datos no ligados con el historial de crédito y la cantidad de registros disponibles para el análisis.

3.2.2. Análisis exploratorio de datos

Teniendo en cuenta lo expuesto anteriormente, se realiza el análisis exploratorio de datos o EDA, por sus siglas en inglés, para determinar si el *dataset* escogido permitiría cumplir con el objetivo del proyecto.

- **Identificación de variables:** Durante la revisión preliminar, se encontró un análisis sobre este *dataset* que permite tener un punto de partida para verificar la calidad de los datos, asimismo provee la descripción de las columnas entre otra información relevante para el desarrollo de este proyecto [94].

  A continuación, en la tabla 5, se presenta una relación entre las columnas y sus respectivas descripciones a fin de conocer el *dataset* a mayor profundidad.

Columna	¿Es historial de crédito?	Tipo	Descripción	Valores
loan_status	No	*string*	Estado actual del préstamo y la **variable objetivo**.	*['Fully Paid' 'Charged Off']*
term	No	*string*	El número de cuotas en las que se difiere el crédito. Se	*[' 36 months' ' 60 months']*

			encuentra entre 36 y 60.	
grade	Sí [95]	*string*	Calificación asignada por *Lending Club*.	*['B' 'A' 'C' 'E' 'D' 'F' 'G']*
sub_grade	Sí [95]	*string*	Sub calificación asignada por *Lending Club*.	*['B4' 'B5' 'B3' 'A2' 'C5' 'C3' 'A1' 'B2' 'C1' 'A5' 'E4' 'A4' 'A3' 'D1' 'C2' 'B1' 'D3' 'D5' 'D2' 'E1' 'E2' 'E5' 'F4' 'E3' 'D4' 'G1' 'F5' 'G2' 'C4' 'F1' 'F3' 'G5' 'G4' 'F2' 'G3']*
emp_title	No	*string*	Ocupación del solicitante.	—
emp_length	No	*string*	Tiempo en años en el que el solicitante desempeña dicha ocupación.	*['10+ years' '4 years' '< 1 year' '6 years' '9 years' '2 years' '3 years' '8 years' '7 years' '5 years' '1 year']*
home_ownership	No	*string*	El tipo de propiedad que tiene el solicitante sobre su casa.	*['RENT' 'MORTGAGE' 'OWN' 'OTHER' 'NONE' 'ANY']*
verification_status	No	*string*	Indica si los ingresos (o la fuente de los mismos) fueron verificados o no por *Lending Club*.	*['Not Verified' 'Source Verified' 'Verified']*
issue_d	No	*string*	El mes en que se financió el préstamo.	—
purpose	No	*string*	El propósito por el cual fue solicitado el préstamo.	*['vacation' 'debt_consolidation' 'credit_card' 'home_improvement' 'small_business' 'major_purchase' 'other' 'medical' 'wedding' 'car' 'moving' 'house' 'educational' 'renewable_energy']*
title	No	*string*	Título del préstamo.	—
address	No	*string*	La dirección en donde vive el aplicante.	—
earliest_cr_line	Sí	*string*	El mes en el que la primera línea de crédito fue abierta por el aplicante.	—
initial_list_status	No	*string*	El estado inicial de listado del préstamo.	*['w' 'f']*
application_type	No	*string*	Indica si el crédito se realiza para uno o más aplicantes	*['INDIVIDUAL' 'JOINT' 'DIRECT_PAY']*
loan_amnt	No	*float64*	La cantidad del crédito que solicita el aplicante.	*[500.0, 40000.0]*
int_rate	No	*float64*	Tasa de interés del préstamo.	*[5.32, 30.99]*
installment	No	*float64*	Valor del pago mensual realizado por el solicitante.	*[16.08, 1533.81]*
annual_inc	No	*float64*	Los ingresos anuales del solicitante.	*[0.0, 8706582.0]*

dti	No	*float64*	Una relación calculada utilizando los pagos de obligaciones totales mensuales sobre los ingresos totales del solicitante, excluyendo hipotecas y servicios públicos.	*[0.0, 9999.0]*
open_acc	Sí	*float64*	El número de cuentas de crédito abiertas del aplicante.	*[0.0, 90.0]*
pub_rec	Sí	*float64*	La cantidad de registros públicos que puedan afectar la reputación del aplicante.	*[0.0, 86.0]*
revol_bal	Sí	*float64*	Balance total rotativo del aplicante, por ejemplo, el cupo total de sus tarjetas de crédito.	*[0.0, 1743266.0]*
revol_util	Sí	*float64*	El cupo de crédito rotativo utilizado por el aplicante.	*[0.0, 892.3]*
total_acc	Sí	*float64*	Número total de cuentas de crédito del aplicante.	*[2.0, 151.0]*
mort_acc	No	*float64*	Número de hipotecas.	*[0.0, 34.0]*
pub_rec_bankruptcies	Sí	*float64*	Número de bancarrotas públicas de las que se tiene registro.	*[0.0, 8.0]*

Tabla 5: Descripción de las columnas del dataset de Lending Club. Fuente propia.

A partir de la información provista en la tabla 4, se procede a analizar la tendencia de los datos, para ello se utiliza a librería *seaborn* sobre este *dataset*, obteniendo un mapa de calor que permite determinar la correlación que existe entre las columnas numéricas:

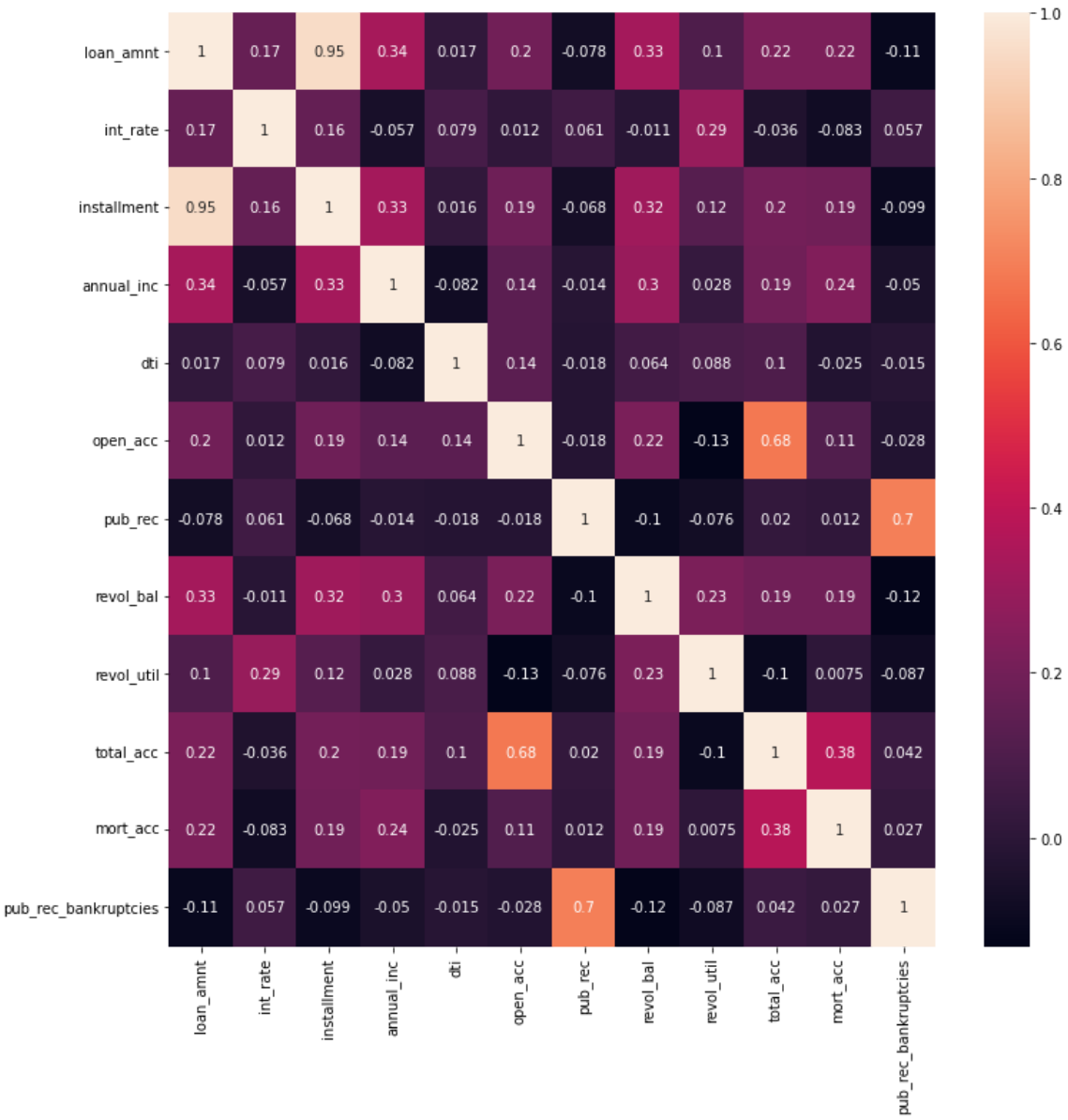


Figura 8. Mapa de calor de correlación entre variables numéricas. *Fuente propia.*

Para apoyar los resultados observados en la figura 8, se propone el siguiente gráfico de dispersión:

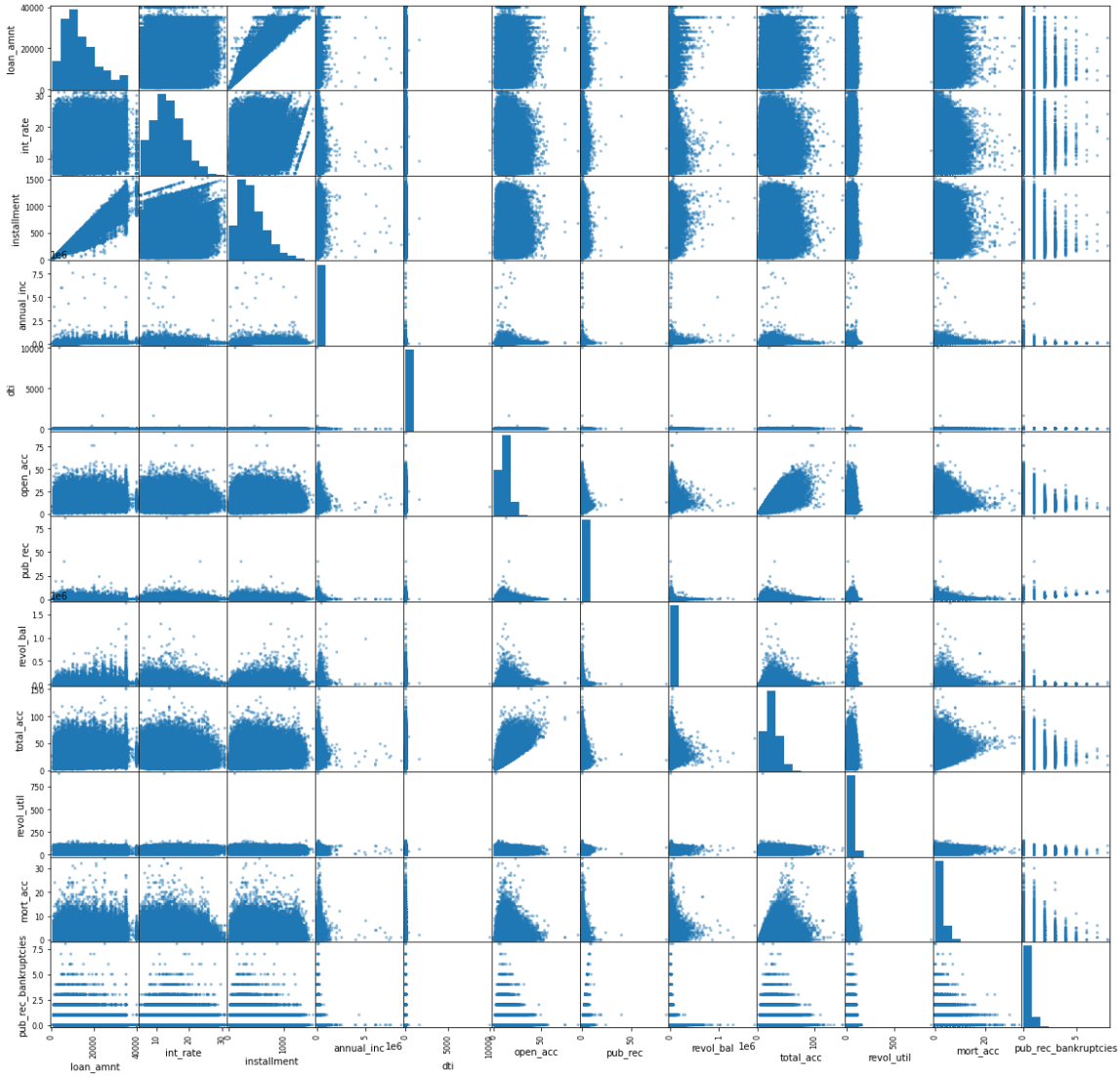


Figura 9. Gráfica de dispersión de los datos numéricos. Fuente propia.

Inicialmente, se puede establecer que, al menos a nivel numérico, el *dataset* tiene coherencia. Esta se ve reflejada en la correlación que existe entre *annual_inc* e *installment*. Adicionalmente, se observa un comportamiento similar entre *total_acc*, *open_acc* y *mort_acc*, también entre *pub_rec* y *pub_rec_bankruptcies*. Por otro lado, la correlación entre las variables restantes se observa baja, cercana a cero.

- **Búsqueda de datos nulos:** Para determinar si existen valores como *null* o *NaN* (*Not a Number*, por sus siglas en inglés) y espacios en blanco dentro de las columnas del *dataset*, se hizo uso del módulo *pandas* de Python, que ofreció de forma tabulada la cantidad de de registros nulos por cada columna, detallados en la siguiente tabla:

Columna	Cantidad *(Null, NaN)*	Porcentaje
mort_acc	*37795*	*9.54%*
emp_title	*22927*	*5.79%*
emp_length	*18301*	*4.62%*
title	*1755*	*0.44%*
pub_rec_bankruptcies	*535*	*0.14%*
revol_util	*276*	*0.07%*
loan_amnt	*0*	*0.00%*
term	*0*	*0.00%*
int_rate	*0*	*0.00%*
installment	*0*	*0.00%*
grade	*0*	*0.00%*
sub_grade	*0*	*0.00%*
home_ownership	*0*	*0.00%*
annual_inc	*0*	*0.00%*
verification_status	*0*	*0.00%*
issue_d	*0*	*0.00%*
loan_status	*0*	*0.00%*
purpose	*0*	*0.00%*
dti	*0*	*0.00%*
earliest_cr_line	*0*	*0.00%*
open_acc	*0*	*0.00%*
pub_rec	*0*	*0.00%*
revol_bal	*0*	*0.00%*
total_acc	*0*	*0.00%*
initial_list_status	*0*	*0.00%*
application_type	*0*	*0.00%*
address	*0*	*0.00%*

Tabla 6: Cantidad y porcentaje de datos nulos por columna. Fuente propia.

Para este caso, la mayor cantidad de datos nulos está en la columna *mort_acc*. Más adelante se validará el impacto de esta y otras variables dentro de la predicción de *loan_status*.

- **Valores atípicos y distribución de los datos:** Para conocer los valores atípicos, este trabajo se apoyó en histogramas y diagramas de caja para

determinar la distribución de los datos numéricos.

Se observa que todas las columnas numéricas presentan valores atípicos, para su correcta visualización, algunos de los datos tuvieron que ser reescalados para encajar dentro de la gráfica. Estas columnas y sus respectivos topes son:

- *annual_inc:* ***300000****.*
- *revol_bal:* ***150000****.*
- *revol_util:* ***200****.*
- *dti:* ***200****.*
- *pub_rec:* ***20****.*

A continuación se relacionan los histogramas y diagramas de caja correspondientes a las variables numéricas del *dataset*:

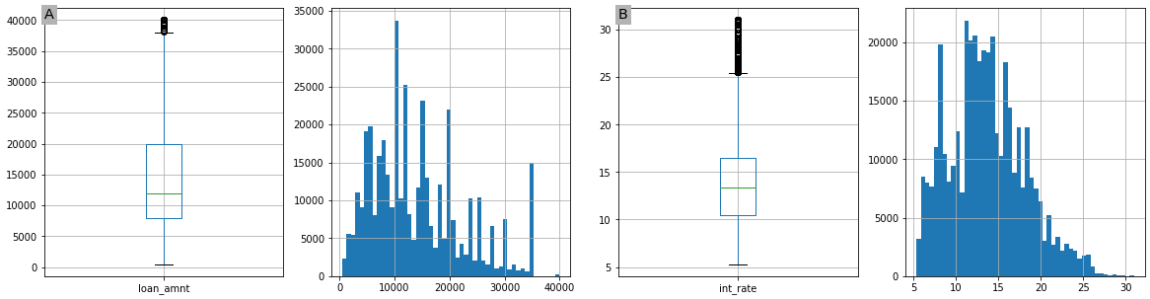


Figura 9.A: Diagrama de caja e histograma de loan_amnt. Fuente propia.
Figura 9.B: Diagrama de caja e histograma de int_rate. Fuente propia.

A partir de la figura 9.A, se puede observar que los datos de *loan_amnt* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *37500*. Los cuartiles Q1 y Q3 se sitúan entre *20000* y *12500,* y *12500* y *7500* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *12500*.

A partir de la figura 9.B, se puede observar que los datos de *int_rate* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *5* y *25*. Los cuartiles Q1 y Q3 se sitúan entre *17* y *12.5,* y *12.5* y *10* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *12.5*.

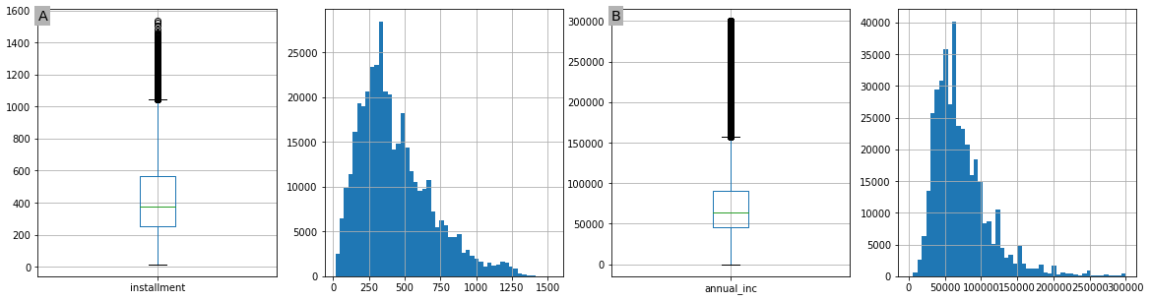


Figura 10.A: Diagrama de caja e histograma de installment. Fuente propia.
Figura 10.B: Diagrama de caja e histograma de annual_inc. Fuente propia.

A partir de la figura 10.A, se puede observar que los datos de *installment* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *1050*. Los cuartiles Q1 y Q3 se sitúan entre *575* y *400,* y *400* y *250* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *400*.

A partir de la figura 10.B, se puede observar que los datos de *annual_inc* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *150500*. Los cuartiles Q1 y Q3 se sitúan entre *90000* y *60000,* y *60000* y *45000* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *60000*.

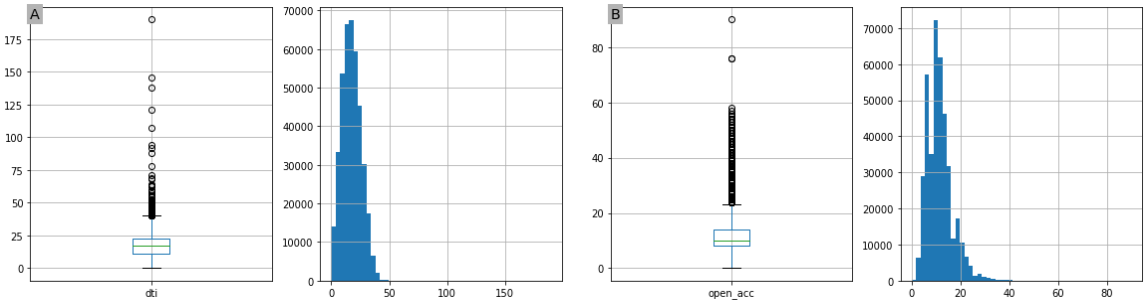


Figura 11.A: Diagrama de caja e histograma de dti. Fuente propia.
Figura 11.B: Diagrama de caja e histograma de open_acc. Fuente propia.

A partir de la figura 11.A, se puede observar que los datos de *dti* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *37.5*. Los cuartiles Q1 y Q3 se sitúan entre *25* y *18.75,* y *18.75* y 12*.5* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de 12*.5*.

A partir de la figura 11.B, se puede observar que los datos de *open_acc* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *25*. Los cuartiles Q1 y Q3 se sitúan entre *15* y *10,* y *10* y *8* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *10*.

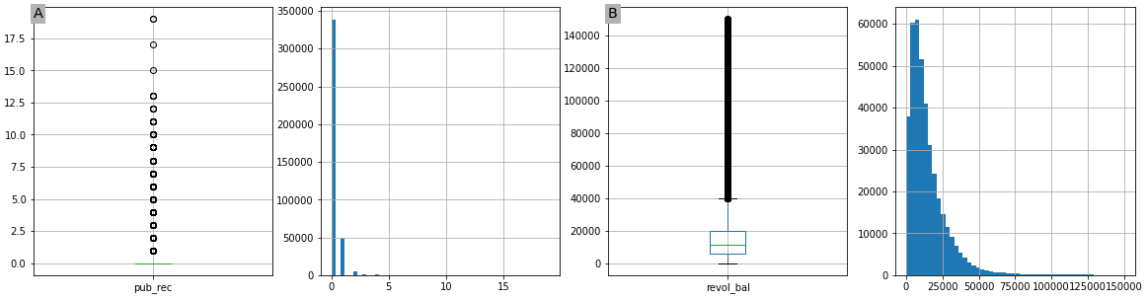


Figura 12.A: Diagrama de caja e histograma de pub_rec. Fuente propia.
Figura 12.B: Diagrama de caja e histograma de revol_bal. Fuente propia.

A partir de la figura 12.A, se puede observar que los datos de *pub_rec* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *20*. Dado que el ***85.41%*** de los valores en esta columna son *0*, no se pueden determinar los cuartiles.

A partir de la figura 12.B, se puede observar que los datos de *revol_bal* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *150000*. Los cuartiles Q1 y Q3 se sitúan entre *20000* y *40000,* y *0* y *5000* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *10000*.

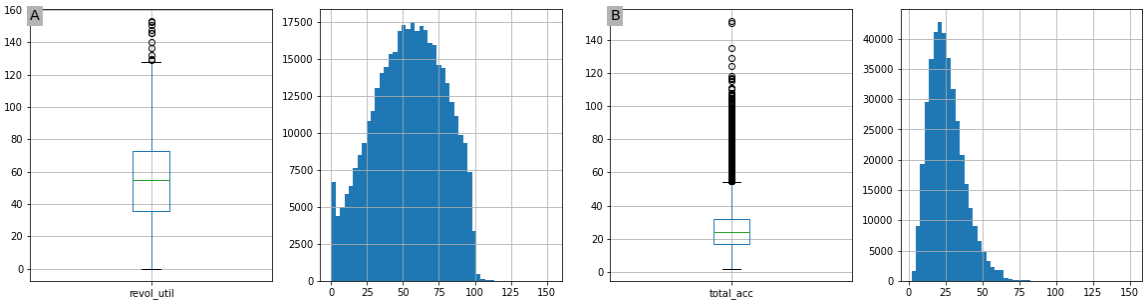


Figura 13.A: Diagrama de caja e histograma de revol_util. Fuente propia.
Figura 13.B: Diagrama de caja e histograma de total_acc. Fuente propia.

A partir de la figura 13.A, se puede observar que los datos de *revol_util* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *890*. Los cuartiles Q1 y Q3 se sitúan entre *72.9* y *130,* y *35.8* y *0* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *53.79*.

A partir de la figura 13.B, se puede observar que los datos de *total_acc* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *2* y *151*. Los cuartiles Q1 y Q3 se sitúan entre *32* y *55,* y *17* y *2* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *25.41*.

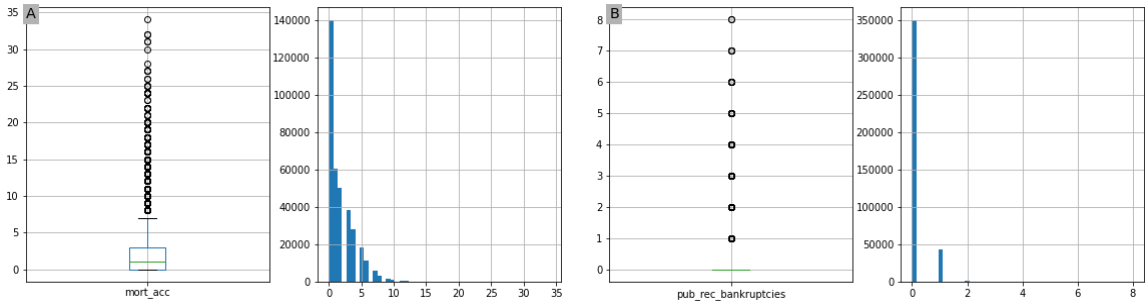


Figura 14.A: Diagrama de caja e histograma de mort_acc. Fuente propia.
Figura 14.B: Diagrama de caja e histograma de pub_rec_bankruptcies. Fuente propia.

A partir de la figura 14.A, se puede observar que los datos de *mort_acc* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *35*. Los cuartiles Q1 y Q3 se sitúan entre *3.5* y *7.5,* y *0* respectivamente, mientras que la mediana o Q2, se sitúa alrededor de *1.5*.

A partir de la figura 14.B, se puede observar que los datos de *pub_rec_bankruptcies* no siguen una distribución normal y poseen datos atípicos. Los valores oscilan en un rango aproximado entre *0* y *8*. Dado que el ***88.47%*** de los valores en esta columna son *0*, no se pueden determinar los cuartiles.

Adicionalmente, es posible notar que ninguna de las columnas numéricas sigue una distribución de datos normal. Esto implica que se deben realizar pruebas para datos no paramétricos en el análisis bivariado.

Para las variables categóricas, se obtuvieron histogramas que muestran la distribución de valores que tienen las distintas categorías:

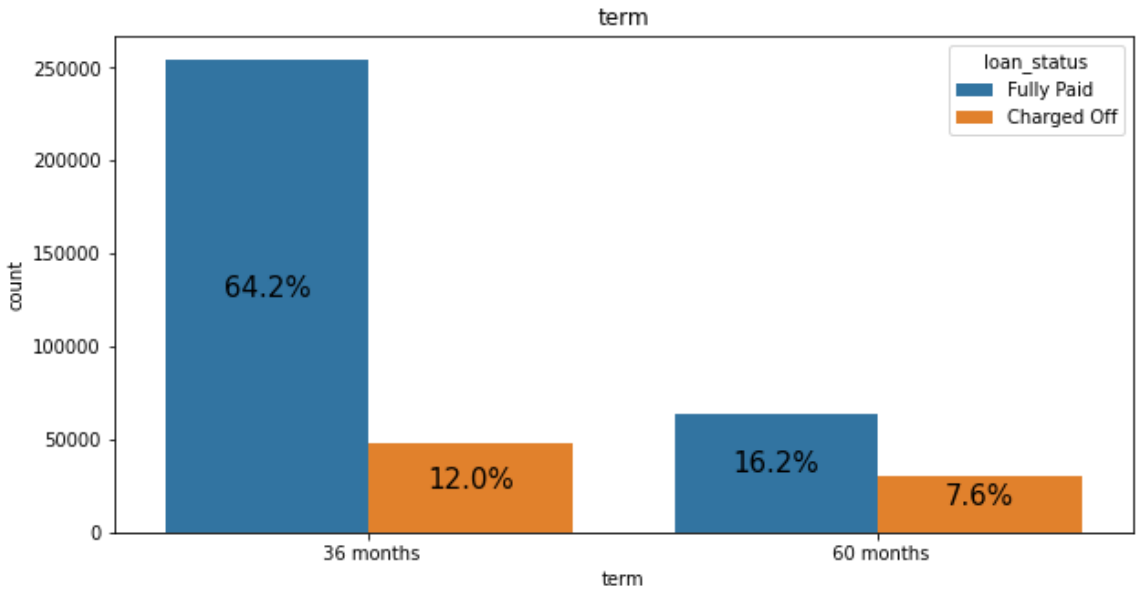


Figura 15: Histograma de term. Fuente propia.

En la figura 15, se observa que los préstamos diferidos a 36 meses tienen mayores probabilidades de pagarse completamente, mientras que los préstamos a 60 meses son menores en cantidad pero tienen mayor probabilidad de no pagarse con relación a los préstamos de 36 meses.

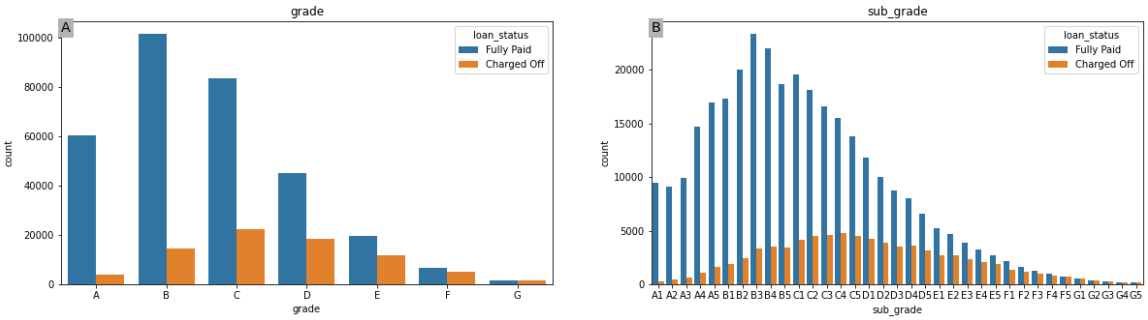


Figura 16.A: Histograma de grade. Fuente propia.
Figura 16.B: Histograma de sub_grade. Fuente propia.

En las figuras 16.A y 16.B, la mayor parte de los datos se concentra entre las categorías *B* y *C*.

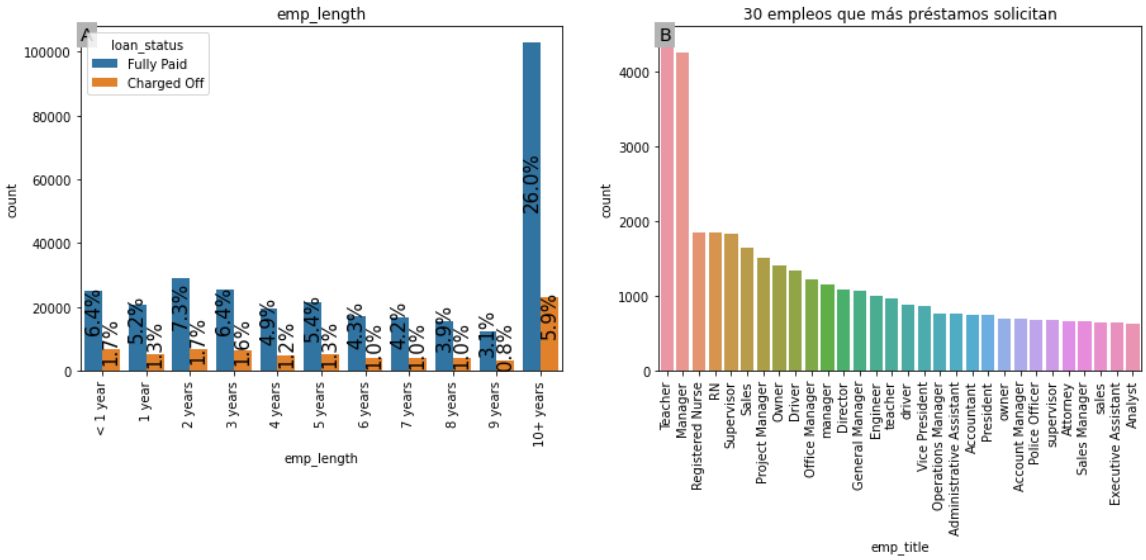


Figura 17.A: Histograma de emp_length. Fuente propia.
Figura 17.B: Los 30 empleos que más préstamos solicitan. Fuente propia.

En la figura 17.A, existe mayor concentración de datos en la categoría *10+ years*, sin embargo, esto puede deberse a que esta categoría representa un rango bastante más extenso que los anteriores.

Por otra parte, en la figura 17.B, se aprecia que los dos empleos con mayor cantidad de préstamos asociados son *Teacher* o profesor y *Manager* o administrador. Es importante anotar que haciendo uso del método *value_counts* sobre dicha columna, la cantidad de empleos registrados es de ***173105*** el cual es un valor engorroso para trabajar debido a su tamaño.

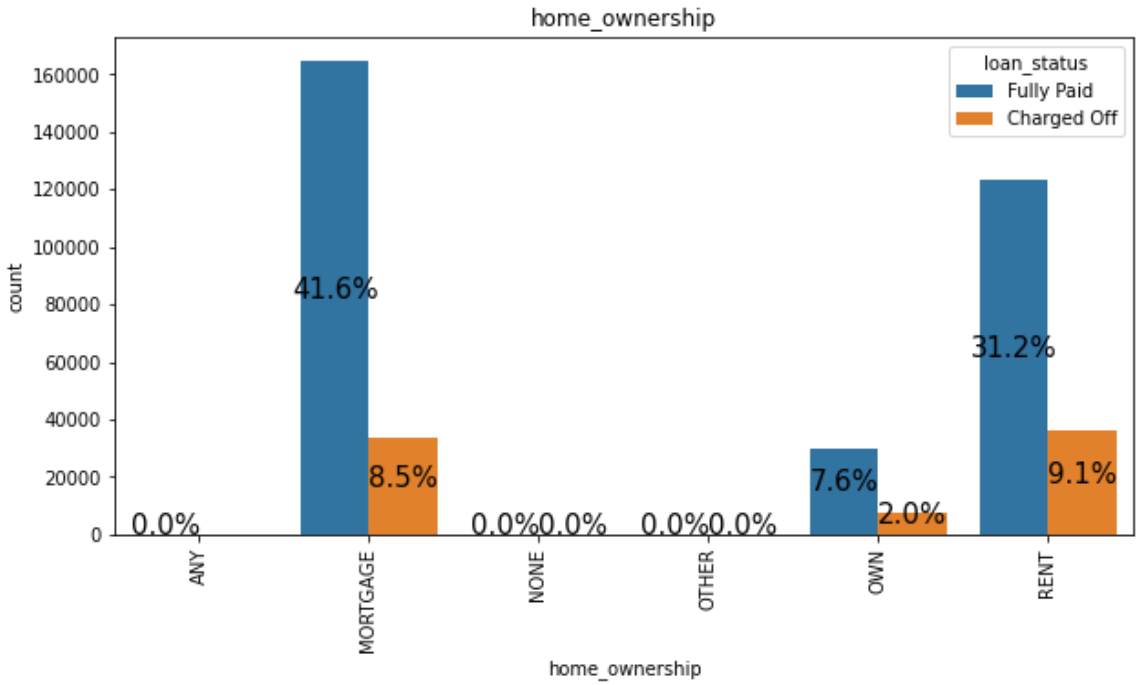


Figura 18: Histograma de home_ownership. Fuente propia.

En la figura 18, la mayor parte de los datos se concentra en las categorías *MORTGAGE* y *RENT*, por lo tanto, los créditos son mayormente solicitados por personas que habitan en un hogar hipotecado o rentado.

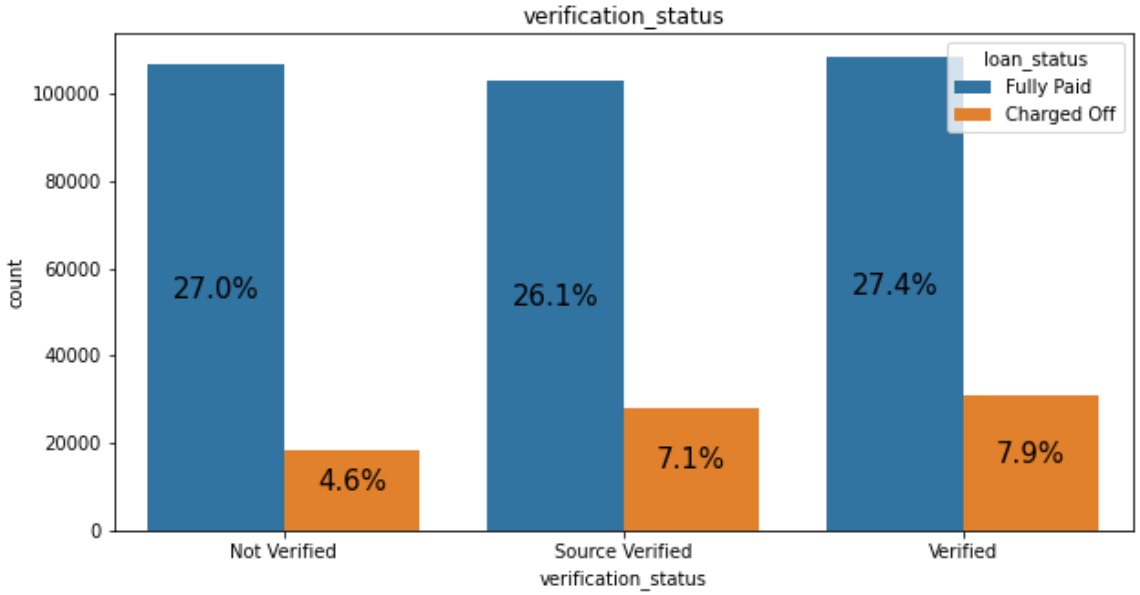


Figura 19: Histograma de verification_status. Fuente propia.

En la figura 19, no existe una diferencia estadística muy amplia entre la cantidad de registros asociados a cada categoría.

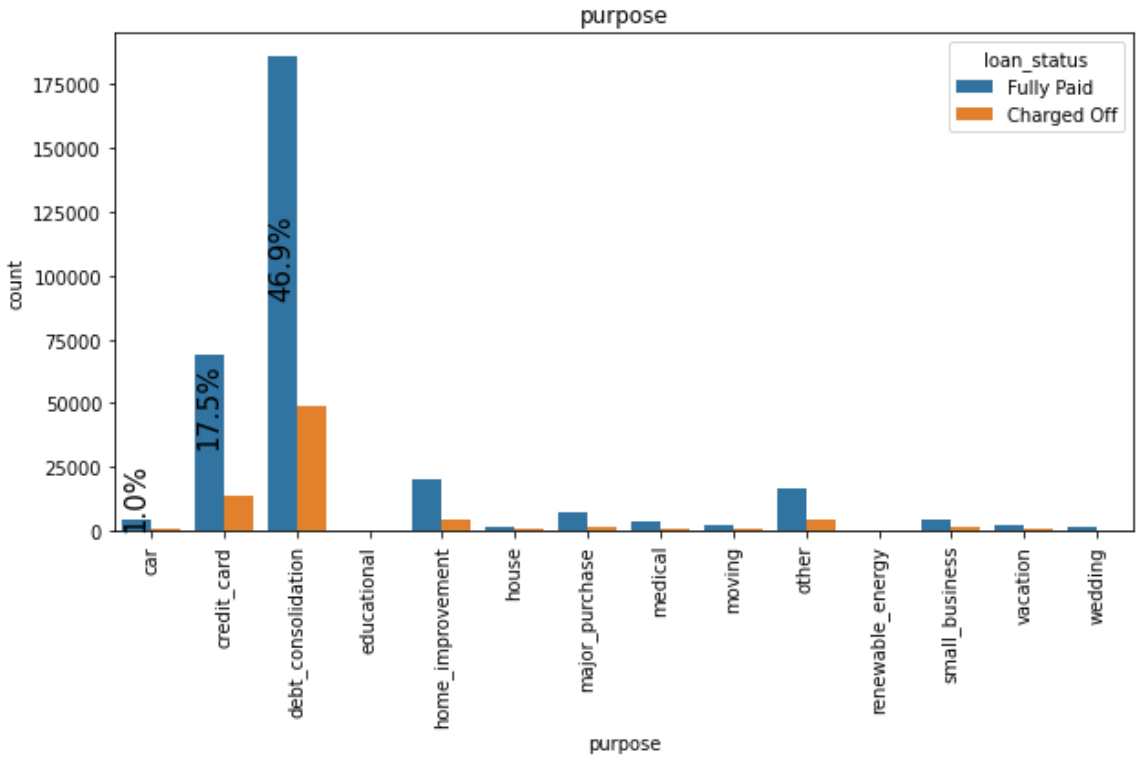


Figura 20: Histograma de purpose. Fuente propia.

En la figura 20, se evidencia que en su mayoría la finalidad de los préstamos es *debt_consolidation* o consolidación de deuda y *credit_card* o pago de tarjeta de crédito

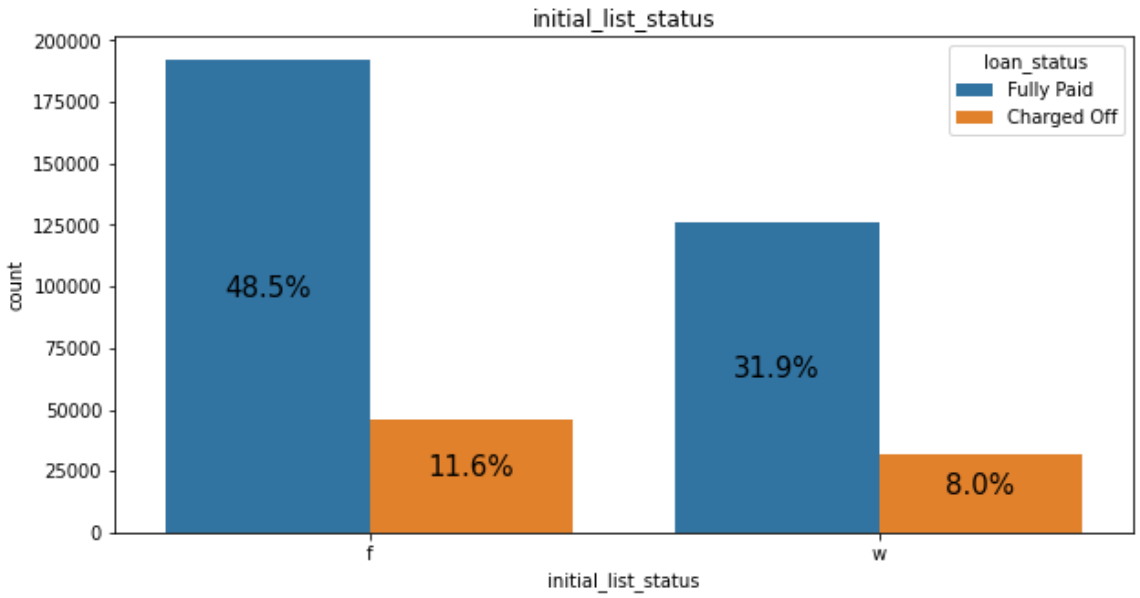


Figura 21: Histograma de initial_list_status. Fuente propia.

En la figura 21, se observa una diferencia notable entre las categorías *w* y *f*. Indicando que la mayoría de los préstamos se realizan en su totalidad por un solo prestador en lugar de varios.

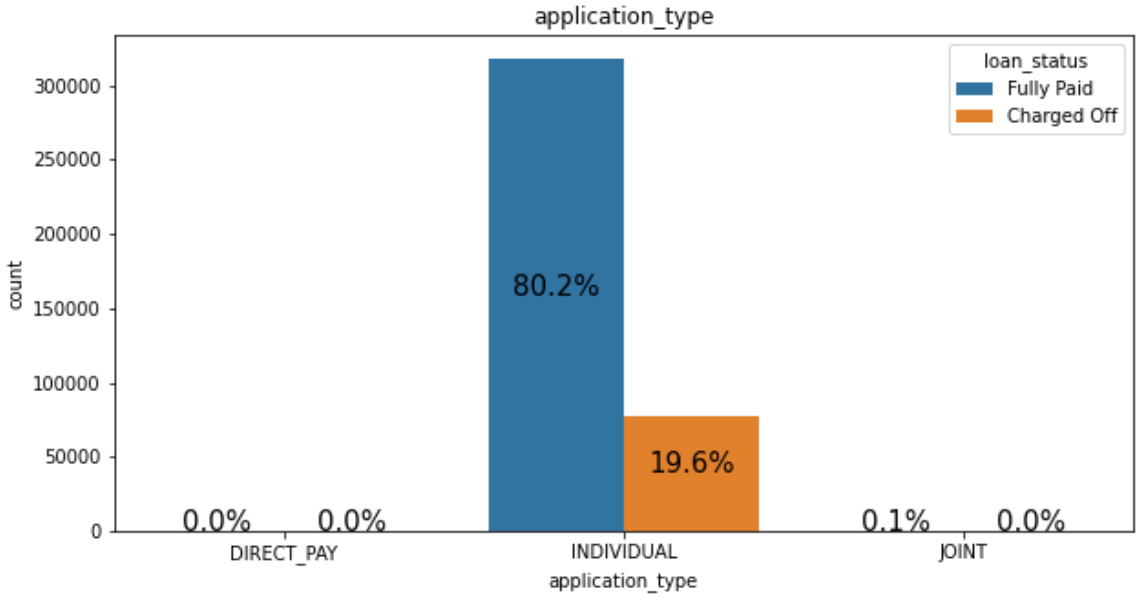


Figura 22: Histograma de application_type. Fuente propia.

En la figura 22, se puede apreciar que casi la totalidad de los préstamos son realizados a título individual, es decir, un solo aplicante o solicitante.

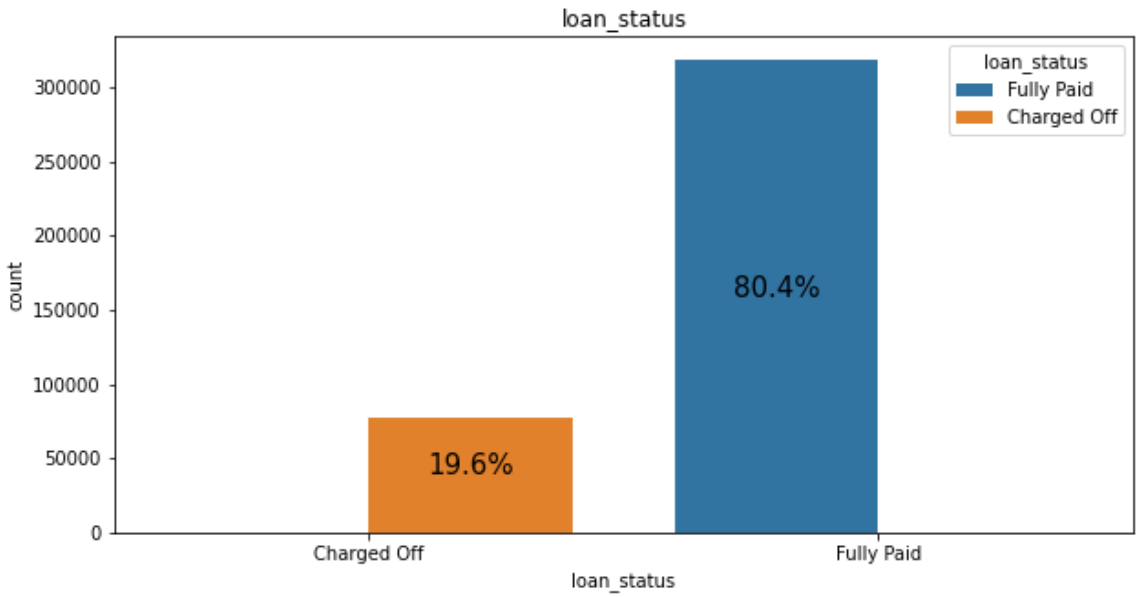


Figura 23: Histograma de loan_status. Fuente propia.

Finalmente, en cuanto a la distribución de *loan_status*, se observa una gran diferencia entre *Fully Paid* y *Charged Off* en cuanto a cantidad de registros, con lo cual es posible afirmar que se trata de un *dataset* desbalanceado respecto a la variable objetivo, lo que puede signi a problemas en la clasificación.

- **Análisis bivariado:** Teniendo en cuenta que la variable objetivo en este caso es *loan_status*, se debe conocer la relación que existe entre ésta y las variables independientes.

    - **Para variables numéricas:** Es importante tener en cuenta que ninguna de estas columnas muestra una distribución normal de sus datos, por lo que es necesario practicar la prueba de *Kruskall-Wallis*, utilizando el módulo *stats* de *scipy*. Esta prueba plantea dos hipótesis a través de las cuales se busca determinar si existe o no una diferencia a nivel estadístico entre las medianas, de la siguiente manera:

        - Hipótesis nula ($p > 0.05$): La mediana es igual en todos los grupos.
        - Hipótesis alternativa ($p \leq 0.05$): La mediana no es igual en todos los grupos.

      El propósito de esta prueba es determinar que la mediana de cada variable independiente es distinta y por lo tanto afirmar que su valor tiene peso sobre la clasificación. En la siguiente tabla se resume dicha información:

Columna	H	p
loan_amnt	*636020.3076*	*0*
int_rate	*635950.9862*	*0*
installment	*635937.658*	*0*
annual_inc	*635954.285*	*0*
dti	*616879.8794*	*0*
open_acc	*635122.6658*	*0*
pub_rec	*536986.0178*	*0*
revol_bal	*622225.3511*	*0*
revol_util	*609468.1844*	*0*
total_acc	*635990.4601*	*0*
mort_acc	*58.29833826*	*2.25E-14*
pub_rec_bankruptcies	*591229.8095*	*0*
loan_amnt	*636020.3076*	*0*

Tabla 7: Prueba de Kruskal-Wallis sobre el dataset de Lending Club. Fuente propia.

Según los resultados de la tabla 7, es posible afirmar que la diferencia entre las medianas de los grupos observados son estadísticamente significativas, por lo tanto, todas las variables numéricas estudiadas son relevantes para la clasificación.

- **Para variables categóricas:** En este caso, se practicó la prueba *Chi-square*, con el fin de establecer si existe dependencia entre *loan_status* y las variables categóricas. Para ello, se utilizó el módulo *chi2_contingency* de *scipy*. Al igual que la prueba de *Kruskall-Wallis*, *Chi-square*, formula las siguientes hipótesis:

  - Hipótesis nula ($p > 0.05$): Ambas variables son independientes.
  - Hipótesis alternativa ($p \leq 0.05$): Las variables tienen algún grado de asociación o relación.

  A continuación, en la tabla 8, se resumen los resultados obtenidos a partir de dicho análisis:

Columna	Coef. Chi^2	p
term	*109223.2417*	*0*
grade	*204724.0782*	*0*
sub_grade	*218276.7733*	*0*
emp_length	*280249.5543*	*1.88E-21*

home_ownership	*608432.4988*	*0*
verification_status	*798.6941747*	*0.00E+00*
purpose	*1834037.325*	*6.57E-291*
initial_list_status	*16201.62716*	*2.41E-09*
application_type	*787799.8173*	*1.14E-13*
term	*109223.2417*	*0*
grade	*204724.0782*	*0*
sub_grade	*218276.7733*	*0*

Tabla 8: Prueba Chi-square sobre el dataset de Lending Club. Fuente propia.

Como se puede observar todos los valores de p de las variables categóricas, se mantienen por debajo de 0.05 con lo que es posible descartar la hipótesis nula y concluir que éstas variables tienen algún tipo de relación entre sí, que es estadísticamente significativa, y por tanto, éstas variables son importantes para la clasificación.

3.2.3. Ingeniería de características

La ingeniería de características es un proceso que se utiliza para seleccionar y transformar variables para provisionar modelos de *Machine Learning* con datos para realizar predicciones. Por lo general incluye la creación, la transformación, la extracción y la selección de características.

- **Creación:** Este *dataset* ya cuenta con un proceso de creación de características donde se obtuvieron columnas como *dti*, *grade* y *sub_grade* [95]. Adicionalmente, se realizó la obtención del código postal o ZIP a partir de la columna *address*. A continuación se presenta la figura 24 que muestra un histograma de codigos de area con relación a la variable objetivo:

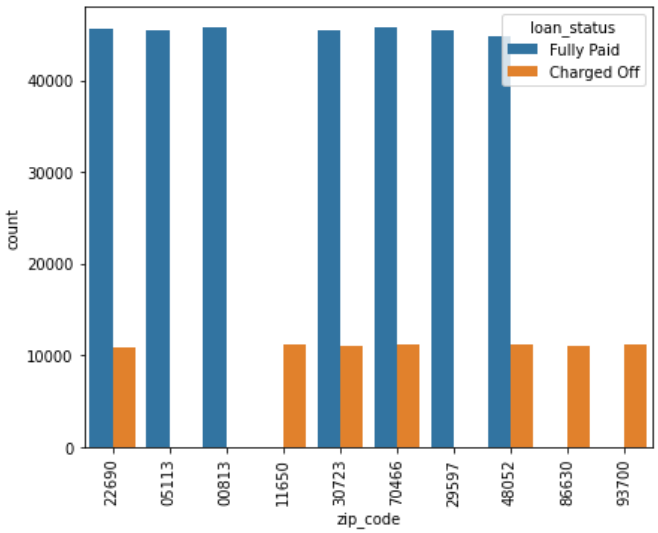


Figura 24. Distribución de créditos por código de área. Fuente propia.

Según lo observado en la figura 24, existe una distribución desbalanceada de créditos pagos y no pagos por área o código zip, lo que puede incrementar el sesgo del algoritmo. Adicionalmente, el dominio del problema no se sitúa en ninguna de las áreas señaladas por la gráfica por lo que se procede a descartar las columnas de código de área y no hacerlas parte del análisis.

- **Transformación:** Para este caso se implementó un *pipeline* utilizando *sklearn* con la finalidad de realizar el escalamiento, la imputación de los datos faltantes y la codificación de los datos categóricos utilizando *One-Hot Encoding* o OHE. A continuación, en la tabla 9, se detalla el proceso de transformación para cada columna:

Columna	Tipo	Transformación	Observaciones
loan_status	*string*	*One-Hot Encoding*	Los valores *Fully Paid* y Charged-Off se mapean a ***1*** y ***0***, respectivamente; y el resultado se incluye dentro de esta misma columna.
term	*string*	*One-Hot Encoding*	Se codifican las dos categorías existentes en dos columnas: ['term_36_months', 'term_60_months'].
grade	*string*	*One-Hot Encoding*	Se codifican las categorías existentes en igual número de columnas.
sub_grade	*string*	*One-Hot Encoding*	Se codifican las categorías existentes en igual número de columnas.
emp_length	*string*	*One-Hot Encoding*	Se codifican las categorías existentes en igual número de columnas.
home_ownership	*string*	*One-Hot Encoding*	Se codifican las categorías existentes en igual número de columnas.
verification_status	*string*	*One-Hot Encoding*	Se codifican las categorías existentes en igual número de columnas.

purpose	*string*	*One-Hot Encoding*	Representa elementos del historial de crédito del aplicante.
initial_list_status	*string*	*One-Hot Encoding*	Representa elementos del historial de crédito del aplicante.
application_type	*string*	*One-Hot Encoding*	Representa elementos del historial de crédito del aplicante.
mort_acc	*float64*	Imputación simple y estandarización	*Teniendo en cuenta que 9.54% de sus datos son nulos, se imputan los datos faltantes y se* normalizan los valores para encajar en un rango de [0, 1].
loan_amnt	*float64*	Estandarización	Se normalizan los valores para encajar en un rango de [0, 1].
int_rate	*float64*	Estandarización	Se normalizan los valores para encajar en un rango de [0, 1].
installment	*float64*	Estandarización	Se normalizan los valores para encajar en un rango de [0, 1].
annual_inc	*float64*	Estandarización	Se normalizan los valores para encajar en un rango de [0, 1].
dti	*float64*	Estandarización	Se normalizan los valores para encajar en un rango de [0, 1].
emp_title	*string*	Eliminada	Al contar con ***173105*** valores posibles, se convierte en una columna de difícil manejo.
issue_d	*string*	Eliminada	Representa elementos del historial de crédito del aplicante.
title	*string*	Eliminada	Representa una descripción cruda del propósito del préstamo por lo que se considera como datos redundantes.
address	*string*	Eliminada	Se eliminó al ser extraído el código zip/
earliest_cr_line	*string*	Eliminada	Representa elementos del historial de crédito del aplicante.
open_acc	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.
pub_rec	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.
revol_bal	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.
revol_util	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.
total_acc	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.
pub_rec_bankruptcies	*float64*	Eliminada	Representa elementos del historial de crédito del aplicante.

Tabla 9: Resumen de la transformación de las características del dataset. Fuente propia.

Como una particularidad en este *dataset*, los créditos pagados son ***317696*** mientras que los no pagados son ***77523***, esto indica que el *dataset* está desbalanceado, por lo cual se realizó un balanceo utilizando la técnica de submuestreo o *undersampling* utilizando KNN con la librería *imblearn*.

- **Selección de características**: Una vez se completaron los pasos anteriores dentro de la ingeniería de características, se emplearon tres métodos distintos para la selección de aquellas características con mayor poder predictivo [96]. Todos los escenarios de prueba se realizaron entrenando un modelo *XGBoost* debido a que este es el modelo soportado por defecto por el *pipeline* de AWS.

  Los métodos empleados fueron:

  - ***Filtering*:** Se aplicó el método de *Chi-square*, donde se utilizaron los módulos *chi2* y *SelectKBest* de *sklearn*, que hacen parametrizar el número máximo de *features* deseados o *k*. Dicha variable se aumentó en saltos de 10 para validar los distintos resultados. A continuación se muestran los resultados obtenidos usando dicha técnica:

Método	Métricas	Resultados							
	k	*10*	*20*	*30*	*40*	*50*	*60*	*70*	*77*
Chi-Square	*Precision*	***0.7811***	***0.7832***	***0.7819***	***0.7865***	***0.7882***	***0.7881***	***0.7883***	***0.7879***
	Recall	***0.9432***	***0.9442***	***0.9467***	***0.9437***	***0.9423***	***0.9425***	***0.9433***	***0.9429***
	ROC AUC	***0.7360***	***0.7395***	***0.7410***	***0.7478***	***0.7537***	***0.7538***	***0.7542***	***0.7584***

Tabla 10: Resultados de la técnica filtering usando Chi-square. Fuente propia.

  - ***Wrapper*:** Se aplicó el método *Recursive Feature Elimination* o RFE, utilizando la metodología descrita en el aparte anterior se varió, en saltos de 10, el valor de *features* máximos deseados para conocer diversos resultados. El modelo utilizado para este método fue *XGBoost*. A continuación se detallan los resultados obtenidos:

Método	Métricas	Resultados							
	n	*10*	*20*	*30*	*40*	*50*	*60*	*70*	*77*
Recursive Feature Elimination	*Precision*	***0.7873***	***0.7884***	***0.7928***	***0.7944***	***0.7964***	***0.7961***	***0.7966***	***0.7969***
	Recall	***0.9326***	***0.9336***	***0.9351***	***0.9352***	***0.9322***	***0.9326***	***0.9324***	***0.9321***
	ROC AUC	***0.7426***	***0.7454***	***0.7521***	***0.7564***	***0.7551***	***0.7558***	***0.7552***	***0.7558***

Tabla 11: Resultados de la técnica wrapper usando Recursive Feature Elimination. Fuente propia.

  - ***Embedded*:** Se aplicaron los métodos *Least Absolute Shrinkage and Selection Operator* (LASSO) y de selección de *features* según su importancia en el entrenamiento para *XGBoost*. A continuación se

detallan los resultados obtenidos:

Método	Métricas	Resultados
LASSO	*Precision*	***0.7881***
	Recall	***0.9427***
	ROC AUC	***0.7584***
XGBoost	*Precision*	***0.7881***
	Recall	***0.9428***
	ROC AUC	***0.7532***

Tabla 12: Resultados de la técnica embedded usando LASSO y XGBoost. Fuente propia.

3.2.4. Balanceo de la fuente de datos

De acuerdo con la figura 23 de la sección 3.2.2 de este documento, es posible concluir que la fuente de datos presenta un desbalance respecto a la variable objetivo, ya que el ***80.4%*** de las observaciones han sido completamente pagadas, es decir, son préstamos positivos, mientras que el ***19.6%*** restante no fueron pagadas.

Este tipo de desbalance en las fuentes de datos hace que el modelo entrenado sea propenso a presentar problemas de sobreajuste, donde el modelo será muy poco efectivo al momento de realizar predicciones acertadas sobre datos no observados.

Para mitigar este problema, se utiliza un módulo conocido como *imblearn* [97], el cual ofrece los métodos requeridos para realizar el balanceo de los datos. Dentro de las opciones disponibles se encuentran las siguientes:

1. ***Random Under Sampling:*** En este proceso, la clase mayoritaria se ve disminuida, removiendo elementos que la componen de manera arbitraria [98].

2. ***Random Over Sampling:*** Al contrario del caso anterior, los elementos que componen la clase minoritaria se multiplican aleatoriamente [99].

3. ***All KNN:*** Es un tipo de *undersampling* que consiste en remover elementos de la clase mayoritaria basándose en el método de K vecinos más cercanos con el fin de escoger muestras de la clase mayoritaria de forma aleatoria y eliminar aquellas muestras más cercanas utilizando la distancia euclidiana entre ellas como criterio principal [100].

Aunque los métodos mencionados anteriormente tienen aplicación práctica, es importante mencionar que los dos primeros son capaces de introducir problemas de sobreajuste, debido a la reducción arbitraria de las dimensiones del *dataset*, lo cual

incrementa el riesgo de perder información valiosa. Adicionalmente, el aprendizaje puede ser deficiente, ya que el modelo se puede volver sesgado al distinguir mejor unas clases por encima de otras dentro del conjunto de datos.

Al hacer uso de *All KNN*, se hace una reducción de la dimensionalidad que mitiga los problemas anteriormente mencionados, al conservar en mayor medida la calidad de los datos. Sobre el *dataset* de *Lending Club* [93], de los ***396030*** registros iniciales, se obtiene una reducción de alrededor del ***23.55%*** de los datos obteniendo una nueva dimensionalidad de ***302738*** registros, de los cuales ***225065*** (frente a los ***318357*** iniciales) son créditos pagados completamente, mientras que los ***77673*** (cifra que se mantuvo) restantes no fueron pagos.

3.3. Modelado

Para determinar los algoritmos y técnicas de *Machine Learning* más apropiados para el contexto del problema, es necesario tomar en consideración los siguientes puntos:

1. Este caso de estudio requiere clasificar muestras de usuarios para determinar cuáles pueden acceder a un crédito y cuáles no.

2. Se requiere detectar las solicitudes de usuarios que realmente tienen capacidad para pagar un crédito y minimizar el riesgo de impago.

3. La implementación de modelos distintos al ofrecido por defecto por la plantilla de AWS, comprende un costo técnico adicional debido a que estas implementaciones requieren la creación y despliegue de un contenedor de Docker para albergar el modelo que va a ser entrenado [101].

Teniendo en cuenta los puntos 1 y 2, es posible determinar que se requiere implementar un algoritmo de clasificación binaria. Los criterios para determinar el conjunto de métricas adecuadas parten del siguiente análisis:

- Para conseguir detectar las solicitudes de personas que realmente tienen capacidad para pagar un crédito, se debe mantener un umbral de selección alto por lo que es probable que se rechacen muchas buenas solicitudes pero aceptando solicitudes más seguras. En este caso se prioriza el valor de *precision* por encima del valor de *recall*.

- Si son aceptadas muchas solicitudes que podrían parecer buenas, a costo de aceptar algunas solicitudes realmente riesgosas y que podrían entrar en mora. En este caso, a diferencia del anterior, se prioriza el valor de *recall* por encima del valor de *precision*. Por tanto, esta aproximación plantea un inconveniente al requerir de una verificación adicional de las solicitudes para mitigar el riesgo.

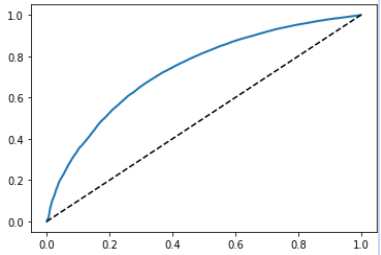
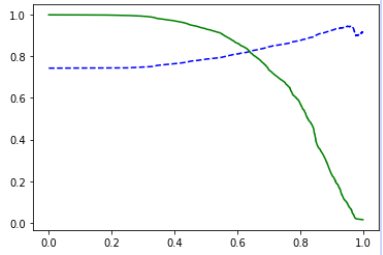
Desafortunadamente, no es posible favorecer alguno de los dos valores sin afectar al otro, es decir: incrementar el *recall* reduce el valor de *precision* y de forma análoga, aumentar el valor de *precision* reduce el *recall*. Por lo tanto, el criterio principal para seleccionar un modelo será el valor de *precision*, ya que permite minimizar el riesgo sin incrementar la complejidad técnica o de negocio.

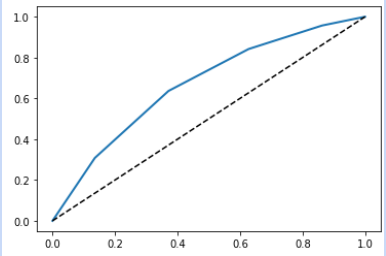
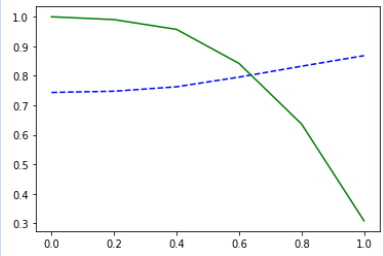
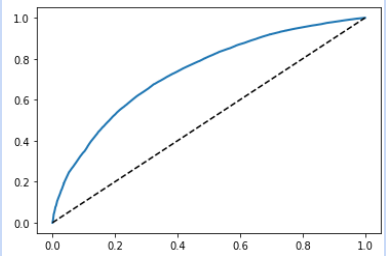
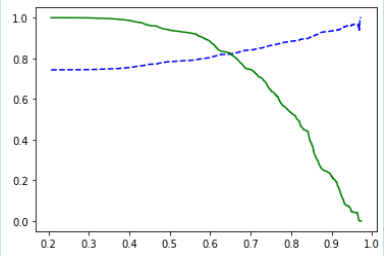
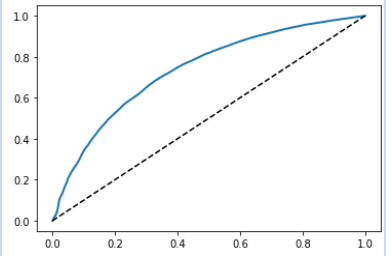
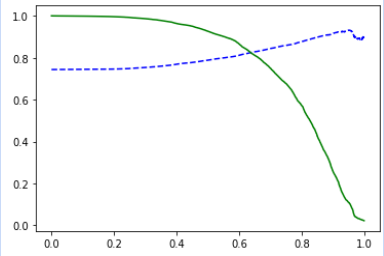
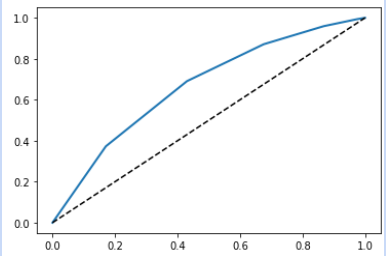
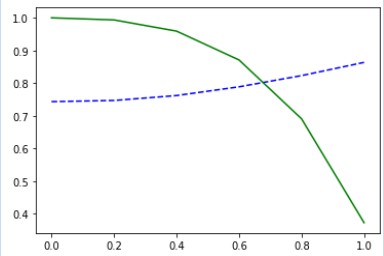
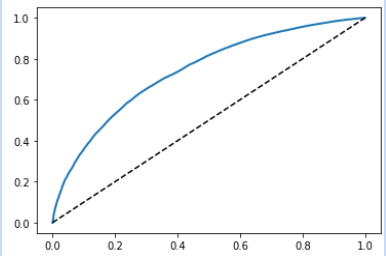
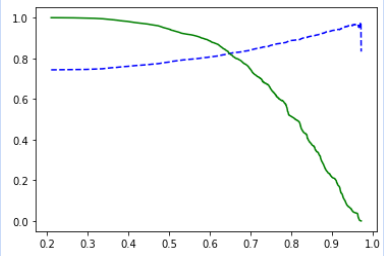
Finalmente, la escogencia de un algoritmo de clasificación se ve limitada por el costo técnico que se escapa del alcance de este piloto, dejando a *XGBoost*, el modelo por defecto ofrecido por el *pipeline* de AWS para *MLOps* como el algoritmo seleccionado. Sin embargo, también se van a evaluar algunos de los algoritmos con mejores resultados dentro de la bibliografía, como lo son *Random Forest*, *K-Nearest Neighbors*, con el fin de conocer las métricas para este caso de estudio.

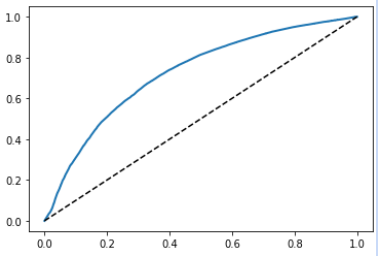
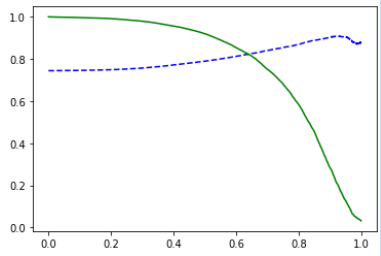
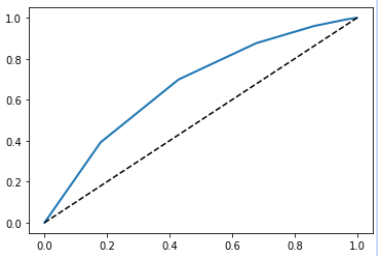
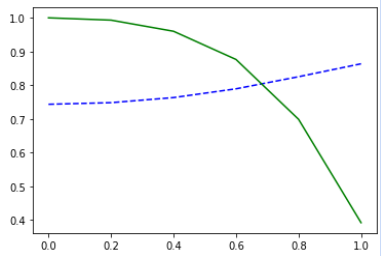
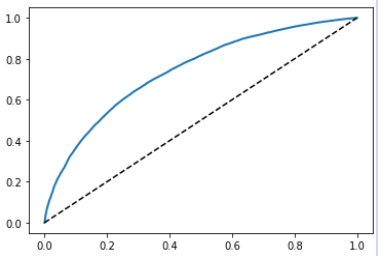
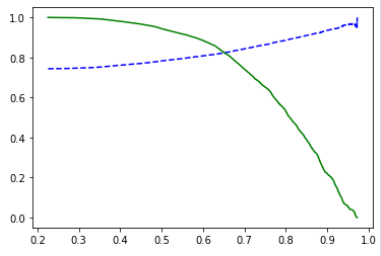
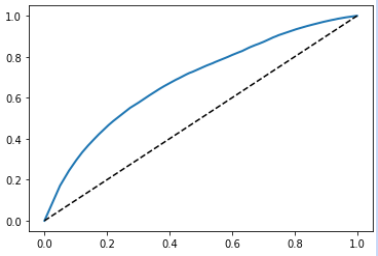
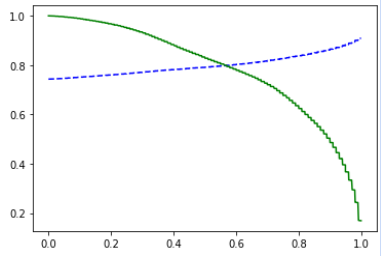
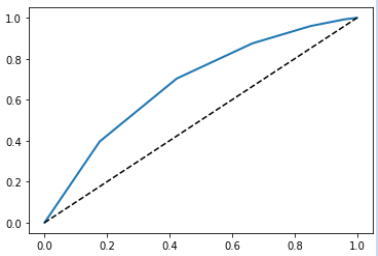
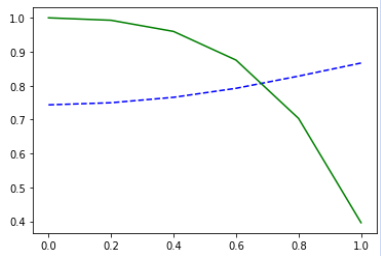
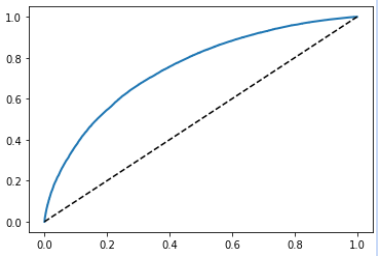
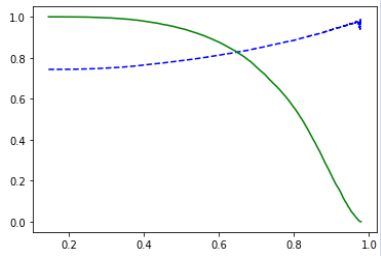
Nota: *Con el fin de iterar más ágilmente en la creación de un modelo inicial, se procede a utilizar la herramienta de Google Colab, que permite documentar, editar y ejecutar notebooks de Jupyter Python.*

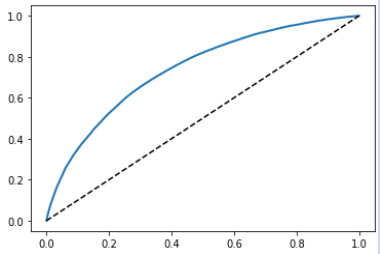
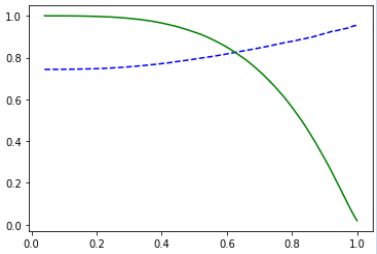
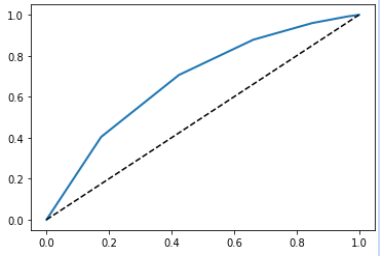
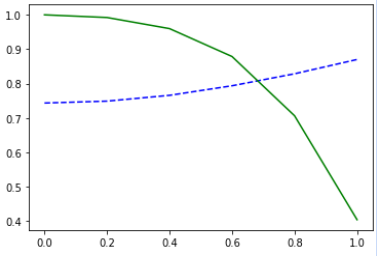
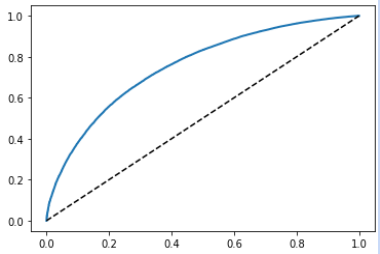
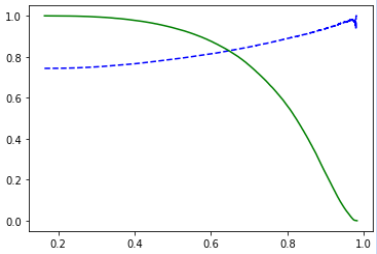
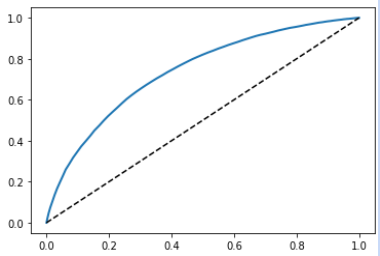
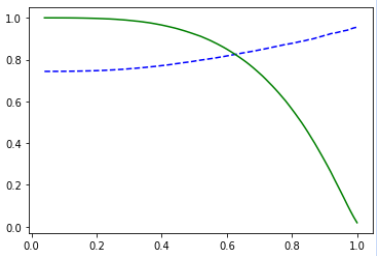
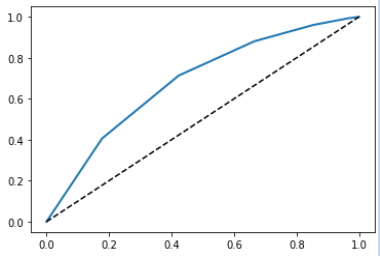
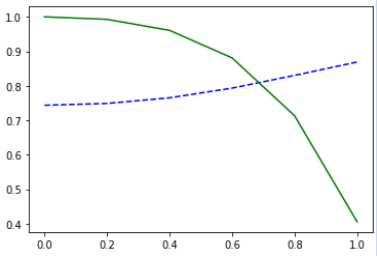
Este proceso tiene como entrada el *dataset* preprocesado, el cual se divide en dos partes para entrenamiento (***67%***) y validación (***33%***). Para obtener los subconjuntos de datos se utiliza el módulo *sklearn.model_selection.StratifiedShuffleSplit* [102]. Dicho módulo realiza una subdivisión de 10 (Valor por defecto) pliegues estratificados o repartidos de forma homogénea del *dataset*, con el fin de mantener la relación de créditos pagados y no pagados en ambos subconjuntos [103]. De forma complementaria, se hace uso del módulo *sklearn.model_selection.cross_val_predict*, el cual permite generar estimados validados de forma cruzada para cada punto de datos. Cabe mencionar que los datos pueden dividirse en partes iguales, según sea la necesidad [104].

Teniendo en cuenta el proceso de ingeniería de características realizado en el punto 3.2.3 de este documento, se procede a realizar las pruebas sobre los tres tipos de modelos mencionados anteriormente, haciendo uso de las diferentes técnicas de selección de características. Los resultados se detallan a continuación:

Modelo	**Métricas**	**ROC AUC**	***Precision — Recall vs Threshold***
Filtrado con Chi-Square y K = 10			
Random Forest	***Precision**: 0.7869* ***Recall**: 0.9360* ***ROC AUC**: 0.7362*		

K-Nearest Neighbors	***Precision:*** *0.7880* ***Recall:*** *0.8704* ***ROC AUC:*** *0.6676*		
XGBoost	***Precision:*** *0.7811* ***Recall:*** *0.9432* ***ROC AUC:*** *0.7360*		
Filtrado con Chi-Square y K = 20			
Random Forest	***Precision:*** *0.7879* ***Recall:*** *0.9362* ***ROC AUC:*** *0.7344*		
K-Nearest Neighbors	***Precision:*** *0.7931* ***Recall:*** *0.8702* ***ROC AUC:*** *0.6670*		
XGBoost	***Precision:*** *0.7832* ***Recall:*** *0.9442* ***ROC AUC:*** *0.7395*		
Filtrado con Chi-Square y K = 30			

Random Forest	***Precision:*** *0.7894* ***Recall:*** *0.9284* ***ROC AUC:*** *0.7243*		
K-Nearest Neighbors	***Precision:*** *0.7911* ***Recall:*** *0.8605* ***ROC AUC:*** *0.6718*		
XGBoost	***Precision:*** *0.7819* ***Recall:*** *0.9467* ***ROC AUC:*** *0.7410*		
Filtrado con Chi-Square y K = 40			
Random Forest	***Precision:*** *0.7916* ***Recall:*** *0.8220* ***ROC AUC:*** *0.6861*		
K-Nearest Neighbors	***Precision:*** *0.7928* ***Recall:*** *0.8796* ***ROC AUC:*** *0.6785*		
XGBoost	***Precision:*** *0.7865* ***Recall:*** *0.9437* ***ROC AUC:*** *0.7487*		

Filtrado con Chi-Square y K = 50			
Random Forest	***Precision:*** *0.7954* ***Recall:*** *0.9165* ***ROC AUC:*** *0.7381*		
K-Nearest Neighbors	***Precision:*** *0.7955* ***Recall:*** *0.8800* ***ROC AUC:*** *0.6824*		
XGBoost	***Precision:*** *0.7882* ***Recall:*** *0.9423* ***ROC AUC:*** *0.7537*		
Filtrado con Chi-Square y K = 60			
Random Forest	***Precision:*** *0.7942* ***Recall:*** *0.9203* ***ROC AUC:*** *0.7419*		
K-Nearest Neighbors	***Precision:*** *0.7950* ***Recall:*** *0.8819* ***ROC AUC:*** *0.6835*		

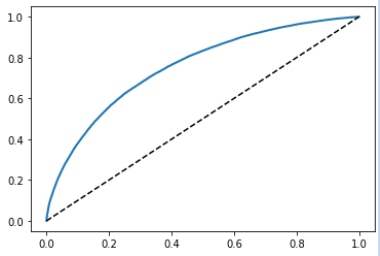
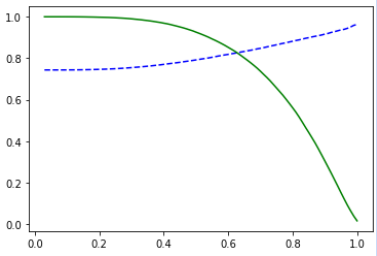
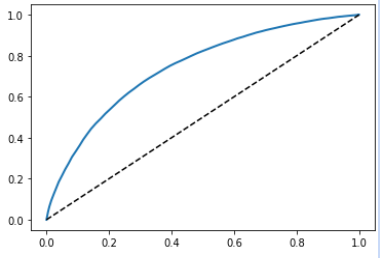
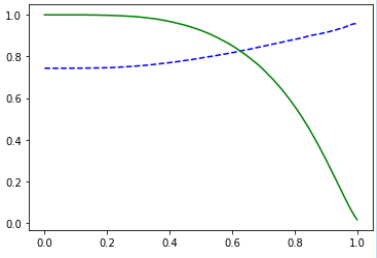
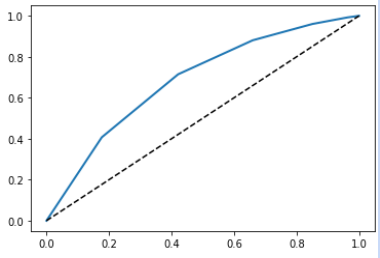
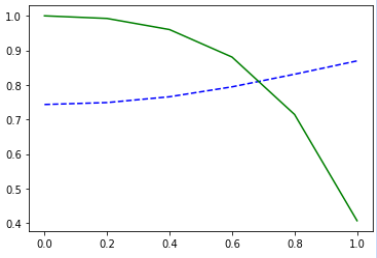
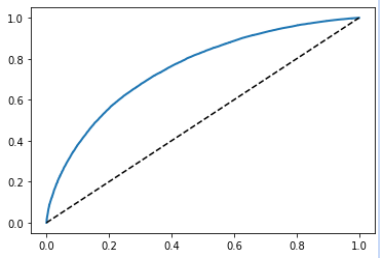
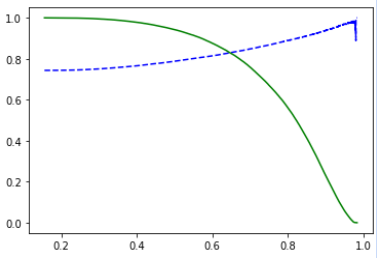
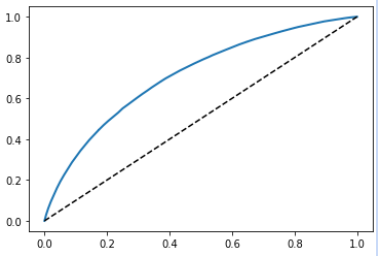
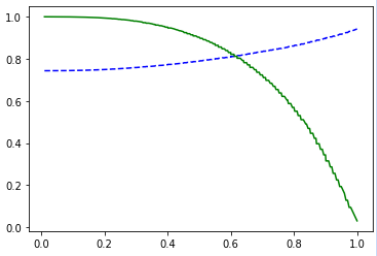
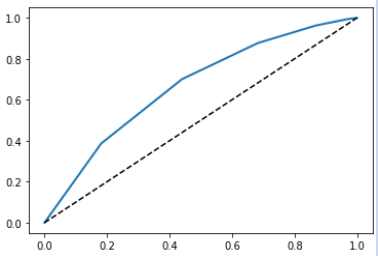
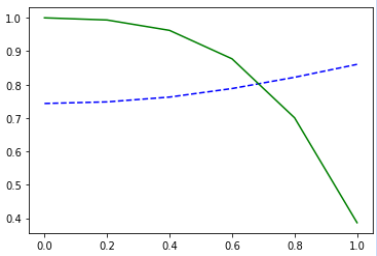
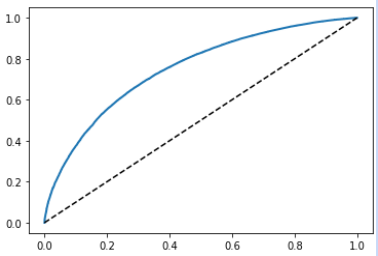
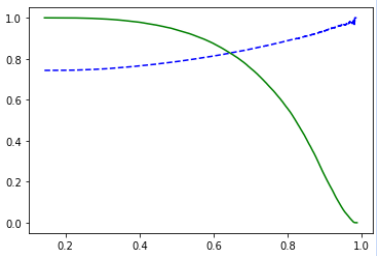
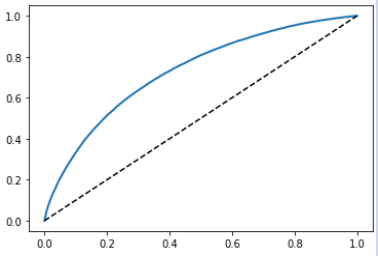
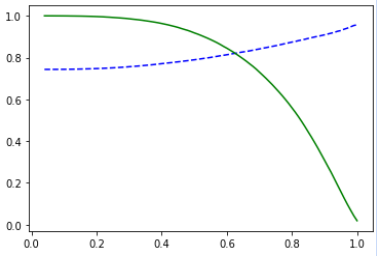
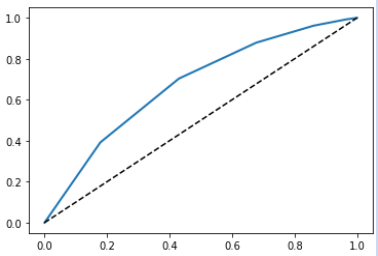
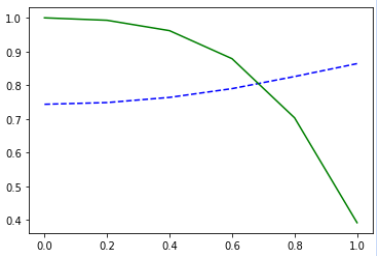
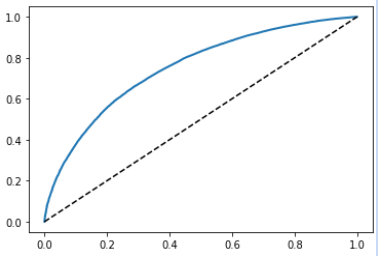
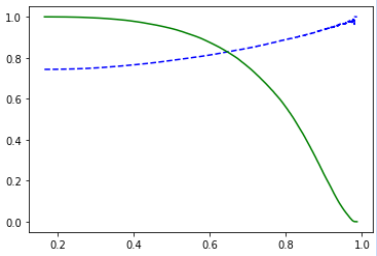
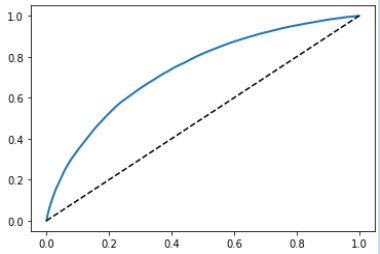
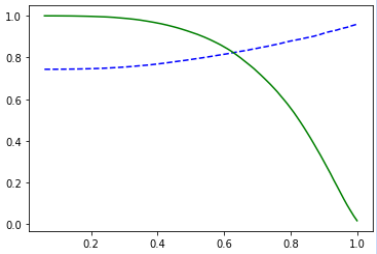
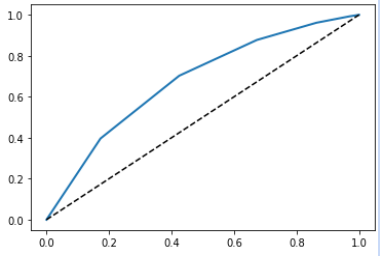
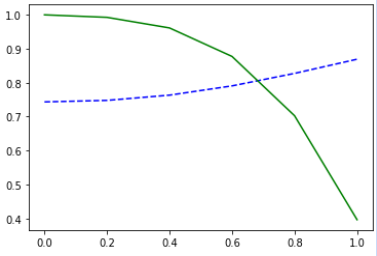
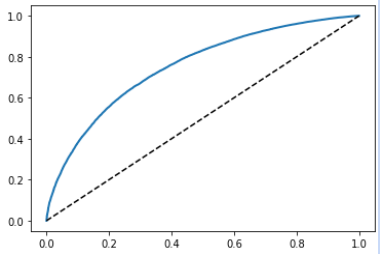
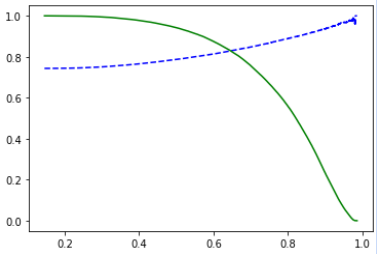
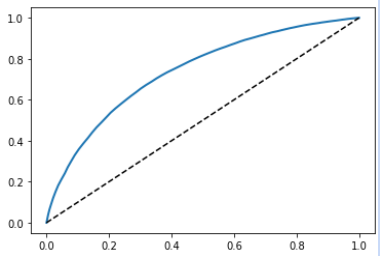
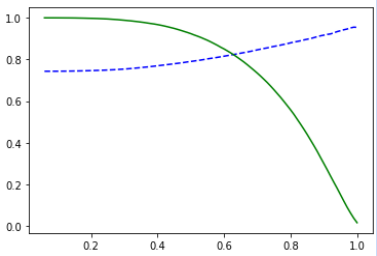
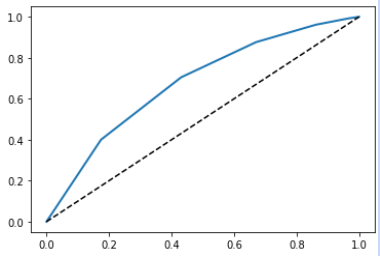
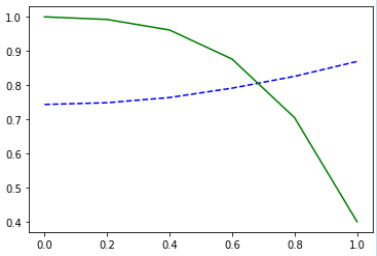
XGBoost	***Precision:*** *0.7882* ***Recall:*** *0.9425* ***ROC AUC:*** *0.7539*		
Filtrado con Chi-Square y K = 70			
Random Forest	***Precision:*** *0.7953* ***Recall:*** *0.9197* ***ROC AUC:*** *0.7420*		
K-Nearest Neighbors	***Precision:*** *0.7959* ***Recall:*** *0.8820* ***ROC AUC:*** *0.6856*		
XGBoost	***Precision:*** *0.7883* ***Recall:*** *0.9424* ***ROC AUC:*** *0.7538*		

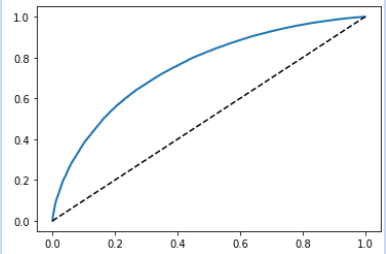
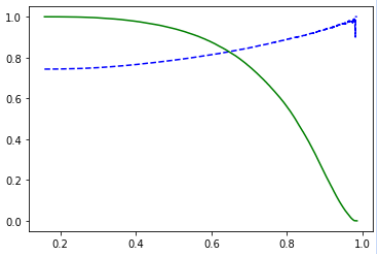
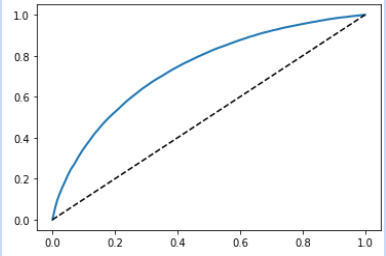
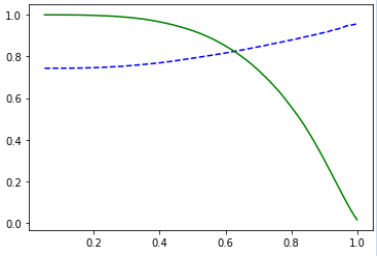
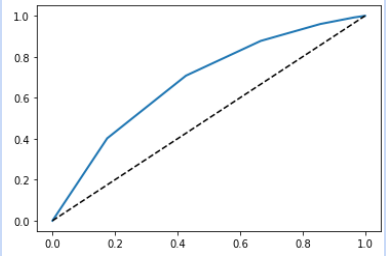
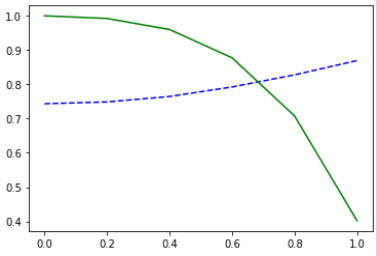
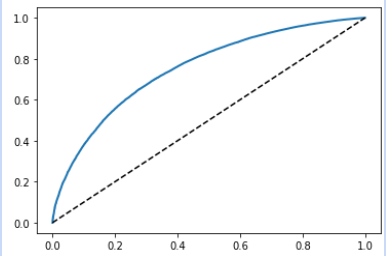
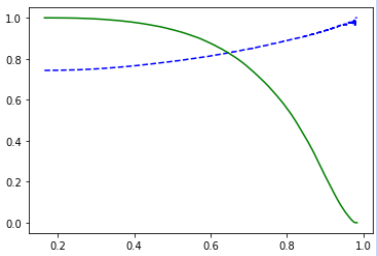
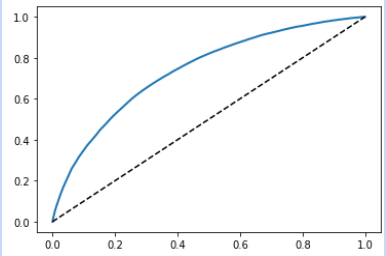
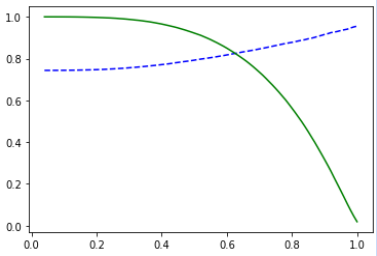
Tabla 13: Resultados de la técnica de filtrado usando Chi-Square combinada con Random Forest, K-Nearest Neighbors y XGBoost. Fuente propia.

Análogamente, se probaron los modelos planteados utilizando los subconjuntos obtenidos a partir de la selección de características utilizando *Recursive Feature Elimination*, detallando los resultados en la siguiente tabla:

Modelo	Métricas	ROC AUC	*Precision — Recall vs Threshold*
Filtrado con Recursive Feature Elimination y n = 10			

Random Forest	***Precision:*** *0.7903* ***Recall:*** *0.8958* ***ROC AUC:*** *0.7115*		
K-Nearest Neighbors	***Precision:*** *0.7875* ***Recall:*** *0.8803* ***ROC AUC:*** *0.6672*		
XGBoost	***Precision:*** *0.7872* ***Recall:*** *0.9415* ***ROC AUC:*** *0.7506*		
Filtrado con Recursive Feature Elimination y n = 20			
Random Forest	***Precision:*** *0.7916* ***Recall:*** *0.9126* ***ROC AUC:*** *0.7285*		
K-Nearest Neighbors	***Precision:*** *0.7909* ***Recall:*** *0.8814* ***ROC AUC:*** *0.6738*		
XGBoost	***Precision:*** *0.7880* ***Recall:*** *0.9413* ***ROC AUC:*** *0.7518*		

Filtrado con Recursive Feature Elimination y n = 30			
Random Forest	***Precision:*** *0.7925* ***Recall:*** *0.9162* ***ROC AUC:*** *0.7348*		
K-Nearest Neighbors	***Precision:*** *0.7908* ***Recall:*** *0.8803* ***ROC AUC:*** *0.6780*		
XGBoost	***Precision:*** *0.7874* ***Recall:*** *0.9422* ***ROC AUC:*** *0.7525*		
Filtrado con Recursive Feature Elimination y n = 40			
Random Forest	***Precision:*** *0.7931* ***Recall:*** *0.9191* ***ROC AUC:*** *0.7372*		
K-Nearest Neighbors	***Precision:*** *0.7919* ***Recall:*** *0.8807* ***ROC AUC:*** *0.6776*		

XGBoost	***Precision:*** *0.7875* ***Recall:*** *0.9421* ***ROC AUC:*** *0.7524*		
Filtrado con Recursive Feature Elimination y n = 50			
Random Forest	***Precision:*** *0.7945* ***Recall:*** *0.9182* ***ROC AUC:*** *0.7379*		
K-Nearest Neighbors	***Precision:*** *0.7936* ***Recall:*** *0.8808* ***ROC AUC:*** *0.6803*		
XGBoost	***Precision:*** *0.7874* ***Recall:*** *0.9423* ***ROC AUC:*** *0.7525*		
Filtrado con Recursive Feature Elimination y n = 60			
Random Forest	***Precision:*** *0.7951* ***Recall:*** *0.9168* ***ROC AUC:*** *0.7402*		

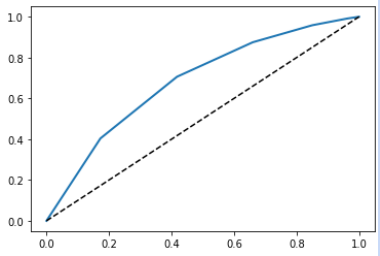
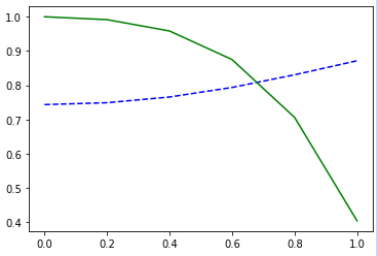
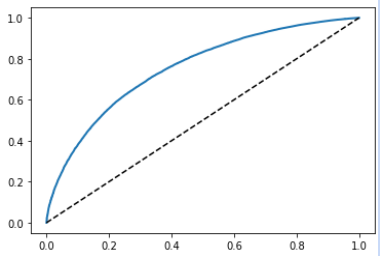
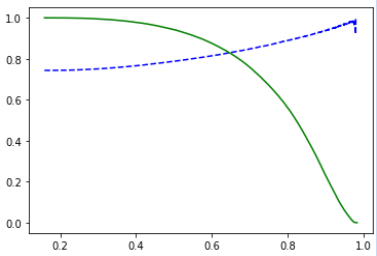
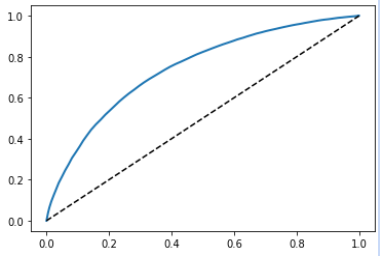
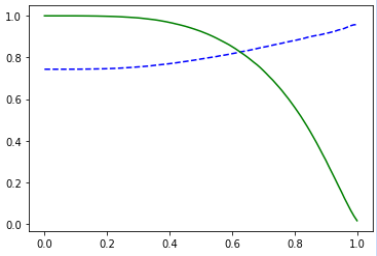
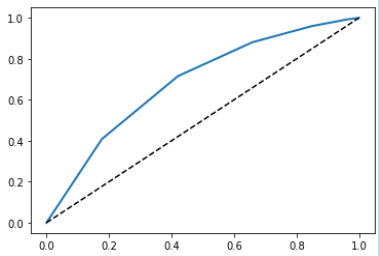
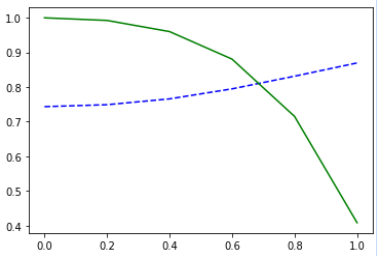
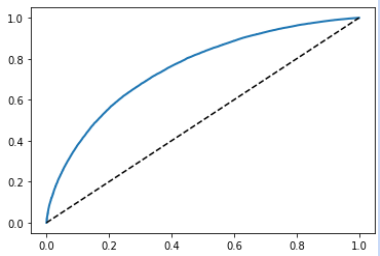
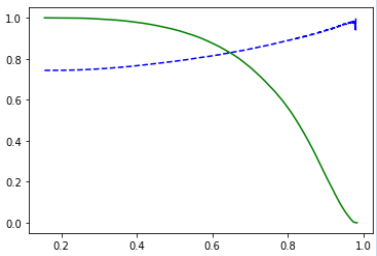
K-Nearest Neighbors	***Precision:*** *0.7956* ***Recall:*** *0.8805* ***ROC AUC:*** *0.6837*		
XGBoost	***Precision:*** *0.7883* ***Recall:*** *0.9422* ***ROC AUC:*** *0.7537*		
Filtrado con Recursive Feature Elimination y n = 70			
Random Forest	***Precision:*** *0.7952* ***Recall:*** *0.9204* ***ROC AUC:*** *0.7422*		
K-Nearest Neighbors	***Precision:*** *0.7957* ***Recall:*** *0.8822* ***ROC AUC:*** *0.6862*		
XGBoost	***Precision:*** *0.7883* ***Recall:*** *0.9428* ***ROC AUC:*** *0.7538*		

Tabla 14: Resultados de la técnica de filtrado usando Recursive Feature Elimination combinada con Random Forest, K-Nearest Neighbors y XGBoost. Fuente propia.

Teniendo en cuenta los resultados plasmados en las tablas 13 y 14, se concluye que la mejor combinación entre el modelo y el método de selección de características, dado el criterio de *precision* sobre *recall*, es *Chi-Square* con *K-Nearest Neighbors*, cuando el valor de k es igual a 70. Sin embargo, debido a las implicaciones técnicas que

representa dicha implementación se opta por elegir *Chi-Square* con *XGBoost* cuando k es igual a 70 para llevar a cabo las pruebas de este piloto.

Finalmente, el paso a paso para llevar esta implementación al *pipeline* de AWS para *MLOps* se detalla en el Anexo B de este documento.

4. RESULTADOS

Una vez seleccionado el algoritmo y el método de selección de características adecuado, y después de haber modificado la plantilla de AWS para *MLOps* con los cambios pertinentes, es necesario realizar el despliegue del *pipeline* implementado con el fin de realizar las tareas de preprocesamiento, entrenamiento, validación y pruebas en el entorno de AWS.

Es importante resaltar que no se cuenta con la capacidad de obtener datos reales para probar esta implementación, por este motivo se debe de hacer uso de los datos del propio *dataset*. Éstos se obtienen a partir de la división del *dataset* que se utilizó para el entrenamiento y la validación. A continuación se detallan las proporciones y los porcentajes de división:

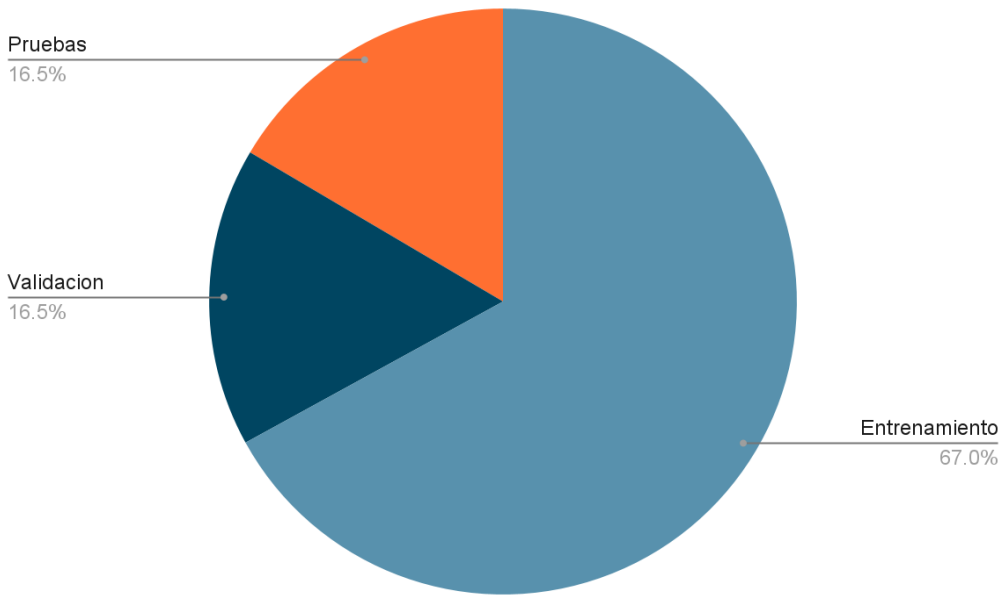


Figura 25. División del dataset utilizada para propósitos de pruebas. Fuente propia.

Una vez ajustada la distribución del *dataset*, y aplicados los cambios pertinentes sobre la plantilla de AWS para *MLOps* escogida, se debe de seguir el procedimiento descrito en el Anexo C de este documento con la finalidad de realizar el despliegue del *pipeline* implementado.

A continuación, se exponen los resultados obtenidos a nivel técnico, con la verificación del despliegue y el consumo del servicio, y a nivel de negocio:

4.1. Verificación del despliegue

Una vez llevado a cabo el proceso definido en el Anexo C, es necesario realizar una validación del despliegue para asegurar que la implementación se encuentra funcional. Para validar que los recursos requeridos estén correctamente provisionados, es necesario dirigirse a *CloudFormation* para verificar la presencia del modelo y del punto de inferencia, el resultado debe lucir de la siguiente manera:

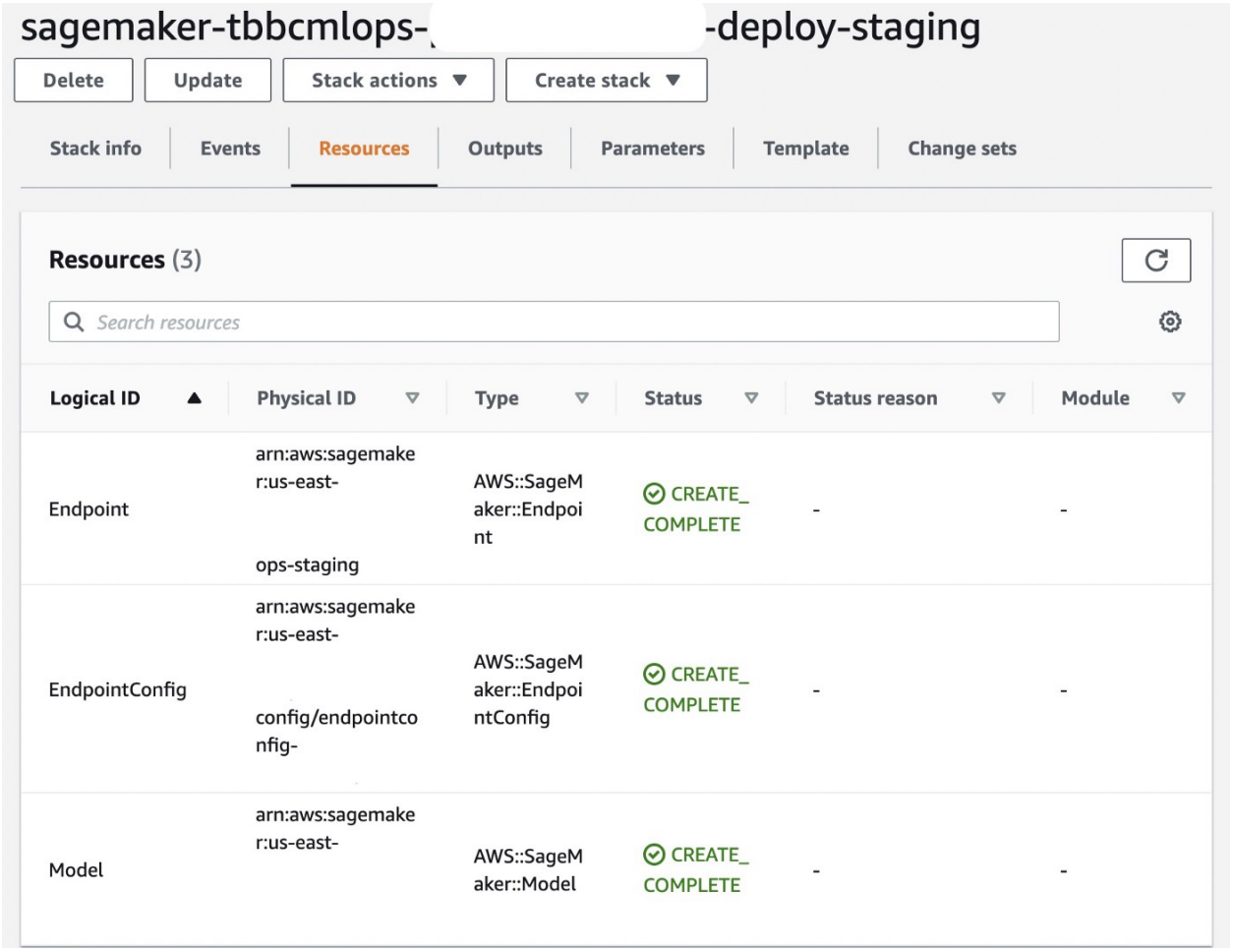


Figura 26: Recursos en CloudFormation: Modelo y punto de consulta. Tomado de [105].

Finalmente, se consulta si el punto o *endpoint* se encuentra en servicio y disponible para ser consultado. El resultado debe lucir de la siguiente manera:

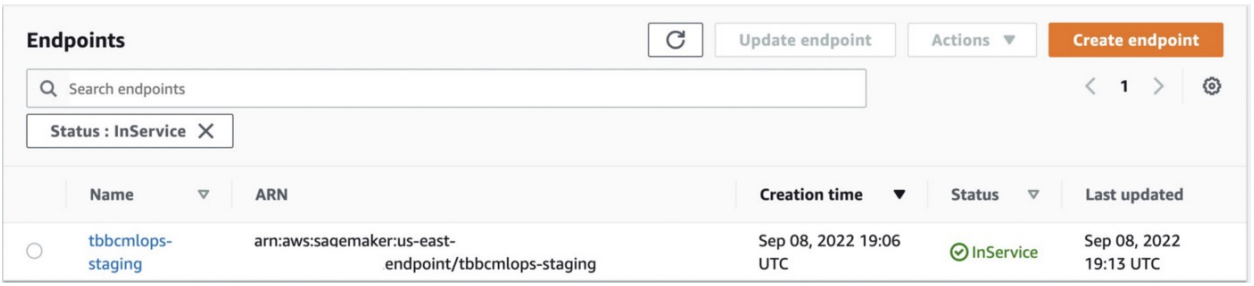


Figura 27: Punto de consulta listo para ser invocado. Tomado de [105].

Ya que el despliegue ha sido verificado, se procede a validar el comportamiento del modelo descrito en el siguiente inciso a través del *endpoint* de inferencia.

4.2. Consumo del servicio

Con el fin de realizar pruebas sobre el modelo desplegado se hace uso del módulo *boto3* de *Python* con el fin de generar una sesión que permita realizar el consumo del *endpoint* desplegado, proveyendo un arreglo de datos que represente un registro del *dataset* con el que se alimentó el modelo, igual al que se ve en la siguiente figura:

```
import boto3

session = boto3.Session(profile_name='dev', region_name='us-east-1')
runtime= session.client('runtime.sagemaker')

result  = runtime.invoke_endpoint(
    EndpointName='mlops-staging',
    Body="1.,1.,1.,1.,0.,1.,1.,0.,1.,0.,1.,1.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,
0.,0.,0.,0.,0.,0.,0.,0.,0.,1.,0.,0.,1.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,1.,0.,1.,0.,1.,0.,0.,0.,0.,0.,0.,0.,0
.,0.,0.,1.,0.,0.",
    ContentType='text/csv'
)

print(result['Body'].read().decode('utf-8'))
```

Figura 28: Pieza de código utilizada para consultar el endpoint de inferencia. Tomado de [105].

Una vez ejecutada esta pieza de código con los valores provistos dentro del apartado *Body*, se obtiene la respuesta del servicio, siendo la siguiente en este caso:

```
➜  tester_endpoint python3.9 main.py
0.9271702170372009
```

Figura 29: Resultado de la consulta del endpoint de inferencia. Tomado de [105].

El valor retornado en la figura 29 se refiere a que el registro que contiene los valores consultados tiene un ***92.71%*** de pertenecer a la clase mayoritaria, es decir, tiene altas probabilidades de pagar el préstamo. En el anexo D de este documento, se presenta una aplicación práctica donde se consume el *endpoint* generado desde una aplicación web.

4.3. A nivel del negocio

Para conocer el desempeño de la implementación final, se consulta el archivo *evaluation.json*, disponible en la carpeta *outputs* del *bucket* de AWS S3 asociado con este proyecto, con el fin de conocer las métricas de rendimiento como se observa en la siguiente figura:

Figura 30: Métricas de desempeño del pipeline desplegado. Fuente propia.

Teniendo en cuenta los resultados obtenidos en los numerales anteriores y a pesar de que estos resalten por demostrar un desempeño notable del modelo, se puede observar que, para el caso del *dataset* de *Lending Club*, el dejar de lado la dependencia de los datos relacionados con el historial de crédito implica sacrificar el desempeño como lo demuestra la figura 30, debido a que un valor de *recall* tan alto implica que el modelo tiende a generalizar a los buenos aplicantes de crédito, es decir aquellos que tienen mayor probabilidad de pagar el crédito solicitado, a costo de pasar algunas malas solicitudes, en lugar de ofrecer predicciones más seguras que proporcionen un mayor nivel de certeza.

5. CONCLUSIONES, APORTES Y LECCIONES APRENDIDAS

A partir de las fuentes de datos conocidas a partir de la bibliografía y de la búsqueda particular, se determinó que el *dataset* de *Lending Club* [93] es el que mejor representa las normas del negocio definidas por Wizit para la construcción y el entrenamiento del modelo de datos, debido al número de registros y a los datos contenidos en el. Los cuales permiten obtener un panorama importante sobre los créditos otorgados.

Adicionalmente, se realizó la exploración de los servicios web ofrecidos por AWS, destacando aquellos que representan mayor valor para las implementaciones que hacen uso de la metodología de *MLOps*, entre los que resaltan: *AWS CodeCommit*, *AWS CodePipeline*, *Amazon S3*, *Amazon SageMaker* y *AWS CloudFormation*. Dichos servicios hacen parte integral del *pipeline* ya que permiten llevar a cabo los procesos de preprocesamiento, entrenamiento, evaluación y despliegue de un proyecto de *MLOps*, resultando en la implementación del modelo de datos y del andamiaje del *pipeline* de *MLOps* diseñado en torno al *dataset* de *Lending Club* [93]. El montaje se realizó haciendo uso de la plantilla "*MLOps template for model building, training, and deployment"* [75].

Como resultado de este trabajo de investigación, se puede concluir que no es posible realizar una predicción confiable sobre la probabilidad de impago sin hacer uso de los datos del historial de crédito de un solicitante. Teniendo en cuenta los valores de *recall* y *precision* ilustrados en la figura 30, adicional a los resultados mostrados en las tablas 13 y 14 para modelos diferentes a *XGBoost*, es necesario incluir algún medio de validación complementario a la predicción del modelo, como puede ser apoyo humano.

5.1. Lecciones aprendidas

Es importante hacer seguimiento de la facturación cuando se utilicen servicios web orientados a *Machine Learning*, debido a que las máquinas utilizadas en estos procesos suelen tener una alta capacidad de cómputo. Su uso, o no uso, aumentará el costo por cada hora, con lo cual, es altamente recomendable apagar las máquinas asociadas a estas tareas cuando su funcionamiento no sea necesario.

Por su versatilidad es recomendable hacer uso de *Google Colab*, al momento de crear *notebooks* para realizar análisis exploratorio de datos, preprocesamiento, entrenamiento o cualquier otro tipo de procedimiento, ya que es un servicio intuitivo y gratuito. Su símil en AWS sería *Jupyter Lab*, que dispone de un conjunto variado de herramientas y funcionalidades muy parecidas a las que ofrece *Google Colab*, sin embargo con el tiempo, puede significar costos según el uso que tengan.

5.2. Trabajos futuros

Teniendo en cuenta las limitaciones técnicas que han sido mencionadas en este trabajo respecto a las implicaciones de usar modelos diferentes al incluido por defecto, se plantea la posibilidad de construir un *pipeline* que incluya la aplicación de modelos como *Random Forest*, por ejemplo.

Adicionalmente, se plantea explorar diversos proveedores de servicios web como lo son *Microsoft Azure* y *Google Cloud Platform*, con el fin de identificar las diferencias que provee cada plataforma para determinar cual de estas es la mejor opción para los proyectos según sus requerimientos particulares.

Finalmente, se propone el uso de datos alternativos para realizar las predicciones, utilizando diversas fuentes como lo son las redes sociales, pasarelas de pago, entre otros datos.

6. REFERENCIAS

[1] “Artificial Intelligence & Autopilot”, *Tesla*. https://www.tesla.com/AI (consultado el 10 de septiembre de 2021).

[2] “Siemens Showcases AI-Pathway for Lung Cancer at HIMSS 2021”. https://appliedradiationoncology.com/articles/siemens-showcases-ai-pathway-for-lung-cancer-at-himss-2021 (consultado el 10 de septiembre de 2021).

[3] “My Credit Score”, el 21 de abril de 2016. https://www.experian.com/blogs/ask-experian/credit-education/score-basics/my-credit-score/ (consultado el 11 de diciembre de 2021).

[4] “Reporte de la situación del crédito en Colombia - Junio de 2021”, *Banco de la República (banco central de Colombia)*, el 2 de agosto de 2021. https://www.banrep.gov.co/es/reporte-situacion-del-credito-colombia-junio-2021 (consultado el 11 de diciembre de 2021).

[5] “Valora, “El 76 % de créditos en Colombia a personas con bajo puntaje crediticio lo dan las fintech”, *Valora Analitik*, el 20 de septiembre de 2020. https://www.valoraanalitik.com/2020/09/20/el-76-de-creditos-en-colombia-a-personas-con-bajo-puntaje-crediticio-lo-dan-las-fintech/ (consultado el 11 de diciembre de 2021).

[6] “What is Machine Learning?”, el 6 de julio de 2022. https://www.ibm.com/cloud/learn/machine-learning (consultado el 26 de noviembre de 2022).

[7] “What is a Decision Tree | IBM”. https://www.ibm.com/topics/decision-trees (consultado el 26 de noviembre de 2022).

[8] “What is Random Forest?”, el 26 de enero de 2021. https://www.ibm.com/cloud/learn/random-forest (consultado el 26 de noviembre de 2022).

[9] F. Eshragh, M. Pooyandeh, y D. J. Marceau, “Automated negotiation in environmental resource management: Review and assessment”, *J. Environ. Manage.*, vol. 162, pp. 148–157, oct. 2015, doi: 10.1016/j.jenvman.2015.07.051.

[10] “What is Logistic regression? | IBM”. https://www.ibm.com/topics/logistic-regression (consultado el 26 de noviembre de 2022).

[11] “What is XGBoost?”, *NVIDIA Data Science Glossary*. https://www.nvidia.com/en-us/glossary/data-science/xgboost/ (consultado el 26 de noviembre de 2022).

[12] “What is Exploratory Data Analysis?”, el 6 de julio de 2021. https://www.ibm.com/cloud/learn/exploratory-data-analysis (consultado el 10 de julio de 2022).

[13] A. Zheng y A. Casari, “Feature Engineering for Machine Learning [Book]”. https://www.oreilly.com/library/view/feature-engineering-for/9781491953235/ (consultado el 27 de junio de 2022).

[14] “Least Absolute Shrinkage and Selection Operator - an overview | ScienceDirect Topics”. https://www.sciencedirect.com/topics/engineering/least-absolute-shrinkage-and-selection-operator (consultado el 3 de diciembre de 2022).

[15] “What is a Pipeline? - Definition from Techopedia”, *Techopedia.com*. http://www.techopedia.com/definition/5312/pipeline (consultado el 5 de febrero de 2023).
[16] S. Zhang, C. Zhang, y Q. Yang, “Data preparation for data mining”, *Appl. Artif. Intell.*, vol. 17, núm. 5–6, pp. 375–381, may 2003, doi: 10.1080/713827180.
[17] D. Pyle, *Data Preparation for Data Mining*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.
[18] “Entrenamiento de modelos de ML - Amazon Machine Learning”. https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/training-ml-models.html (consultado el 27 de septiembre de 2022).
[19] “Evaluación de la precisión del modelo - Amazon Machine Learning”. https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/evaluating-model-accuracy.html (consultado el 27 de septiembre de 2022).
[20] “What is Model Deployment”, *Iguazio*. https://www.iguazio.com/glossary/model-deployment/ (consultado el 15 de octubre de 2022).
[21] G. James, D. Witten, T. Hastie, y R. Tibshirani, “Resampling Methods”, en *An Introduction to Statistical Learning: with Applications in R*, G. James, D. Witten, T. Hastie, y R. Tibshirani, Eds. New York, NY: Springer US, 2021, pp. 197–223. doi: 10.1007/978-1-0716-1418-1_5.
[22] “What is DevOps? - Amazon Web Services (AWS)”, *Amazon Web Services, Inc.* https://aws.amazon.com/devops/what-is-devops/ (consultado el 27 de junio de 2022).
[23] “What is CI/CD? | GitLab”. https://about.gitlab.com/topics/ci-cd (consultado el 27 de junio de 2022).
[24] “Machine Learning Glossary”, *Google Developers*. https://developers.google.com/machine-learning/glossary (consultado el 10 de septiembre de 2021).
[25] “Glossary of Terms”, *Mach. Learn.*, vol. 30, núm. 2, pp. 271–274, feb. 1998, doi: 10.1023/A:1017181826899.
[26] “Classification: Precision and Recall | Machine Learning Crash Course”, *Google Developers*. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall (consultado el 27 de junio de 2022).
[27] A. A. Taha y A. Hanbury, “Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool”, *BMC Med. Imaging*, vol. 15, núm. 1, p. 29, ago. 2015, doi: 10.1186/s12880-015-0068-x.
[28] “Classification: ROC Curve and AUC | Machine Learning”, *Google Developers*. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc (consultado el 26 de septiembre de 2022).
[29] “sklearn.metrics.f1_score”, *scikit-learn*. https://scikit-learn/stable/modules/generated/sklearn.metrics.f1_score.html (consultado el 23 de octubre de 2022).
[30] C. E. Metz, “Basic principles of ROC analysis”, *Semin. Nucl. Med.*, vol. 8, núm. 4, pp. 283–298, oct. 1978, doi: 10.1016/S0001-2998(78)80014-2.
[31] W. H. Kruskal y W. A. Wallis, “Use of Ranks in One-Criterion Variance Analysis”,

J. Am. Stat. Assoc., vol. 47, núm. 260, pp. 583–621, 1952, doi: 10.2307/2280779.
[32] S. Turney, “Chi-Square (X^2) Tests | Types, Formula & Examples”, *Scribbr*, el 23 de mayo de 2022. https://www.scribbr.com/statistics/chi-square-tests/ (consultado el 5 de noviembre de 2022).
[33] “ml-ops.org”. https://ml-ops.org/ (consultado el 10 de septiembre de 2021).
[34] “What is Feature Selection? Definition and FAQs | HEAVY.AI”. https://www.heavy.ai/technical-glossary/feature-selection (consultado el 4 de noviembre de 2022).
[35] “Training ML Models - Amazon Machine Learning”. https://docs.aws.amazon.com/machine-learning/latest/dg/training-ml-models.html (consultado el 4 de noviembre de 2022).
[36] “Validate a Machine Learning Model - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-model-validation.html (consultado el 4 de noviembre de 2022).
[37] “Safely update models in production - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/model-ab-testing.html (consultado el 4 de noviembre de 2022).
[38] “What Is Amazon SageMaker? - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/whatis.html (consultado el 26 de septiembre de 2022).
[39] “Tutorial: Preparación de datos para machine learning - Amazon Web Services”. https://aws.amazon.com/es/getting-started/hands-on/machine-learning-tutorial-prepare-data-with-minimal-code/ (consultado el 26 de septiembre de 2022).
[40] “¿Qué es AWS CloudFormation? - AWS CloudFormation”. https://docs.aws.amazon.com/es_es/AWSCloudFormation/latest/UserGuide/Welcome.html (consultado el 26 de septiembre de 2022).
[41] J. Ruiz-Rosero, G. Ramirez-Gonzalez, y J. Viveros-Delgado, “Software survey: ScientoPy, a scientometric tool for topics trend analysis in scientific publications”, *Scientometrics*, vol. 121, núm. 2, pp. 1165–1188, nov. 2019, doi: 10.1007/s11192-019-03213-w.
[42] Y. Xia, X. Yang, y Y. Zhang, “A rejection inference technique based on contrastive pessimistic likelihood estimation for P2P lending”, *Electron. Commer. Res. Appl.*, vol. 30, pp. 111–124, jul. 2018, doi: 10.1016/j.elerap.2018.05.011.
[43] S. K. Trivedi, “A study on credit scoring modeling with different feature selection and machine learning approaches”, *Technol. Soc.*, vol. 63, p. 101413, nov. 2020, doi: 10.1016/j.techsoc.2020.101413.
[44] A. Safiya Parvin y B. Saleena, “An Ensemble Classifier Model to Predict Credit Scoring - Comparative Analysis”, en *2020 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)*, dic. 2020, pp. 27–30. doi: 10.1109/iSES50453.2020.00017.
[45] J. P. Barddal, F. Enembreck, L. Loezer, y R. Lanzuolo, “Combining Slow and Fast Learning for Improved Credit Scoring”, en *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, oct. 2020, pp. 1149–1154. doi: 10.1109/SMC42975.2020.9283453.
[46] J. Xiao, X. Zhou, Y. Zhong, L. Xie, X. Gu, y D. Liu, “Cost-sensitive semi-supervised selective ensemble model for customer credit scoring”,

Knowl.-Based Syst., vol. 189, p. 105118, feb. 2020, doi: 10.1016/j.knosys.2019.105118.
[47] F. Xiong, “Credit Loan Modeling: How Different Factors Shape a Client’s Behaviors”, en *Proceedings of the 3rd International Conference on Information Technologies and Electrical Engineering*, New York, NY, USA: Association for Computing Machinery, 2020, pp. 434–438. Consultado: el 9 de septiembre de 2021. [En línea]. Disponible en: https://doi.org/10.1145/3452940.3453023
[48] P. Diaconescu y V.-E. Neagoe, “Credit Scoring Using Deep Learning Driven by Optimization Algorithms”, en *2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, jun. 2020, pp. 1–6. doi: 10.1109/ECAI50035.2020.9223139.
[49] J. P. Barddal, L. Loezer, F. Enembreck, y R. Lanzuolo, “Lessons learned from data stream classification applied to credit scoring”, *Expert Syst. Appl.*, vol. 162, p. 113899, ago. 2020, doi: 10.1016/j.eswa.2020.113899.
[50] M. Malekipirbazari y V. Aksakalli, “Risk assessment in social lending via random forests”, *Expert Syst. Appl.*, vol. 42, núm. 10, pp. 4621–4631, jun. 2015, doi: 10.1016/j.eswa.2015.02.001.
[51] M. Herasymovych, K. Märka, y O. Lukason, “Using reinforcement learning to optimize the acceptance threshold of a credit scoring model”, *Appl. Soft Comput.*, vol. 84, p. 105697, nov. 2019, doi: 10.1016/j.asoc.2019.105697.
[52] “Credit scoring by one-class classification driven dynamical ensemble learning: Journal of the Operational Research Society: Vol 73, No 1”. https://www.tandfonline.com/doi/abs/10.1080/01605682.2021.1944824 (consultado el 25 de octubre de 2022).
[53] “UCREDIT—Credit Scoring Using Social Media Platforms”. https://www.researchgate.net/publication/356688875_UCREDIT-Credit_Scoring_Using_Social_Media_Platforms (consultado el 26 de octubre de 2022).
[54] M. Hossain y C. Mullally, “Using evaluation data to predict loan performance among poor borrowers: The case of BRAC’s asset transfer and microcredit programmes”, *Dev. Policy Rev.*, vol. 40, núm. 3, p. e12579, 2022, doi: 10.1111/dpr.12579.
[55] V. Moscato, A. Picariello, y G. Sperlí, “A benchmark of machine learning approaches for credit score prediction”, *Expert Syst. Appl.*, vol. 165, p. 113986, mar. 2021, doi: 10.1016/j.eswa.2020.113986.
[56] A. Ashofteh y J. M. Bravo, “A conservative approach for online credit scoring”, *Expert Syst. Appl.*, vol. 176, p. 114835, ago. 2021, doi: 10.1016/j.eswa.2021.114835.
[57] G. Li, H.-D. Ma, R.-Y. Liu, M.-D. Shen, y K.-X. Zhang, “A Two-Stage Hybrid Default Discriminant Model Based on Deep Forest”, *Entropy*, vol. 23, núm. 5, Art. núm. 5, may 2021, doi: 10.3390/e23050582.
[58] H. Akaike, “A new look at the statistical model identification”, *IEEE Trans. Autom. Control*, vol. 19, núm. 6, pp. 716–723, dic. 1974, doi: 10.1109/TAC.1974.1100705.
[59] G. Schwarz, “Estimating the Dimension of a Model”, *Ann. Stat.*, vol. 6, núm. 2, pp. 461–464, mar. 1978, doi: 10.1214/aos/1176344136.
[60] “Biogeography based optimization for mining rules to assess credit risk - Giri - 2021 - Intelligent Systems in Accounting, Finance and Management - Wiley Online

Library”. https://onlinelibrary.wiley.com/doi/abs/10.1002/isaf.1486 (consultado el 7 de noviembre de 2022).
[61] D. Swain, P. K. Pattnaik, y T. Athawale, *Machine Learning and Information Processing: Proceedings of ICMLIP 2020*. Springer Nature, 2021.
[62] V. B. Djeundje, J. Crook, R. Calabrese, y M. Hamid, “Enhancing credit scoring with alternative data”, *Expert Syst. Appl.*, vol. 163, p. 113766, ene. 2021, doi: 10.1016/j.eswa.2020.113766.
[63] D. Tripathi, D. R. Edla, A. Bablani, A. K. Shukla, y B. R. Reddy, “Experimental analysis of machine learning methods for credit score classification”, *Prog. Artif. Intell.*, vol. 10, núm. 3, pp. 217–243, sep. 2021, doi: 10.1007/s13748-021-00238-2.
[64] D. Ilter, E. Deniz, y O. Kocadagli, “Hybridized artificial neural network classifiers with a novel feature selection procedure based genetic algorithms and information complexity in credit scoring”, *Appl. Stoch. Models Bus. Ind.*, vol. 37, núm. 2, pp. 203–228, 2021, doi: 10.1002/asmb.2614.
[65] C.-L. Chuang y S.-T. Huang, “A hybrid neural network approach for credit scoring”, *Expert Syst.*, vol. 28, núm. 2, pp. 185–196, 2011, doi: 10.1111/j.1468-0394.2010.00565.x.
[66] “A feature selection enabled hybrid-bagging algorithm for credit risk evaluation - Dahiya - 2017 - Expert Systems - Wiley Online Library”. https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.12217 (consultado el 8 de noviembre de 2022).
[67] D. Tripathi, D. R. Edla, R. Cheruku, y V. Kuppili, “A novel hybrid credit scoring model based on ensemble feature selection and multilayer ensemble classification”, *Comput. Intell.*, vol. 35, núm. 2, pp. 371–394, 2019, doi: 10.1111/coin.12200.
[68] D. Xu, X. Zhang, y H. Feng, “Generalized fuzzy soft sets theory-based novel hybrid ensemble credit scoring model”, *Int. J. Finance Econ.*, vol. 24, núm. 2, pp. 903–921, 2019, doi: 10.1002/ijfe.1698.
[69] S. B. Patel, P. Bhattacharya, S. Tanwar, y N. Kumar, “KiRTi: A Blockchain-Based Credit Recommender System for Financial Institutions”, *IEEE Trans. Netw. Sci. Eng.*, vol. 8, núm. 2, pp. 1044–1054, abr. 2021, doi: 10.1109/TNSE.2020.3005678.
[70] N. T. Luu y P. D. Hung, “Loan Default Prediction Using Artificial Intelligence for the Borrow – Lend Collaboration”, en *Cooperative Design, Visualization, and Engineering: 18th International Conference, CDVE 2021, Virtual Event, October 24–27, 2021, Proceedings*, Berlin, Heidelberg, oct. 2021, pp. 256–270. doi: 10.1007/978-3-030-88207-5_26.
[71] M. Madaan, A. Kumar, C. Keshri, R. Jain, y P. Nagrath, “Loan default prediction using decision trees and random forest: A comparative study”, *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1022, núm. 1, p. 012042, ene. 2021, doi: 10.1088/1757-899X/1022/1/012042.
[72] K. Q. Tran, B. V. Duong, L. Q. Tran, A. L.-H. Tran, A. T. Nguyen, y K. V. Nguyen, “Machine Learning-Based Empirical Investigation for Credit Scoring in Vietnam’s Banking”, en *Advances and Trends in Artificial Intelligence. From Theory to Practice*, Cham, 2021, pp. 564–574. doi: 10.1007/978-3-030-79463-7_48.
[73] A. Altmann, L. Toloşi, O. Sander, y T. Lengauer, “Permutation importance: a corrected feature importance measure”, *Bioinformatics*, vol. 26, núm. 10, pp. 1340–1347, may 2010, doi: 10.1093/bioinformatics/btq134.

[74] S. Barua, D. Gavandi, P. Sangle, L. Shinde, y J. Ramteke, “Swindle: Predicting the Probability of Loan Defaults using CatBoost Algorithm”, en *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, abr. 2021, pp. 1710–1715. doi: 10.1109/ICCMC51019.2021.9418277.
[75] “Use SageMaker-Provided Project Templates - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/sagemaker-projects-templates-sm.html#sagemaker-projects-templates-code-commit (consultado el 25 de junio de 2022).
[76] “How to Build CI/CD Pipelines Using AWS SageMaker”, *Edlitera*. https://www.edlitera.com/blog/posts/aws-sagemaker-ci-cd-pipelines (consultado el 22 de septiembre de 2022).
[77] “Global Infrastructure Regions & AZs”, *Amazon Web Services, Inc.* https://aws.amazon.com/about-aws/global-infrastructure/regions_az/ (consultado el 25 de junio de 2022).
[78] “Amazon Resource Names (ARNs) - AWS General Reference”. https://docs.aws.amazon.com/general/latest/gr/aws-arns-and-namespaces.html (consultado el 25 de junio de 2022).
[79] “Register and Deploy Models with Model Registry - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/model-registry.html (consultado el 25 de junio de 2022).
[80] “Git”. https://git-scm.com/ (consultado el 15 de noviembre de 2022).
[81] “Google Colab”. https://research.google.com/colaboratory/faq.html (consultado el 15 de noviembre de 2022).
[82] “Documentation for Visual Studio Code”. https://code.visualstudio.com/docs (consultado el 15 de noviembre de 2022).
[83] “Loans management from KIVA”. https://kaggle.com/oliversinn/loans-management-from-kiva (consultado el 25 de junio de 2022).
[84] “Datatón Bancolombia 2019”. https://kaggle.com/competitions/datatn-bancolombia-2019 (consultado el 25 de junio de 2022).
[85] “UCI Machine Learning Repository: Statlog (German Credit Data) Data Set”. https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data) (consultado el 25 de junio de 2022).
[86] “UCI Machine Learning Repository: Statlog (Australian Credit Approval) Data Set”. https://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval) (consultado el 25 de junio de 2022).
[87] “www.ppgia.pucpr.br”. https://www.ppgia.pucpr.br/~jean.barddal/datasets/CSDS.zip (consultado el 25 de junio de 2022).
[88] M.-J. Kim y I. Han, “The discovery of experts’ decision rules from qualitative bankruptcy data using genetic algorithms”, *Expert Syst. Appl.*, vol. 25, núm. 4, pp. 637–646, nov. 2003, doi: 10.1016/S0957-4174(03)00102-7.
[89] “UCI Machine Learning Repository: Qualitative_Bankruptcy Data Set”. https://archive.ics.uci.edu/ml/datasets/qualitative_bankruptcy (consultado el 13 de noviembre de 2022).

[90] “UCI Machine Learning Repository: Bank Marketing Data Set”. https://archive.ics.uci.edu/ml/datasets/bank+marketing (consultado el 13 de noviembre de 2022).
[91] “UCI Machine Learning Repository: default of credit card clients Data Set”. https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients (consultado el 13 de noviembre de 2022).
[92] “UCI Machine Learning Repository: Japanese Credit Screening Data Set”. https://archive.ics.uci.edu/ml/datasets/Japanese+Credit+Screening (consultado el 13 de noviembre de 2022).
[93] “All Lending Club loan data”. https://www.kaggle.com/wordsforthewise/lending-club (consultado el 25 de junio de 2022).
[94] “🏦 Lending Club Loan 💰 Defaulters 🏃 Prediction”. https://kaggle.com/faressayah/lending-club-loan-defaulters-prediction (consultado el 25 de junio de 2022).
[95] “Rate information - Lending Club”. https://www.lendingclub.com/foliofn/rateDetail.action (consultado el 25 de junio de 2022).
[96] Swapnilbobe, “Feature Selection in Machine Learning”, *Analytics Vidhya*, el 12 de marzo de 2021. https://medium.com/analytics-vidhya/feature-selection-in-machine-learning-ec1f5d053007 (consultado el 25 de junio de 2022).
[97] “imbalanced-learn documentation — Version 0.10.1”. https://imbalanced-learn.org/stable/index.html (consultado el 27 de febrero de 2023).
[98] “RandomUnderSampler — Version 0.10.1”. https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html (consultado el 27 de febrero de 2023).
[99] “RandomOverSampler — Version 0.10.1”. https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html (consultado el 27 de febrero de 2023).
[100] “AllKNN — Version 0.10.1”. https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.AllKNN.html (consultado el 27 de febrero de 2023).
[101] “Using Docker containers with SageMaker - Amazon SageMaker”. https://docs.aws.amazon.com/sagemaker/latest/dg/docker-containers.html (consultado el 9 de noviembre de 2022).
[102] “sklearn.model_selection.StratifiedShuffleSplit — scikit-learn 1.1.3 documentation”. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html (consultado el 9 de noviembre de 2022).
[103] “sklearn.model_selection.StratifiedShuffleSplit”, *scikit-learn*. https://scikit-learn/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html (consultado el 25 de junio de 2022).
[104] “sklearn.model_selection.cross_val_predict”, *scikit-learn*. https://scikit-learn/stable/modules/generated/sklearn.model_selection.cross_val_pre

dict.html (consultado el 9 de noviembre de 2022).
[105] E. Cerón, “Applying MLOPS practices with SageMaker — Part 3 —The inference endpoint”, *Medium*, el 22 de septiembre de 2022. https://medium.com/@restebance/applying-mlops-practices-with-sagemaker-part-3-the-inference-endpoint-cee433db05eb (consultado el 13 de noviembre de 2022).
[106] D. Vasquez, “Applying MLOps practices with SageMaker — Part 2: Pipeline”, *Medium*, el 22 de septiembre de 2022. https://medium.com/@daniel.vasquez.97/applying-mlops-practices-with-sagemaker-part-2-pipeline-af1549a7ca8 (consultado el 10 de noviembre de 2022).

PROPUESTA DE IMPLEMENTACIÓN DE UN PILOTO DE MLOPS SIGUIENDO LOS LINEAMIENTOS DEFINIDOS POR LA EMPRESA WIZIT MIND BLOWING SOLUTIONS S.A.S.

ANEXOS

DANIEL SANTIAGO VÁSQUEZ ASTAIZA

Director: PhD. GUSTAVO ADOLFO RAMÍREZ GONZÁLEZ
Asesor: Ing. RAFAEL ESTEBAN CERÓN ESPINOSA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE TELEMÁTICA
SERVICIOS AVANZADOS SOBRE INTERNET
POPAYÁN, 2022

Anexo A: Acceder a las plantillas de AWS para *MLOps*

Requisitos:

- Una cuenta de AWS
- Un usuario en *Amazon SageMaker*
- Una aplicación asociada al usuario de *Amazon SageMaker*

Pasos:

1. Dirigirse al panel de control de *Amazon SageMaker* y lanzar la aplicación asociada en *Studio*.

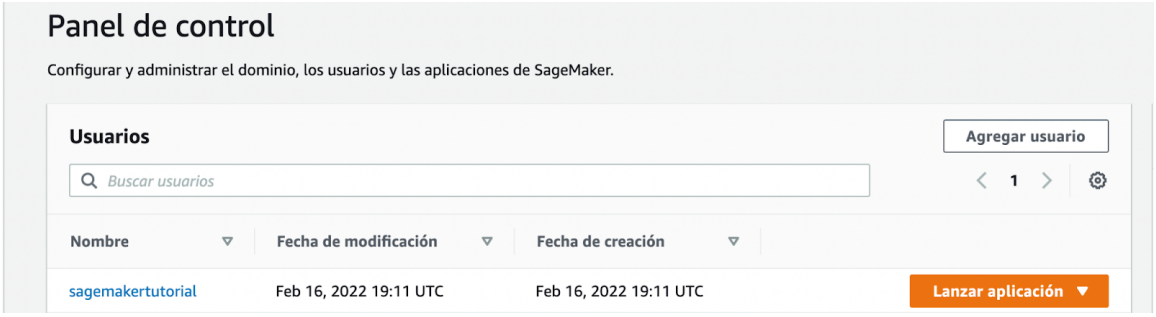


Figura 31: Panel de control de Amazon SageMaker. Fuente propia.

2. Una vez se ejecute la aplicación se mostrará una pantalla inicial con un conjunto de opciones. Se selecciona la opción "*New project*" ubicada en la parte inferior izquierda.

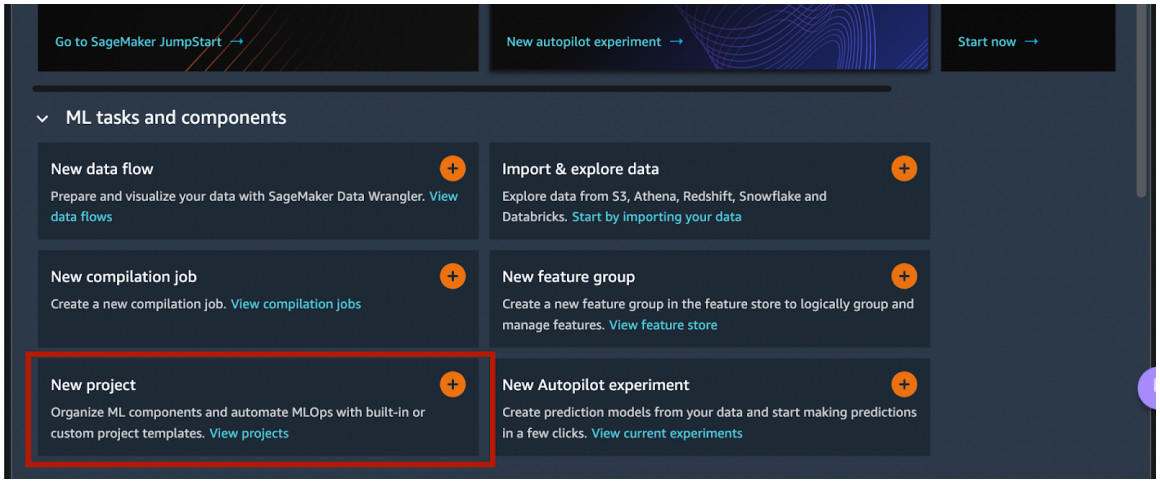


Figura 32: Panel de control de Amazon SageMaker con Jupyter Lab. Fuente propia.

3. Se selecciona una plantilla según las características requeridas para el proyecto.

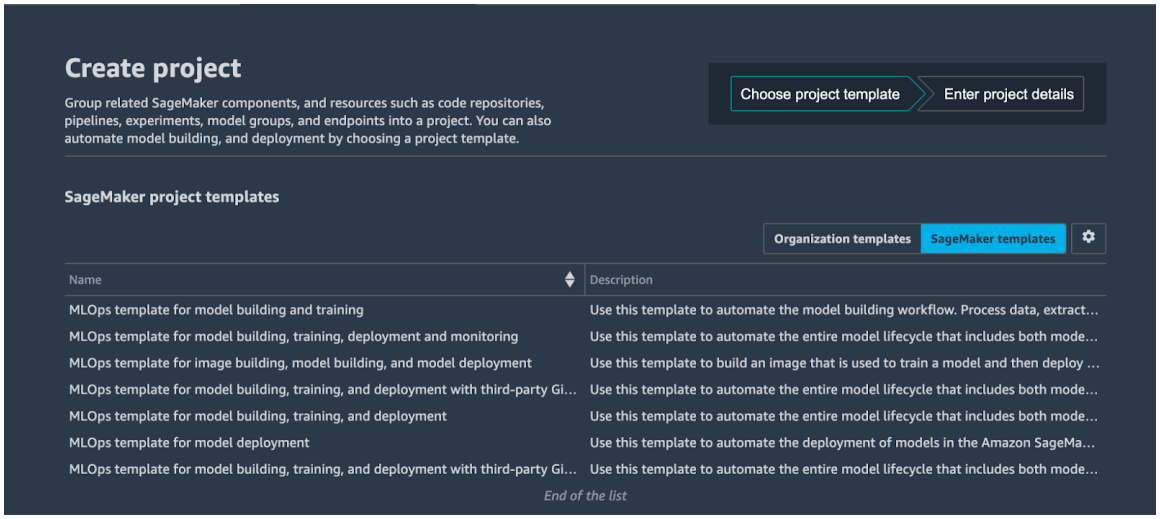


Figura 33: Panel de control de Amazon SageMaker para la creación de la plantilla de MLOps. Fuente propia.

Una vez seleccionada una plantilla y diligenciada su información se aprovisionan automáticamente los recursos y servicios web requeridos.

Anexo B: Refactorización de la plantilla de AWS del *pipeline* de *MLOps*

Puesto que ya se han obtenido los resultados durante el entrenamiento y la validación, y se completó el proceso de selección del modelo, es importante integrar la implementación realizada a la plantilla del *pipeline* de AWS para *MLOps*.

El proceso a continuación está tomado de [106] y se hacen las indicaciones respectivas para lograr realizar la refactorización de una plantilla nueva.

Requisitos:

- Plantilla de AWS para *MLOps* con sus recursos previamente provisionados.
- *Notebooks* de *Jupyter* o código en Python para preprocesamiento y entrenamiento.
- Entorno de desarrollo (Por ejemplo: *Visual Studio Code)*.

Descripción de la plantilla:

Al analizar la estructura del proyecto se pueden observar los siguientes archivos:

```
| — codebuild-buildspec.yml
| — CONTRIBUTING.md
| — pipelines
| | — abalone
| | | — evaluate.py
| | | — pipeline.py
| | | — preprocess.py
| | ` — __init__.py
| | — get_pipeline_definition.py
| | — __init__.py
| | — run_pipeline.py
| | — _utils.py
| ` — __version__.py
| — README.md
| — sagemaker-pipelines-project.ipynb
| — setup.cfg
| — setup.py
| — tests
| ` — test_pipelines.py
` — tox.ini
```

Figura 34: Estructura de archivos de la plantilla MLOps template for model building, training, and deployment. Fuente propia.

Para este caso en particular, no es necesario explorar todos los ficheros existentes en el proyecto, basta con conocer los que se encuentran resaltados en amarillo en la figura 34.

Obtención de los datos desde Amazon S3:

```python
if __name__ == "__main__":
    logger.debug("Starting preprocessing.")
    parser = argparse.ArgumentParser()
    parser.add_argument("--input-data", type=str, required=True)
    args = parser.parse_args()

    base_dir = "/opt/ml/processing"
    pathlib.Path(f"{base_dir}/data").mkdir(parents=True, exist_ok=True)
    input_data = args.input_data
    bucket = input_data.split("/")[2]
    key = "/".join(input_data.split("/")[3:])

    logger.info("Downloading data from bucket: %s, key: %s", bucket, key)
    fn = f"{base_dir}/data/abalone-dataset.csv"
    s3 = boto3.resource("s3")
    s3.Bucket(bucket).download_file(key, fn)

    logger.debug("Reading downloaded data.")
    df = pd.read_csv(
        fn,
        header=None,
        names=feature_columns_names + [label_column],
        dtype=merge_two_dicts(feature_columns_dtype, label_column_dtype),
    )
    os.unlink(fn)
```

Figura 35: Archivo preprocess.py, señalando la línea donde se especifica el dataset. Fuente propia.

Para cargar el *dataset* apropiado, se debe especificar en la línea señalada en la figura 35 la ruta del archivo csv en AWS S3.

Preprocesamiento:

```
logger.debug("Defining transformers.")
numeric_features = list(feature_columns_names)
numeric_features.remove("sex")
numeric_transformer = Pipeline(
    steps=[("imputer", SimpleImputer(strategy="median")), ("scaler", StandardScaler())]
)

categorical_features = ["sex"]
categorical_transformer = Pipeline(
    steps=[
        ("imputer", SimpleImputer(strategy="constant", fill_value="missing")),
        ("onehot", OneHotEncoder(handle_unknown="ignore")),
    ]
)

preprocess = ColumnTransformer(
    transformers=[
        ("num", numeric_transformer, numeric_features),
        ("cat", categorical_transformer, categorical_features),
    ]
)

logger.info("Applying transforms.")
y = df.pop("rings")
X_pre = preprocess.fit_transform(df)
y_pre = y.to_numpy().reshape(len(y), 1)
```

Figura 36: Archivo preprocess.py, señalando la línea donde se especifica el área donde se hace el preprocesamiento de los datos. Fuente propia.

Todas las transformaciones necesarias para el funcionamiento del modelo, comprendiendo las transformaciones sobre variables numéricas y categóricas, además de procesos de ingeniería de características se deben realizar sobre el área marcada.

División del *dataset*:

```
X = np.concatenate((y_pre, X_pre), axis=1)

logger.info("Splitting %d rows of data into train, validation, test datasets.", len(X))
np.random.shuffle(X)
train, validation, test = np.split(X, [int(0.7 * len(X)), int(0.85 * len(X))])

logger.info("Writing out datasets to %s.", base_dir)
pd.DataFrame(train).to_csv(f"{base_dir}/train/train.csv", header=False, index=False)
pd.DataFrame(validation).to_csv(
    f"{base_dir}/validation/validation.csv", header=False, index=False
)
pd.DataFrame(test).to_csv(f"{base_dir}/test/test.csv", header=False, index=False)
```

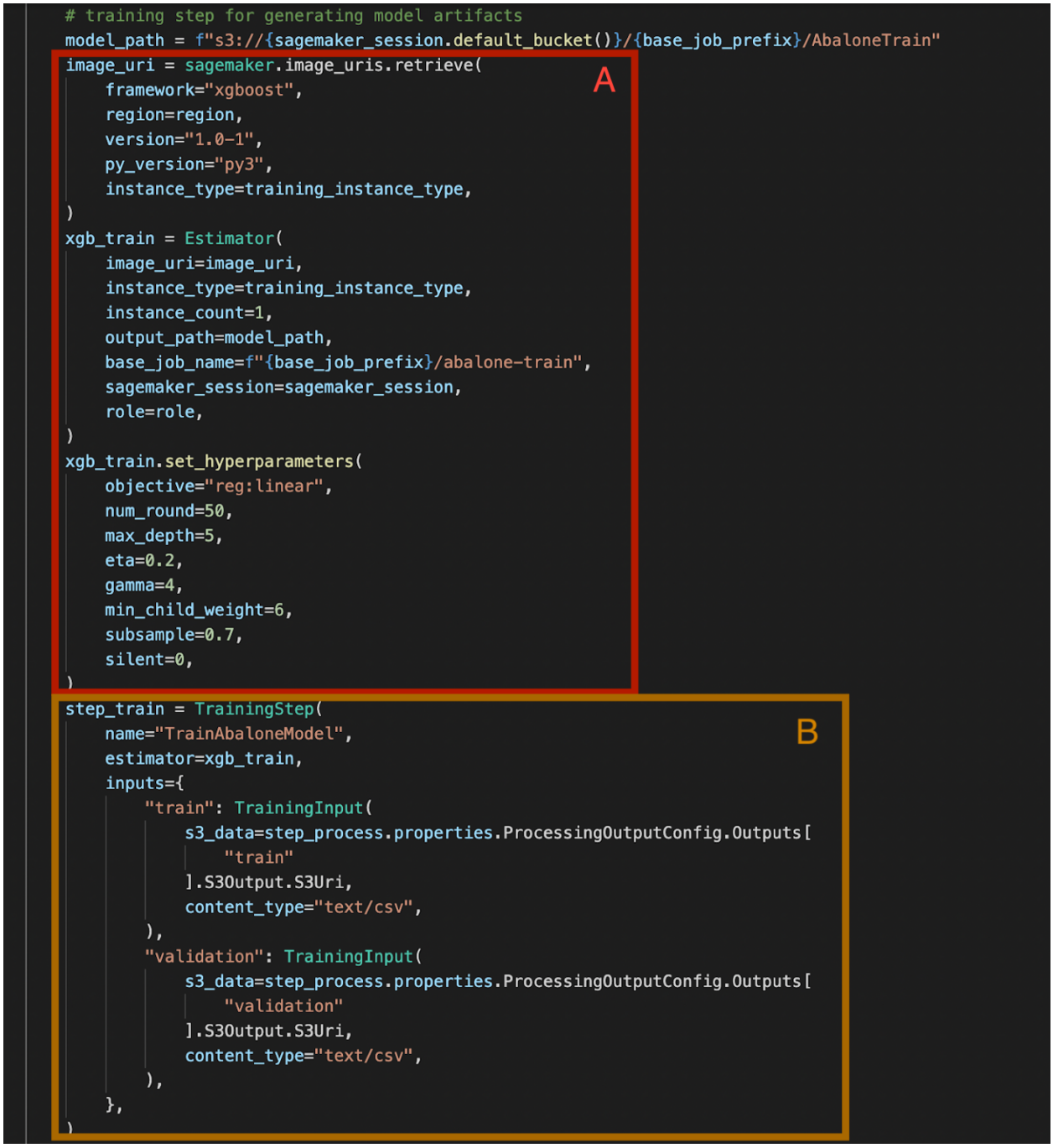
Figura 37: Archivo preprocess.py, señalando la línea donde se especifica el área donde se hace el la división del dataset. Fuente propia.

Para realizar el entrenamiento, la validación y las pruebas; se modifica a conveniencia la implementación mostrada en la figura 37, en caso de requerir un proceso de división distinto al aleatorio, para el caso de este proyecto se hace de forma estratificada.

Nota: *Es importante crear una división de pruebas del dataset puesto que no se cuenta con datos reales.*

Entrenamiento:

Para realizar el entrenamiento del modelo, se debe precargar la imagen de este. La imagen por defecto para la plantilla AWS es la de *XGBoost*, con lo cual en la figura 38 se muestra como esta es cargada y asignada para ser entrenada y probada:




```
# training step for generating model artifacts
model_path = f"s3://{sagemaker_session.default_bucket()}/{base_job_prefix}/AbaloneTrain"
image_uri = sagemaker.image_uris.retrieve(
    framework="xgboost",
    region=region,
    version="1.0-1",
    py_version="py3",
    instance_type=training_instance_type,
)
xgb_train = Estimator(
    image_uri=image_uri,
    instance_type=training_instance_type,
    instance_count=1,
    output_path=model_path,
    base_job_name=f"{base_job_prefix}/abalone-train",
    sagemaker_session=sagemaker_session,
    role=role,
)
xgb_train.set_hyperparameters(
    objective="reg:linear",
    num_round=50,
    max_depth=5,
    eta=0.2,
    gamma=4,
    min_child_weight=6,
    subsample=0.7,
    silent=0,
)
step_train = TrainingStep(
    name="TrainAbaloneModel",
    estimator=xgb_train,
    inputs={
        "train": TrainingInput(
            s3_data=step_process.properties.ProcessingOutputConfig.Outputs[
                "train"
            ].S3Output.S3Uri,
            content_type="text/csv",
        ),
        "validation": TrainingInput(
            s3_data=step_process.properties.ProcessingOutputConfig.Outputs[
                "validation"
            ].S3Output.S3Uri,
            content_type="text/csv",
        ),
    },
)
```

Figura 38: Archivo pipeline.py, señalando las áreas A y B. Fuente propia.

Siguiendo la figura 38, lo contenido en el área A permite realizar la carga del modelo para ser entrenado a partir de una imagen que luego es instanciada a través de la clase *Estimator* y finalmente los híper parámetros son configurados. Por otro lado, el área B permite obtener los datos de entrenamiento y validación, referenciando directamente el proceso anterior para nutrir el modelo cargado en A.

Evaluación del modelo y pruebas:

Este proceso se realiza teniendo como referencia los datos de prueba, ya que no se cuenta con datos reales. A continuación se muestra como se lleva a cabo el proceso de evaluación:

```python
# processing step for evaluation
script_eval = ScriptProcessor(
    image_uri=image_uri,
    command=["python3"],
    instance_type=processing_instance_type,
    instance_count=1,
    base_job_name=f"{base_job_prefix}/script-abalone-eval",
    sagemaker_session=sagemaker_session,
    role=role,
)
evaluation_report = PropertyFile(
    name="AbaloneEvaluationReport",
    output_name="evaluation",
    path="evaluation.json",
)
step_eval = ProcessingStep(
    name="EvaluateAbaloneModel",
    processor=script_eval,
    inputs=[
        ProcessingInput(
            source=step_train.properties.ModelArtifacts.S3ModelArtifacts,
            destination="/opt/ml/processing/model",
        ),
        ProcessingInput(
            source=step_process.properties.ProcessingOutputConfig.Outputs[
                "test"
            ].S3Output.S3Uri,
            destination="/opt/ml/processing/test",
        ),
    ],
    outputs=[
        ProcessingOutput(output_name="evaluation", source="/opt/ml/processing/evaluation"),
    ],
    code=os.path.join(BASE_DIR, "evaluate.py"),
    property_files=[evaluation_report],
)
```

A

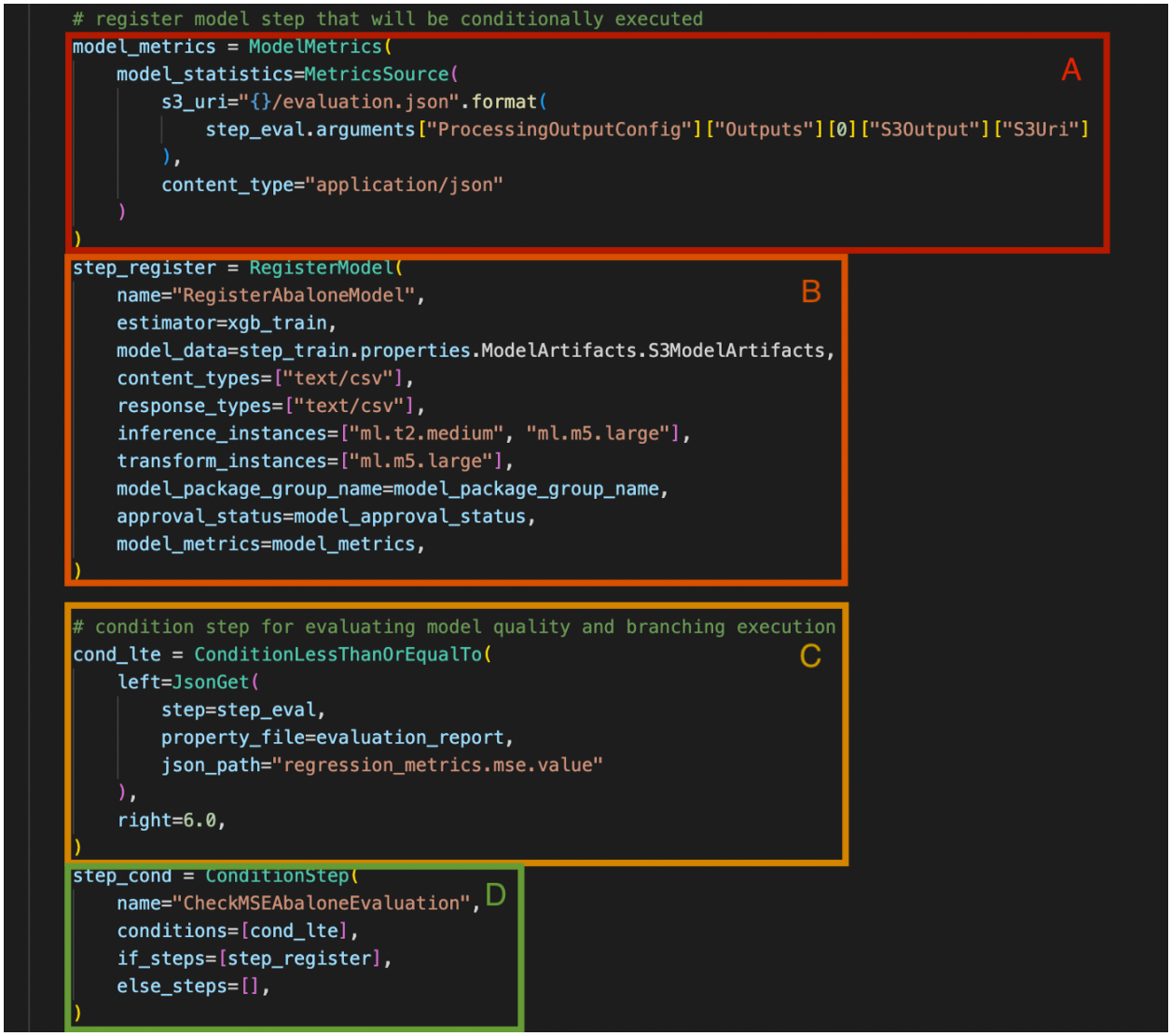
Figura 39: Archivo pipeline.py, señalando el área A. Fuente propia.

En la figura 39 se denota el área A, la cual se encarga de utilizar el modelo entrenado y validado para determinar su desempeño. A no ser que se busque realizar una evaluación customizada, no se recomienda modificar esta pieza de código.

Registro del modelo y pasos condicionales:

Teniendo en cuenta que se está alcanzando la etapa final del *pipeline*, se deben definir las acciones a tomar dependiendo del desempeño del modelo. A continuación, se

detallan las secciones que deben ser modificadas para evaluar el modelo basado en los criterios requeridos por el proyecto:




```
# register model step that will be conditionally executed
model_metrics = ModelMetrics(
    model_statistics=MetricsSource(
        s3_uri="{}/evaluation.json".format(
            step_eval.arguments["ProcessingOutputConfig"]["Outputs"][0]["S3Output"]["S3Uri"]
        ),
        content_type="application/json"
    )
)
step_register = RegisterModel(
    name="RegisterAbaloneModel",
    estimator=xgb_train,
    model_data=step_train.properties.ModelArtifacts.S3ModelArtifacts,
    content_types=["text/csv"],
    response_types=["text/csv"],
    inference_instances=["ml.t2.medium", "ml.m5.large"],
    transform_instances=["ml.m5.large"],
    model_package_group_name=model_package_group_name,
    approval_status=model_approval_status,
    model_metrics=model_metrics,
)
# condition step for evaluating model quality and branching execution
cond_lte = ConditionLessThanOrEqualTo(
    left=JsonGet(
        step=step_eval,
        property_file=evaluation_report,
        json_path="regression_metrics.mse.value"
    ),
    right=6.0,
)
step_cond = ConditionStep(
    name="CheckMSEAbaloneEvaluation",
    conditions=[cond_lte],
    if_steps=[step_register],
    else_steps=[],
)
```

Figura 40: Archivo pipeline.py, señalando las áreas A, B, C y D.

De acuerdo con la figura 40, en el área A se obtienen las métricas que se van a evaluar sobre el modelo. En el área B, se define el paso de registro del modelo entrenado y las instancias o tipos de máquina que lo van a correr. En el área C, se define un umbral o condición que permite registrar al modelo en caso de exhibir el desempeño deseado. Finalmente, en el área D se incluyen la condición planteada anteriormente y el paso que permite registrar en caso de ser aceptable para el umbral.

Definición del *pipeline* final:

Por último, se define el *pipeline* que va a ejecutar todos los pasos anteriores y va a desplegar el modelo. En la siguiente figura se detalla su funcionamiento a nivel de código:

```
    # pipeline instance
    pipeline = Pipeline(
        name=pipeline_name,
        parameters=[
            processing_instance_type,
            processing_instance_count,
            training_instance_type,
            model_approval_status,
            input_data,
        ],
        steps=[step_process, step_train, step_eval, step_cond],
        sagemaker_session=sagemaker_session,
    )
    return pipeline
```

Figura 41: Archivo pipeline.py, definición del pipeline final. Fuente propia.

De acuerdo con la figura 41, se definen los parámetros de entrada del *pipeline* y los pasos o *steps* que lo componen.

Anexo C: Despliegue del *pipeline* en AWS

Se busca conocer el desempeño de la implementación realizada en el entorno de AWS con los ajustes que demanda el proyecto.

El proceso a continuación está tomado de [105] y se hacen las indicaciones respectivas para llevar a cabo el proceso de despliegue del *pipeline* implementado en AWS.

Requisitos:

- Plantilla editada de AWS para *MLOps* con sus recursos previamente provisionados.
- Entorno de desarrollo o terminal integrada.

Publicar los cambios en *CodeCommit*:

Teniendo en cuenta que el despliegue del *pipeline* se realiza cuando un evento de *commit* se emite desde *CodeCommit*, es importante publicar los cambios realizados a la rama *main* del repositorio *modelbuild*, como se indica a continuación:

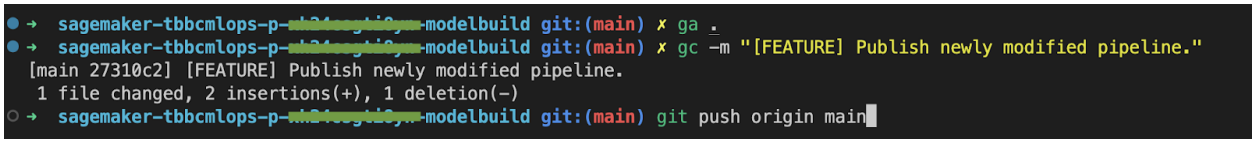


Figura 42: Terminal desde donde se publican los cambios del pipeline. Fuente propia.

Una vez se inicia el proceso de canalización para la generación del *pipeline* de *MLOps*, es posible visualizar los pasos del proceso y el estado de los mismos a través de *CodePipeline*, cabe aclarar que en esta misma consola se pueden ver los registros o *logs* detallados del funcionamiento de la compilación del modelo. A continuación se observa una demostración:

Nombre	Estado	Contexto	Duración	Hora de inicio	Hora de finalización
SUBMITTED	Realizado correctamente	-	<1 sec	En. 30, 2023 12:06 p. m. (UTC-5:00)	En. 30, 2023 12:06 p. m. (UTC-5:00)
QUEUED	Realizado correctamente	-	3 secs	En. 30, 2023 12:06 p. m. (UTC-5:00)	En. 30, 2023 12:06 p. m. (UTC-5:00)
PROVISIONING	Realizado correctamente	-	94 secs	En. 30, 2023 12:06 p. m. (UTC-5:00)	En. 30, 2023 12:07 p. m. (UTC-5:00)
DOWNLOAD_SOURCE	Realizado correctamente	-	4 secs	En. 30, 2023 12:07 p. m. (UTC-5:00)	En. 30, 2023 12:07 p. m. (UTC-5:00)
INSTALL	Realizado correctamente	-	55 secs	En. 30, 2023 12:07 p. m. (UTC-5:00)	En. 30, 2023 12:08 p. m. (UTC-5:00)
PRE_BUILD	Realizado correctamente	-	<1 sec	En. 30, 2023 12:08 p. m. (UTC-5:00)	En. 30, 2023 12:08 p. m. (UTC-5:00)
BUILD	Realizado correctamente	-	878 secs	En. 30, 2023 12:08 p. m. (UTC-5:00)	En. 30, 2023 12:23 p. m. (UTC-5:00)
POST_BUILD	Realizado correctamente	-	<1 sec	En. 30, 2023 12:23 p. m. (UTC-5:00)	En. 30, 2023 12:23 p. m. (UTC-5:00)
UPLOAD_ARTIFACTS	Realizado correctamente	-	<1 sec	En. 30, 2023 12:23 p. m. (UTC-5:00)	En. 30, 2023 12:23 p. m. (UTC-5:00)
FINALIZING	Realizado correctamente	-	2 secs	En. 30, 2023 12:23 p. m. (UTC-5:00)	En. 30, 2023 12:23 p. m. (UTC-5:00)
COMPLETED	Realizado correctamente	-	-	En. 30, 2023 12:23 p. m. (UTC-5:00)	-

Figura 43: Pasos del proceso de compilación del modelo. Fuente propia.

Verificación del panel de control de *SageMaker*:

Una vez ejecutada la publicación de los cambios realizados, es necesario consultar el panel de control de *SageMaker* para conocer el estado de los trabajos que se están llevando a cabo, principalmente *Training* o Entrenamiento y *Processing* o Procesamiento, como se ilustra en la siguiente figura:

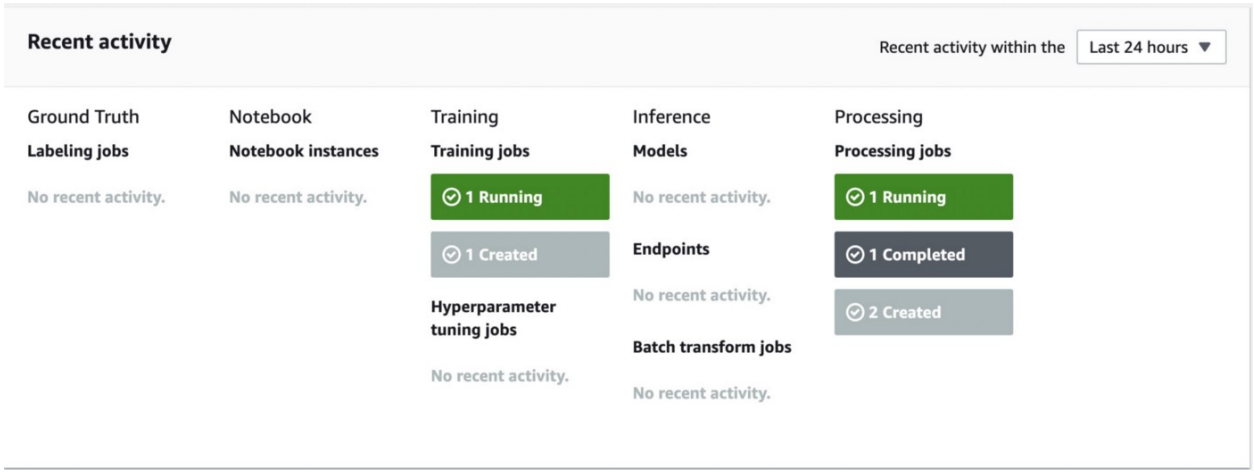


Figura 44: Trabajos ejecutados en el panel de control de SageMaker. Tomado de [105].

Una vez se haya completado el registro del modelo, el punto de inferencia podrá ser desplegado, seguido a ello, el *pipeline* está listo para desplegar el modelo en el entorno *Staging* y *Production*, sin embargo, el paso a este último entorno es opcional en este punto. A continuación se muestra el proceso de despliegue del *pipeline* desde la consola de AWS:

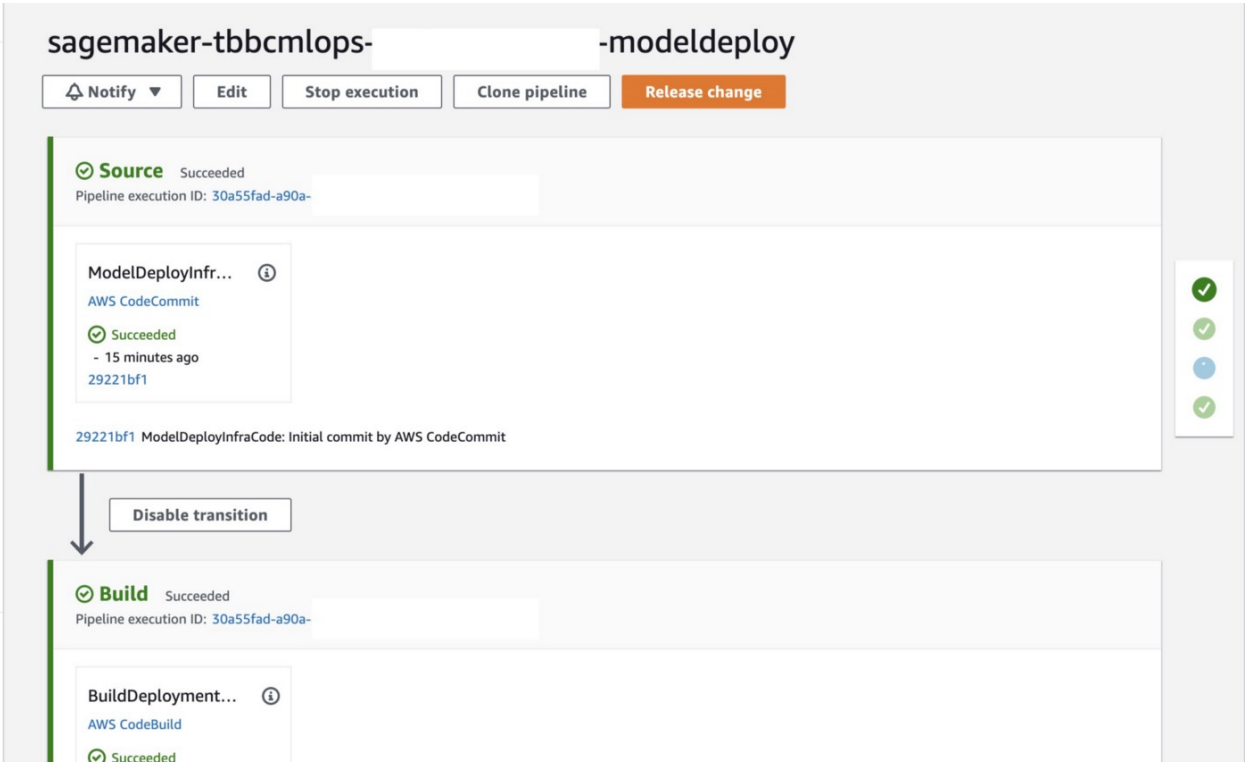


Figura 45: Pasos del despliegue del pipeline en SageMaker. Tomado de [105].

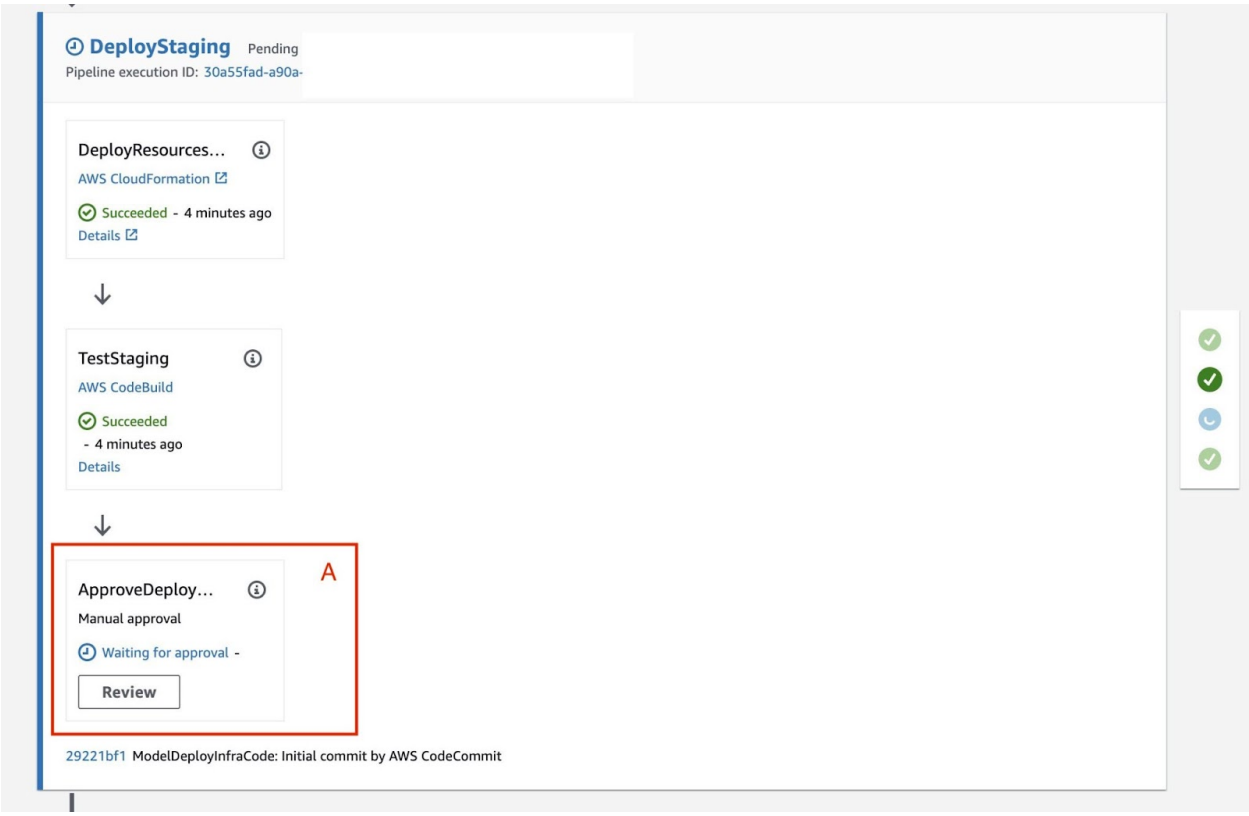


Figura 46: Pasos del despliegue del pipeline en entorno Staging en SageMaker. Tomado de [105].

El área marcada como A dentro de la figura 41, señala un paso que debe ser llevado a cabo de manera manual para el despliegue completo del pipeline a *Staging*. Dicho paso tiene la función de servir como medida de seguridad adicional para prevenir despliegues no autorizados por el usuario. Una vez autorizado el despliegue, el punto de inferencia estará disponible para ser consultado via solicitudes HTTP, como se ilustra en la siguiente figura:

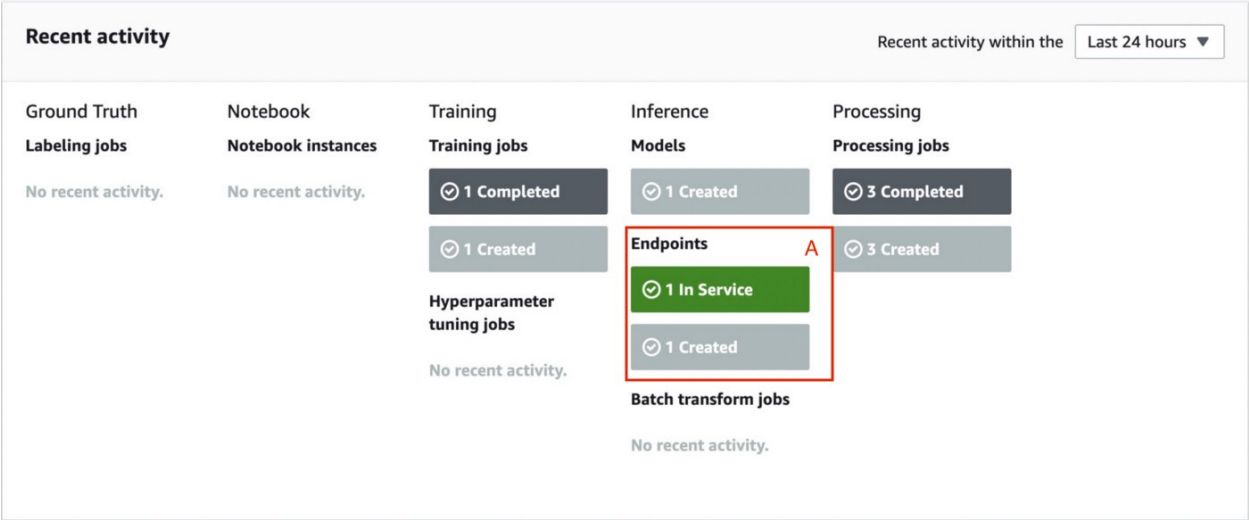


Figura 47: Panel de control de SageMaker con el punto de inferencia en servicio. Tomado de [105].

Ahora el proceso de despliegue a entorno *Staging* se encuentra completo y el *pipeline* implementado ya es funcional para ser consumido desde un aplicativo.

Anexo D: Aplicación implementada

Para realizar el consumo del servicio web expuesto mediante el *endpoint* de inferencia, se realizó la implementación de una aplicación web, que recibe un conjunto de entradas de datos de un usuario a través de un formulario con el fin de determinar su perfil de riesgo.

A continuación, se ilustra el proceso para hacer uso de dicha aplicación, a través de capturas de pantalla:

Autenticación:

Es necesario autenticarse para hacer uso de la aplicación, en este caso se cuenta con dos mecanismos: Inicio de sesión y registro de usuario. A continuación, se muestran capturas de pantalla de los dos métodos en los cuales un usuario puede ingresar a la plataforma:

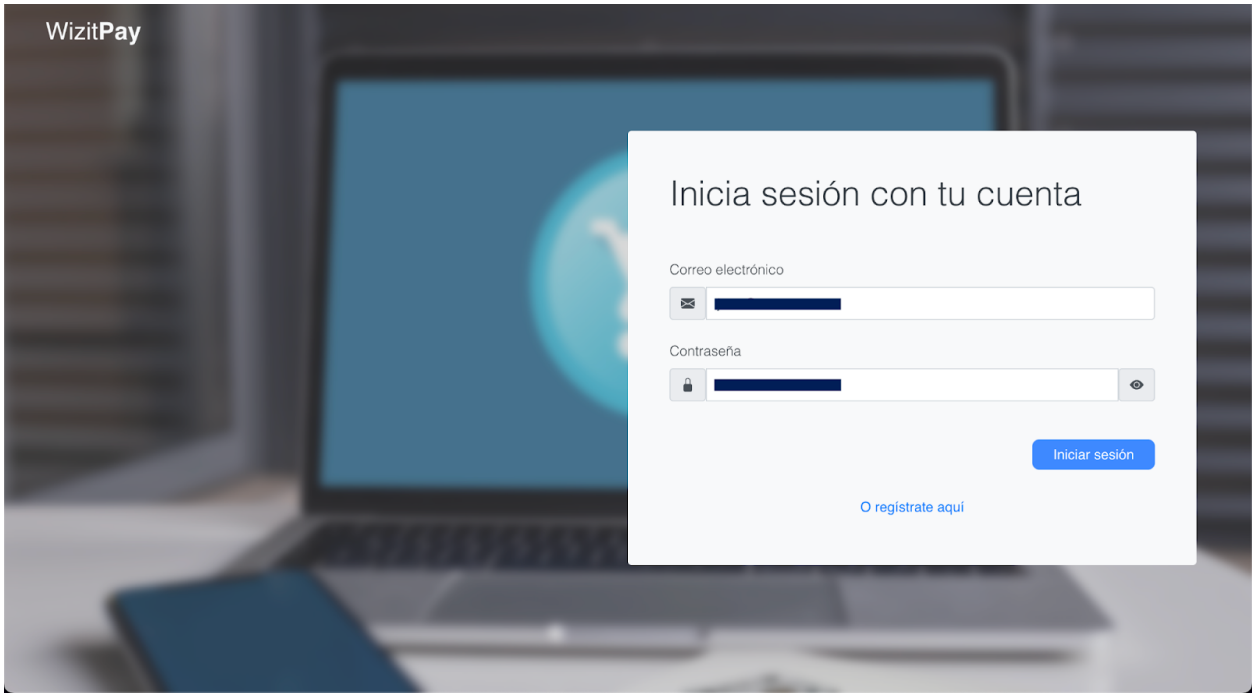


Figura 48: Inicio de sesión en WizitPay. Fuente propia.

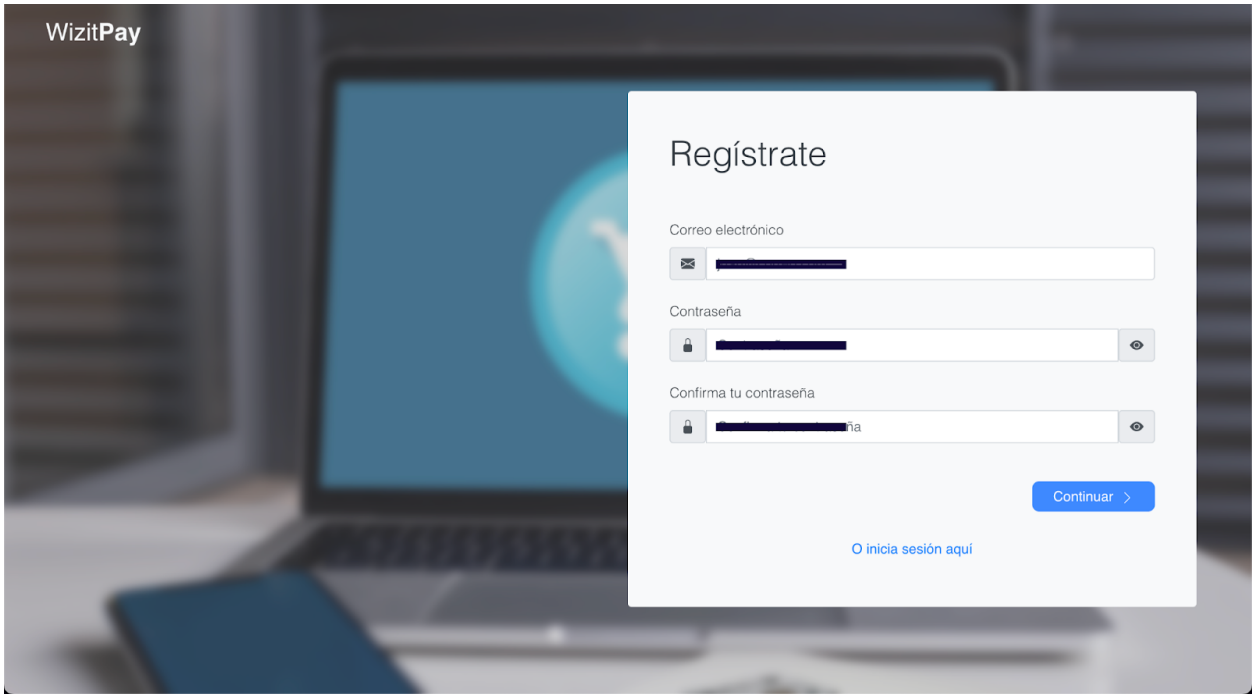


Figura 49: Registro de usuario en WizitPay. Fuente propia.

Ingreso de datos de formulario:

Una vez autenticado, el usuario es redirigido al formulario mediante el cual se establece su perfil de riesgo, el cual cuenta con los siguientes campos: Cantidad del préstamo, ganancias mensuales, gastos mensuales, valor de sus deudas, tipo de propiedad de su vivienda y el propósito de su solicitud. A continuación, se observa la disposición de dichos atributos en la implementación:

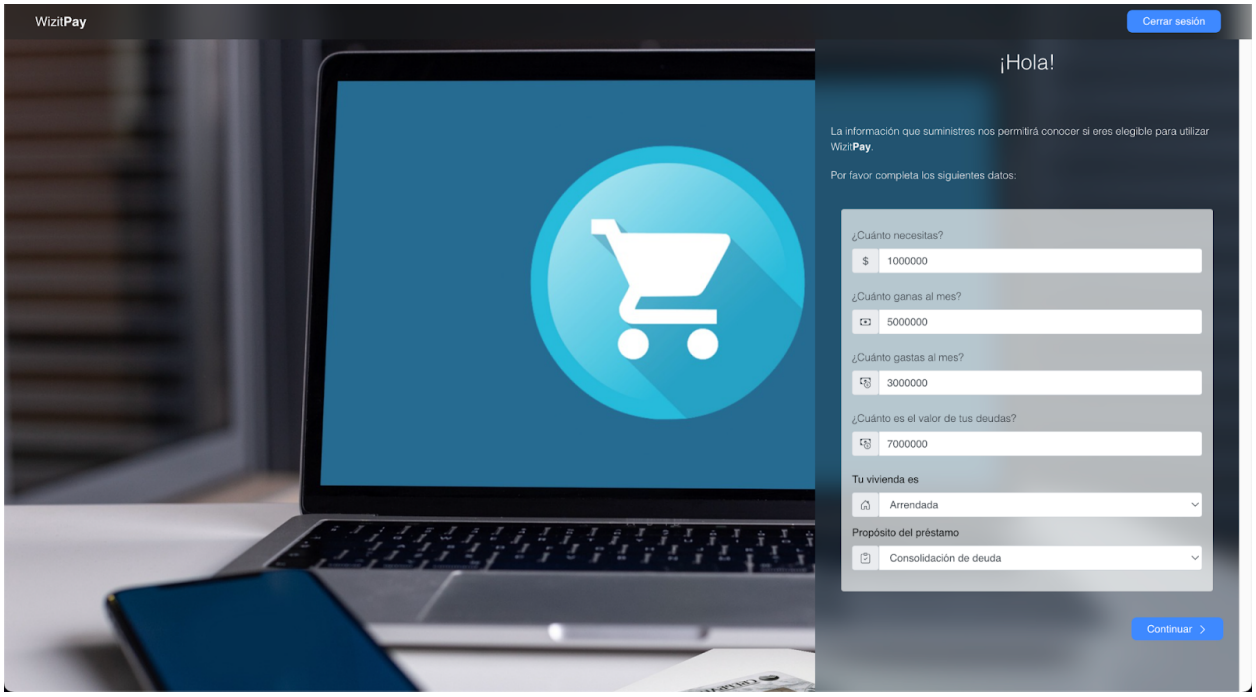


Figura 50: Formulario de evaluación del perfil de riesgo. Fuente propia.

Obtención de resultados:

Una vez fueron procesados los datos ingresados por el usuario y estos han sido enviados al *endpoint* de inferencia, a través de un umbral en el cual, únicamente las solicitudes con resultados mayores o iguales a ***0.75*** serán aprobadas. A continuación se muestran los dos resultados posibles para este proceso:

Figura 51: Préstamo aprobado. Fuente propia.

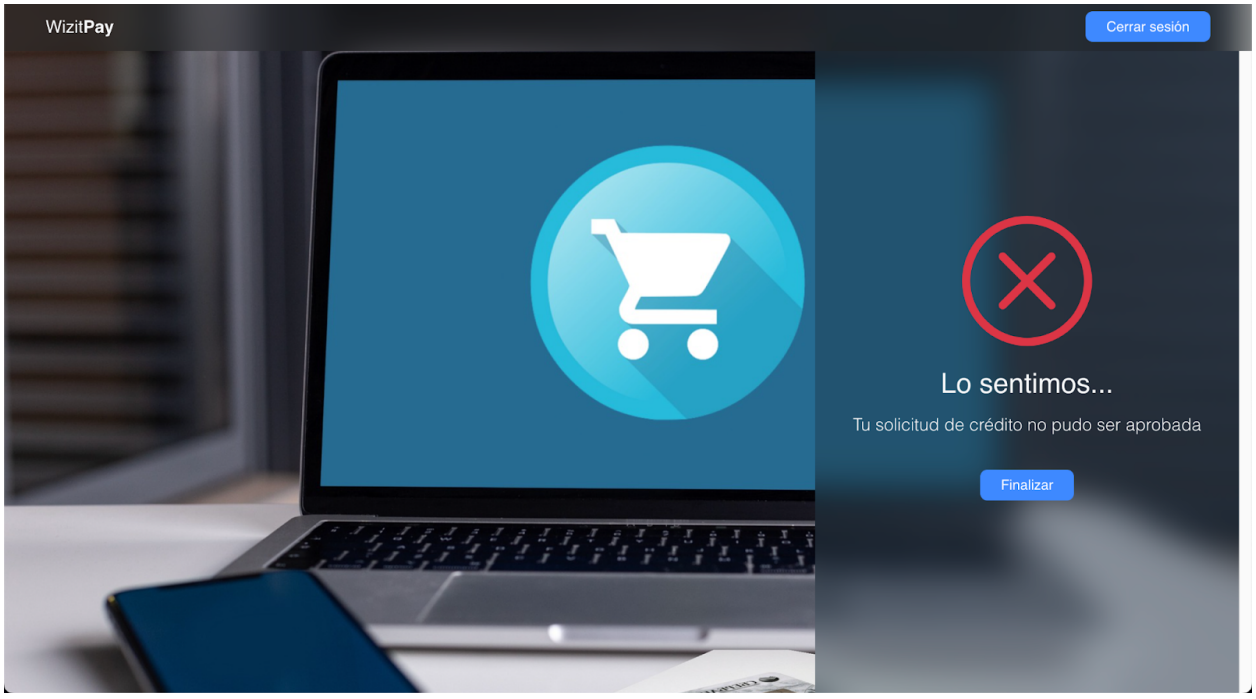


Figura 52: Préstamo rechazado. Fuente propia.