UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

SOLUCIÓN BASADA EN MACHINE LEARNING PARA PREDECIR COMPORTAMIENTO DE PAGO DE CUENTAS POR COBRAR EN UNA EMPRESA FINANCIERA

POR:

JAVIER RODRÍGUEZ LÓPEZ

TESIS PRESENTADA COMO REQUISITO PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERÍA EN COMPUTACIÓN

CHIHUAHUA, CHIH., MÉXICO **ENERO DE 2024**

Solución basada en machine learning para predecir comportamiento de pago de cuentas por cobrar en una empresa financiera. Tesis, presentada por Javier Rodríguez López como requisito parcial para obtener el grado de Maestría en Ingeniería en Computación, ha sido aprobado y aceptado por:

M.I. Fabián Vinicio Hernández Martínez
Director de la Facultad de Ingeniería

Dr. Fernando Martínez Reyes
Secretario de Investigación y Posgrado

M.S.I. Karina Rocío Requena Yáñez
Coordinadora Académica

Dr. Luis Carlos González Gurrola
Director de Tesis

Diciembre 2023

COMITÉ
Dr. Luis Carlos González Gurrola
Dr. Manuel Montes y Gómez
Dr. Jesús Roberto López Santillán
Dr. Fernando Martínez Reyes

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

11 de diciembre de 2023.

ING. JAVIER RODRÍGUEZ LÓPEZ
Presente. -

En atención a su solicitud relativa al trabajo de tesis para obtener el grado de Maestro en Ingeniería en Computación, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. Luis Carlos González Gurrola** para que lo desarrolle como Tesis, con el título **"SOLUCIÓN BASADA EN MACHINE LEARNING PARA PREDECIR COMPORTAMIENTO DE PAGO DE CUENTAS POR COBRAR EN UNA EMPRESA FINANCIERA"**.

Índice de Contenido

FACULTAD DE INGENIERÍA
Circuito No. 1, Campus Universitario 2
Chihuahua, Chih., México. C.P. 31125
Tel. (614) 442-95-00
www.fing.uach.mx

+uach

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

Capítulo 2: Marco teórico

2.1. Aprendizaje máquina (Machine learning, ML)
2.2. Aprendizaje Supervisado
2.3. Algoritmos de machine learning
2.4. Métricas de evaluación
2.5. Interpretabilidad del modelo

Capítulo 3: Metodología

Capítulo 4: Implementación y desarrollo

4.1. Comprensión De Los Datos
4.2. Preparación de los datos
4.3. Modelado de los algoritmos de aprendizaje de máquina
4.4. Evaluación de los algoritmos de aprendizaje de máquina
4.5. Despliegue

Capítulo 5: Conclusiones

Referencias

A T E N T A M E N T E

"naturam subiecit aliis"

FACULTAD DE INGENIERÍA U.A.CH.

EL DIRECTOR

M.I. FABIÁN VINICIO HERNÁNDEZ MARTÍNEZ

SECRETARIO DE INVESTIGACIÓN Y POSGRADO

DR. FERNANDO MARTÍNEZ REYES

DIRECCIÓN

Dedicatoria

A mi madre querida, por todo su apoyo incondicional. Finalmente lo logramos.

Agradecimientos

Un sincero agradecimiento a mi tutor el Dr. Luis Carlos Gurrola González, quien se hizo responsable de mi plan de estudios y me supo guiar durante el trabajo de tesis. Muchas gracias por la confianza que me diste aún desde los primeros días cuando todavía me encontraba en la Habana.

Un apartado especial para la M.S.I. Karina Rocío Requena Yáñez, coordinadora de la Maestría, muchas gracias por guiarme durante todo el proceso del posgrado. Aún recuerdo hasta su preocupación durante mi primer invierno en Chihuahua sobre si sabía usar correctamente la calefacción. Gracias.

A Oraldo, muchas gracias por todo tu apoyo durante mi estancia.

Al Consejo de la Ciencia y Tecnología (CONACYT) de México por su apoyo económico, sin este hubiera sido muy difícil solventar mi estancia en este maravilloso país.

A la Secretaría de Investigación y Posgrado de la Facultad de Ingeniería de la Universidad Autónoma de Chihuahua, por confiar en mí y darme la oportunidad siendo extranjero.

A cada uno de los profesores que con sus clases apoyaron mi crecimiento como profesional durante este postgrado.

Resumen

Poder predecir cuándo se pagará una factura se convierte en un punto crítico para las empresas dedicadas al factoraje. Este modelo de negocio se basa en la compra de facturas pendientes de empresas que tienen clientes que pagan lentamente y buscan aumentar el flujo de caja. Sin embargo, debido a la complejidad de los datos relacionados con las facturas y al hecho de que el proceso de toma de decisiones es complejo, realizar la predicción de que cuentas se pagarán a tiempo depende de la experiencia a nivel humano, lo que hace que el proceso sea largo y difícil de automatizar. En este trabajo se presenta un sistema end-to-end basado en Machine Learning (ML) que es capaz de ayudar a los ejecutivos a predecir el pago de facturas. Este sistema fue desarrollado en alianza con la empresa Blú Capital. El modelo fue construido usando la librería Scikit-Learn de Python, se usó el framework Flask de Python para el desarrollo de la aplicación encargada de realizar las inferencias y por otra parte se desarrolló una aplicación web cliente para recolectar la información necesaria para poder realizar dicha predicción. Se empleó la metodología CRISP-DM la cual cuenta con todos los pasos a seguir para entregar un sistema listo de ML para producción. Los resultados obtenidos de los algoritmos ML logran un rendimiento similar a los reportados en la literatura. Dada la naturaleza del problema de predicción, así como a requerimientos del cliente se proporciona además una capa de interpretabilidad mediante el uso de valores SHAP para comprender el papel que desempeñan las características en el resultado final, así como el uso del método LIME para la comprensión de predicciones locales.

Índice de Contenido

Índice de Tablas

Índice de Figuras

Capítulo 1: Introducción

Factoraje financiero

La forma de préstamo, que hoy se denomina factoraje, es una de las más antiguas y tiene profundas raíces históricas. Esta forma de préstamo de productos básicos se utilizó en las relaciones comerciales en Babilonia desde el año 4000 a.C. y más tarde en el Imperio Romano. Sin embargo, el factoraje alcanzó su mayor desarrollo en el siglo XIV en Inglaterra, el cual se debió al surgimiento de las relaciones capitalistas y al desarrollo intensivo de la producción textil [1].

A medida que las empresas de factoraje se capitalizaban, se formaban recursos para el pago anticipado de las deudas de los compradores y los pagos a los productores de bienes [1]. Pero, ¿qué es el factoraje? Este es una operación financiera en la que el proveedor cede el derecho de cobro de su factura a un tercero con descuento, a cambio el proveedor recibe un anticipo de efectivo. En este proceso intervienen tres partes, el cedente (proveedor), el prestamista (factor) que adquiere la propiedad de la factura y el deudor (comprador) con la responsabilidad de pagar al legítimo tenedor del derecho de cobro al vencimiento [2]. El proceso de factura a efectivo implica varios pasos, desde la creación de la factura hasta la liquidación o conciliación de la deuda (pago) del cliente. Un paso clave de este proceso es el cobro de las cuentas por cobrar. Las cuentas por cobrar (AR, por sus siglas en inglés) se refieren a las facturas emitidas por una empresa por productos o servicios ya entregados, pero aún no pagados por sus clientes. La gestión adecuada de AR es una actividad contable central y una preocupación de cualquier empresa, relacionada con su flujo de caja [3]. Por lo que cada vez que la empresa no cobra la cantidad correspondiente a una AR, resulta ser una deuda incobrable que representa una pérdida para la empresa. Por lo tanto, con el objetivo de evitar tales pérdidas, los ejecutivos de las empresas requieren encarecidamente administrar las AR y pronosticar las AR que se esperan recibir en un período de tiempo determinado [4]. En consecuencia, la previsión de cuentas por cobrar juega un papel importante para los ejecutivos de la empresa, ya que puede tener un gran impacto en el

capital de trabajo de la misma. A través de la previsión adecuada, la empresa puede obtener una idea de la cantidad de efectivo que recibirá de los clientes contra los servicios que prestó durante un periodo de tiempo. A partir de esta predicción podrá decidir las mejores estrategias para su negocio.

Sin embargo, debido a la complejidad de los datos relacionados con las facturas y al hecho de que el proceso de toma de decisiones no está registrado en el sistema de cuentas por cobrar, realizar esta predicción se convierte en un desafío, por lo que el uso de las tecnologías en especial técnicas de Inteligencia Artificial (IA), como lo es Machine Learning (ML) cobra vital importancia en este proceso.

Los enfoques de modelado predictivo se utilizan ampliamente en una serie de dominios relacionados, como la gestión de crédito y la recaudación de impuestos. La predicción del pago de facturas también puede modelarse como un problema de clasificación.

Antecedentes

Ya existen diferentes investigaciones sobre la predicción de cuentas por cobrar haciendo énfasis en el uso de herramientas de Machine Learning para conocer estos resultados. Los siguientes trabajos analizados dan sustento a la investigación y sirven como punto de partida para la selección de características y algoritmos a emplear.

Uno de ellos es el de Zeng y otros [5], donde los autores lo formulan como un clásico problema de clasificación multiclase haciendo uso de aprendizaje supervisado utilizando modelos predictivos para predecir el comportamiento de pago de una factura, clasificando a los clientes dentro de cinco clases diferentes: a tiempo, de 1 a 30 días, 31 a 60 días, 61 a 90 días o más de 90 días de retraso. Estas clases suelen estar relacionadas con los procesos de AR y las contramedidas para hacer frente a las facturas atrasadas. Para ello emplearon distintos modelos como C4.5 Decision Tree Induction, Naïve Bayes, Logistic Regression, Boosting Decision Stumps y adicionalmente experimentaron con el algoritmo PART. Los mejores resultados se obtuvieron con el algoritmo C4.5 donde alcanzó una exactitud de hasta 92.65%. Sin embargo, se centran principalmente en las facturas de los clientes recurrentes, debido a la riqueza de los datos históricos de la factura, dónde a partir de sólo tres características a nivel de factura (Monto, Plazo de pago y Categoría) construyen el resto de datos históricos. Por último, proponen que a partir de sus resultados construir modelos separados para la predicción de clientes recurrentes.

En el trabajo de Ana Paula y otros [3] presentan un prototipo para predecir si una factura se pagará a tiempo o no, obteniendo un 81% de exactitud en sus resultados. Este se define como un problema clásico de clasificación binaria usando aprendizaje supervisado. Un detalle a destacar es que su Dataset no contenía información relevante acerca de los clientes y por ello solo trabajan con información relativa a la factura. A partir de esta información, usan los datos históricos para crear nuevas características (paid invoice, total paid invoices, sum amount paid invoices, total invoices late, sum amount late invoices, total outstanding invoices, total outstanding late, sum total outstanding, sum late outstanding,

average days late, average days outstanding late, standard deviation invoices late, standard deviation invoices outstanding late, payment frequency difference). Un aporte de este artículo es que definen que el comportamiento de pago reciente de un cliente tiene mayor poder predictivo que todo su historial de pago, por lo que obtienen como resultado una ventana de tiempo de 4 meses hacia atrás. Hacen uso de diferentes algoritmos como XGBoost, Naïve Bayes, Logistic Regression, k-NN, Random Forest y DNN, obteniendo los mejores resultados con XGBoost.

De la misma manera, Bailey y otros [6] analizan varias estrategias para priorizar las llamadas de cobranza. Ellos proponen usar modelos predictivos basados en regresión logística binaria y análisis discriminativo para determinar qué clientes entregar a una agencia de cobranza externa para un procesamiento de cobranza adicional.

En el trabajo de Parth Kapadia y otros [4] explican que basado en sus resultados experimentales, un ensemble de un K-Means clustering y Random Forest Classifier produce mejores resultados para predecir las cuentas por cobrar de sus empresas en un período de tiempo específico. Otros algoritmos analizados fueron Logistic Regression y Random Forest Classifier. La métrica de evaluación utilizada fue la precisión de predicción. El modelo planteado está limitado a clientes recurrentes, todas las características que usa para la construcción del algoritmo son relativas a la factura.

El artículo [7], haciendo uso de los datos históricos de las facturas ya pagadas de los clientes, su autor demuestra cómo con la ayuda de algoritmos de ML se puede predecir el comportamiento de pago de facturas que aún no han sido pagadas. Abordan el problema como un modelo de clasificación multiclase intentado clasificar la factura dentro de uno de los siguientes grupos: (A tiempo,1-30, 31-60, 61-90, 91-120, 121-150, Más de 150 días). Para dar solución al problema usan el algoritmo Adaboost Decision Tree logrando una exactitud por encima del 85%. Sin embargo, es de notar que la exactitud del modelo disminuye a medida que se acerca al último grupo de clasificación (Más de 150 días)

siendo esta de tan solo el 50% aproximadamente. Por otra parte, el autor no hace mención al comportamiento dado nuevos clientes.

En el trabajo [8], se aborda el problema de la predicción de pagos de facturas morosas de forma anticipada con aprendizaje automático, todo esto a partir de datos históricos de facturas. Demuestra cómo se pueden utilizar modelos de aprendizaje supervisado para detectar las facturas que tendrían retrasos en los pagos, así como los clientes problemáticos. Los modelos analizados fueron Decision Tree, Random Forests, AdaBoost, Logistic Regression y Suppor Vector Machine (SVM). El Random Forests obtuvo los mejores resultados tanto para la predicción binaria como la predicción multiclase.

En el trabajo [9], se tiene como objetivo identificar el algoritmo con mayor exactitud usando una comparativa de técnicas de ML para detectar comportamiento de pago de clientes con cuentas por cobrar. La metodología utilizada está basada KDD. Los algoritmos analizados fueron Decision Tree, Logistic Regression, Naives Bayes, KNN, SVM, Multilayer perceptron, Random Forests. Tanto el Random Forest como el Decision Tree presentan los mejores resultados, pero se toma como mejor algoritmo el Decision Tree en base a una menor tasa de error y un menor tiempo de ejecución.

En el trabajo [10], se realiza un modelo predictivo cuya objetivo es predecir si se le otorga un crédito hipotecario a un cliente. La metodología utilizada fue CRISP-DM. Los algoritmos analizados fueron Decision Tree Classifier, Support Vector Machine, Naive Bayes, Random Forest Classifier, y Logistic Regression siendo este último el ganador por la métrica ROC_AUC de 0.71. Se validó el modelo mediante Cross Validation, con un KFold de 10, cuyos resultados fueron de 0.70 y 0.0073 para la media y la desviación estándar respectivamente.

En el artículo [11], se evalúan los algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de créditos para predecir si otorgar una tarjeta de crédito o no

a un cliente. El modelo con mejor resultado fue XGBoost teniendo en cuenta la métrica AUC.

En el trabajo [12], se tiene como objetivo haciendo uso de algoritmos de ML, predecir la probabilidad de acuerdo de pago de un deudor en una Agencia de Cobro de Deudas (DCA). La metodología utilizada fue CRISP-DM. Los algoritmos analizados fueron Random Forrest Classifier, Gradient Boosting Machine, Regresión logística y un Multilayer Perceptron, utilizando una técnica de submuestreo aleatorio. El rendimiento se comparó utilizando las métricas de evaluación de sensibilidad, especificidad y AUC. Siendo el algoritmo Gradient Boosting Machine el seleccionado con una puntuación de sensibilidad de 0,97, especificidad de 0,93 y AUC de 0,98 en el conjunto de validación.

Como es posible observar en los resúmenes de las investigaciones anteriores algunos suelen emplear como metodología CRISP-DM [13], la cual ha sido seleccionada para el presente trabajo. A su vez, si bien ya existen una serie de publicaciones sobre el tema, no hay un consenso adecuado sobre cómo abordarlo pues varía en dependencia de los datos que posea cada empresa es por ello que en este trabajo se busca construir un modelo robusto o aplicar uno existente que se adapte a las características del negocio, y que sea capaz de predecir con la mayor precisión posible.

Preguntas de la investigación

¿Qué características de la factura pueden ser utilizadas para el problema de predicción de cuentas por cobrar en un contexto de Machine Learning?

¿Cuáles algoritmos de Machine Learning presentan un mejor desempeño en la clasificación del comportamiento de pago de las cuentas por cobrar?

¿Cómo interpretar el comportamiento del modelo?

¿Cómo permitir que los funcionarios de Blú Capital puedan tomar acciones preventivas relacionadas a clientes que tienen facturas con un riesgo de pago tardío?

Objetivos

Con el fin de poder responder cada una de las preguntas de la investigación planteadas se traza como **objetivo general**:

Desarrollar un sistema que haga uso de un modelo de ML para predecir el comportamiento de pago de las cuentas por cobrar.

Para dar seguimiento al objetivo general se definen los siguientes **objetivos específicos**:

- Identificar y evaluar algoritmos de ML usados en la literatura enfocados en el problema de predicción de AR.
- Proporcionar una capa de interpretabilidad que permita conocer como el modelo seleccionado realiza las predicciones.
- Crear una aplicación que permita al cliente realizar predicciones de pago de AR.

Justificación

En los últimos tiempos, varias firmas/empresas brindan servicios a sus clientes a crédito. Estas firmas reciben los pagos contra los servicios y/o compras prestadas a los clientes, pero no pagadas, denominadas Cuentas por Cobrar. Con el objetivo de planificar las finanzas además de decidir estrategias para sus negocios, los ejecutivos de estas empresas desean saber las cuentas por cobrar de sus empresas para un período de tiempo específico [4], por lo que poder predecir cuándo se pagarán estas cuentas por cobrar se convierte en un eje estratégico en múltiples industrias y respalda los procesos de toma de decisiones en la mayoría de los flujos de trabajo financieros.

Sin embargo, el desafío en ese ámbito consiste en tratar con datos complejos, así como la falta de datos relacionados con los procesos de toma de decisiones no registrados en el sistema de cuentas por cobrar.

Es por ello que en el presente trabajo se pretende, a través del desarrollo de un sistema que haga uso de un modelo de ML, predecir cuándo se pagará una cuenta por cobrar y de esta manera proveer a los directivos de la empresa de una herramienta que les apoye en la toma de decisiones. Actualmente es un proceso que se realiza de manera manual y con esta nueva propuesta ayudaría a fundamentar mejor las decisiones que se tomen.

Capítulo 2: Marco teórico

En el presente capítulo se describe el conocimiento científico necesario para sustentar el trabajo de investigación, así como los algoritmos utilizados en el mismo.

2.1. Aprendizaje máquina (Machine learning, ML)

El aprendizaje automático es un subcampo de la informática que evolucionó a partir del estudio del reconocimiento de patrones y la teoría del aprendizaje computacional en inteligencia artificial. El aprendizaje automático explora la construcción y el estudio de algoritmos que pueden aprender y hacer predicciones sobre los datos. Dichos algoritmos funcionan construyendo un modelo a partir de entradas de ejemplo para hacer predicciones o decisiones basadas en datos, en lugar de seguir instrucciones de programa estrictamente estáticas [14].

Las tareas de aprendizaje automático generalmente se clasifican en tres categorías amplias, las cuales son: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo [14]. En el presente trabajo se busca desarrollar modelos clasificadores, por lo que se empleará el aprendizaje supervisado. Los algoritmos a tener en cuenta serán DecisionTree Classifier, Random Forest y XGBoost Classifier.

2.2. Aprendizaje Supervisado

El aprendizaje supervisado en ML se aplica cuando cada dato, o conjunto de datos de entrada (muestra) tiene asociada una etiqueta. Los algoritmos de aprendizaje supervisado se utilizan más comúnmente en tareas de regresión o de clasificación. Estos algoritmos aprenden a reconocer patrones en conjuntos de datos. Una vez que encuentran esos patrones, los aplican en datos nunca antes vistos para clasificarlos o predecir el valor de una de sus propiedades. Para que sean capaces de encontrar estos patrones, necesitan de datos ejemplo. Los ejemplos tienen que contener todas las características y la respuesta correcta sobre el valor buscado o la clase a la que pertenece, o sea cada dato, o conjunto de datos de entrada (muestra) tiene asociada una etiqueta. Así logra aprender las relaciones que hay entre las características dadas y el objetivo [15], [16].

2.3. Algoritmos de machine learning

A continuación, se describen los algoritmos de Machine Learning empleados en el trabajo.

2.3.1. Árboles de decisión (Decision Tree Classifier)

Este modelo utiliza un árbol de decisión como modelo predictivo que mapea las observaciones sobre un elemento a conclusiones sobre el valor objetivo del elemento. Es uno de los enfoques de modelado predictivo utilizados en estadísticas, minería de datos y aprendizaje automático. Los modelos de árbol de decisión en los que la variable de destino puede tomar un conjunto finito de valores se denominan árboles de clasificación. En estas estructuras de árbol, las hojas representan etiquetas de clase y las ramas representan conjunciones de características que conducen a esas etiquetas de clase. Los árboles de decisión en los que la variable objetivo puede tomar valores continuos (normalmente números reales) se denominan árboles de regresión [14].

En la Figura 1 se muestra un Decision Tree tomado de [14], ("sibsp" es el número de cónyuges o hermanos a bordo). Las cifras debajo de las hojas muestran la probabilidad de supervivencia y el porcentaje de observaciones en la hoja:

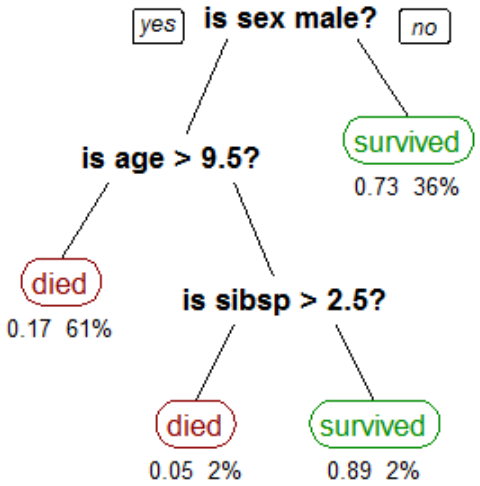

Figura 1 Un árbol que muestra la supervivencia de los pasajeros en el Titanic.

2.3.2. Bosques Aleatorios (Random Forests Classifier)

Este es un método de aprendizaje de conjuntos para tareas de clasificación, regresión y así como otras. Este algoritmo se basa en la idea de que, si un árbol de decisión es bueno, varios generados aleatoriamente podrían ser mejores, para ello opera mediante la construcción de una multitud de árboles de decisión en el momento del entrenamiento y la salida de la clase que es la moda de las clases (clasificación) o predicción media (regresión) de cada uno de los árboles individuales.

Los bosques aleatorios son una forma de promediar múltiples árboles de decisión profundos, entrenados en diferentes partes del mismo conjunto de entrenamiento, con el objetivo de reducir la varianza. Esto se produce a expensas de un pequeño aumento en el sesgo y cierta pérdida de interpretabilidad, pero generalmente aumenta en gran medida el rendimiento del modelo final.

Los bosques aleatorios corrigen el hábito de los árboles de decisión de sobre ajustarse a su conjunto de entrenamiento [14].

2.3.3. eXtreme Gradient Boosting (XGBoost)

Es una versión escalable y mejorada del algoritmo de aumento de gradiente diseñado para la eficacia, la velocidad computacional y el rendimiento del modelo. Es una biblioteca de código abierto y forma parte de la comunidad de aprendizaje automático distribuido. XGBoost es una combinación perfecta de capacidades de software y hardware diseñada para mejorar las técnicas de impulso existentes con precisión en el menor tiempo posible [17].

El algoritmo XGBoost tiene las siguientes características [11]:

- Consiste en un ensamblado secuencial de árboles de decisión (este ensamblado se conoce como CART, acrónimo de “Classification and Regression Trees”). Los árboles se agregan secuencialmente a fin de aprender del resultado de los árboles previos y corregir el error producido por los mismos, hasta que ya no se pueda corregir más dicho error (esto se conoce como “gradiente descendente”).

- La principal diferencia entre los algoritmos XGBoost y Random Forest es que en el primero el usuario define la extensión de los árboles mientras que en el segundo los árboles crecen hasta su máxima extensión.
- Utiliza procesamiento en paralelo, poda de árboles, manejo de valores perdidos y regularización (optimización que penaliza la complejidad de los modelos) para evitar en lo posible sobreajuste o sesgo del modelo.

2.3.4. Logistic Regression

Lo primero que hay que decir es que la regresión logística no es una regresión, sino un algoritmo de aprendizaje de clasificación. El nombre proviene de la estadística y se debe a que la formulación matemática de la regresión logística es similar a la de la regresión lineal [18].

Este algoritmo mide la relación entre la variable dependiente categórica y una o más variables independientes, que suelen ser (pero no necesariamente) continuas, mediante la estimación de probabilidades. Por tanto, trata el mismo conjunto de problemas que la regresión probit utilizando técnicas similares; el primero asume una función logística y el segundo una función de distribución normal estándar.

La regresión logística se utiliza ampliamente en muchos campos. Puede usarse para predecir si un paciente tiene una enfermedad determinada (por ejemplo, diabetes, enfermedad coronaria), basándose en las características observadas del paciente. La técnica también se puede utilizar en ingeniería, especialmente para predecir la probabilidad de falla de un proceso, sistema o producto determinado [14].

2.3.5. AdaBoost Classifier

Este algoritmo AdaBoost (adaptive boosting) crea un conjunto de learners deficientes al mantener una colección de ponderaciones sobre los datos de entrenamiento y los ajusta de forma adaptativa después de cada ciclo de aprendizaje débil. Los pesos de las muestras de entrenamiento que están mal clasificadas por el learner débil actual se

incrementarán, mientras que los pesos de las muestras que están clasificadas correctamente se reducirán [19].

AdaBoost es uno de los algoritmos de aprendizaje automático más prometedores, de convergencia rápida y fácil de implementar. No requiere conocimientos previos sobre el learner débil y se puede combinar fácilmente con otros métodos para encontrar hipótesis débiles, como la máquina de vectores de soporte [19].

2.3.6. KNeighbors Classifier

El algoritmo del vecino más cercano (KNN) pertenece a la clase de métodos estadísticos de reconocimiento de patrones. El método no impone a priori ninguna suposición sobre la distribución de la que se extrae la muestra del modelo. Se trata de un conjunto de entrenamiento con valores tanto positivos como negativos. Una nueva muestra se clasifica calculando la distancia al caso de entrenamiento vecino más cercano. El signo de ese punto determinará la clasificación de esa muestra. En el clasificador de k vecinos más cercanos, se consideran los k puntos más cercanos y se utiliza el signo de la mayoría para clasificar la muestra. El rendimiento del algoritmo KNN está influenciado por tres factores principales: (1) la medida de distancia utilizada para localizar a los vecinos más cercanos; (2) la regla de decisión utilizada para derivar una clasificación de los k vecinos más cercanos; y (3) el número de vecinos utilizados para clasificar la nueva muestra. Se puede demostrar que, a diferencia de otros métodos, este método es universalmente asintóticamente convergente, es decir: a medida que aumenta el tamaño del conjunto de entrenamiento, si las observaciones son independientes y están distribuidas idénticamente, independientemente de la distribución de la que se extrae la muestra, la clase predicha convergerá a la asignación de clase que minimiza el error de clasificación errónea [14].

2.3.7. Support Vector Machine

En ML, las máquinas de vectores de soporte (SVM) son modelos de aprendizaje supervisados con algoritmos de aprendizaje asociados que analizan datos y reconocen patrones, y se utilizan para clasificación y análisis de regresión. Dado un conjunto de

ejemplos de entrenamiento, cada uno marcado por pertenecer a una de dos categorías, un algoritmo de entrenamiento SVM construye un modelo que asigna nuevos ejemplos a una categoría u otra, convirtiéndolo en un clasificador lineal binario no probabilístico. Un modelo SVM es una representación de los ejemplos como puntos en el espacio, mapeados de modo que los ejemplos de las categorías separadas estén divididos por una brecha clara que sea lo más amplia posible. Luego se asignan nuevos ejemplos a ese mismo espacio y se predice que pertenecen a una categoría según el lado de la brecha en el que se encuentran [14].

Más formalmente, una máquina de vectores de soporte construye un hiperplano o un conjunto de hiperplanos en un espacio de alta o infinita dimensión, que puede usarse para clasificación, regresión u otras tareas. Intuitivamente, el hiperplano que tiene la mayor distancia al punto de datos de entrenamiento más cercano de cualquier clase logra una buena separación (el llamado margen funcional), ya que en general, cuanto mayor es el margen, menor es el error de generalización del clasificador [14].

2.3.8. Multinomial Naive Bayes

En el aprendizaje automático, los clasificadores Naive Bayes son una familia de clasificadores probabilísticos simples basados en la aplicación del teorema de Bayes con una suposición fuerte (naive) de independencia entre las características [14].

Esta es una técnica simple para construir clasificadores: modelos que asignan etiquetas de clase a instancias de problemas, representadas como vectores de valores de características, donde las etiquetas de clase se extraen de algún conjunto finito. No es un algoritmo único para entrenar tales clasificadores, sino una familia de algoritmos basados en un principio común: todos los clasificadores Naive Bayes asumen que el valor de una característica particular es independiente del valor de cualquier otra característica, dada la variable de clase. Por ejemplo, una fruta puede considerarse manzana si es roja, redonda y mide unos 3 cm de diámetro. Un clasificador Naive Bayes considera que cada una de estas características contribuye de forma independiente a la probabilidad de que esta fruta sea

una manzana, independientemente de cualquier posible correlación entre las características de color, redondez y diámetro [14].

2.4. Métricas de evaluación

Para evaluar el desempeño de modelos clasificadores se utilizan las métricas de

Accuracy y F1-score. Estas métricas basan sus cálculos a partir de una matriz de confusión la cual muestra el número de verdaderos positivos (TP), verdaderos negativos (TN), falsos positivos (FP) y falsos negativos (FN).

Tabla 1 Matriz de confusión

		PREDICCIÓN	
		Positivos	Negativos
VALORES REALES	Positivos	TP	FN
	Negativos	FP	TN

2.4.1. Accuracy

Es la fracción de predicciones que el modelo realizó correctamente. Esta se encuentra en un rango entre 0 y 1 [15].

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

2.4.2. Precision

Muestra la proporción de aciertos en la predicción a través de la fracción de valores clasificados correctamente de una sola clase entre el total de valores predichos en esa misma clase. [15].

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

2.4.3. Recall

Es la fracción de ejemplos de una clase predichos correctamente, entre la cantidad total de ejemplos pertenecientes a dicha clase. Proporciona un valor de calidad relativo al número total de muestras positivas [15].

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

2.4.4. F1-score

Es la media armónica entre *precision* y *recall*, se calcula de la siguiente manera [15].

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

2.5. Interpretabilidad del modelo

Poder comprender las razones detrás de las predicciones es bastante importante para evaluar la confianza del modelo, lo cual es fundamental si se van a tomar medidas basadas en esa predicción. A continuación, se plantean dos técnicas usadas para ayudar a interpretar los modelos usados en el presente trabajo.

2.5.1. SHAP (SHapley Additive exPlanation) Values

Es un framework para la interpretación de predicciones. Se propuso originalmente para estimar la importancia de un jugador individual en un equipo colaborativo. Este calcula los valores SHAP cuantificando la contribución de cada característica a la predicción del modelo, mediante la incorporación de conceptos de teoría de juegos y explicaciones locales. A diferencia de otros enfoques, SHAP se ha justificado como el único enfoque consistente de atribución de características con varias propiedades únicas que concuerdan con la intuición humana. Debido a sus sólidas garantías teóricas, SHAP se

convierte en una de las principales recomendaciones de enfoques de interpretación de modelos en la industria [20]–[22].

Destacar que Lundberg y Lee [20] demuestran que SHAP está mejor alineado con la intuición humana que otros métodos de interpretación de ML. Mencionar que existen tres beneficios que vale la pena destacar [22].

- **Interpretabilidad global**: los valores SHAP colectivos pueden mostrar cuánto contribuye cada predictor, ya sea positiva o negativamente, a la variable objetivo.
- **Interpretabilidad local**: cada observación obtiene su propio conjunto de valores SHAP. Esto aumenta en gran medida su transparencia. Podemos explicar por qué un caso recibe su predicción y las contribuciones de los predictores
- El framework SHAP sugiere una **aproximación agnóstica del modelo** para los valores SHAP, pudiéndose calcular para cualquier modelo ML.

2.5.2. LIME

Explicaciones locales interpretables agnósticas del modelo (LIME) plantea una novedosa técnica que explica las predicciones de cualquier clasificador de forma interpretable y fiel. Representando predicciones individuales representativas y sus explicaciones de forma no redundante proporcionando una visión global del modelo a los usuarios [23].

LIME admite explicaciones para modelos tabulares, clasificadores de texto y clasificadores de imágenes. En la Figura 2 se muestra un modelo que predice que un paciente tiene gripe y LIME resalta los síntomas en el historial del paciente que llevaron a la predicción. El estornudo y el dolor de cabeza se describen como factores que contribuyen a la predicción de la "gripe", mientras que la "ausencia de fatiga" es una evidencia en contra. Con estos datos, un médico puede tomar una decisión informada sobre si confiar en la predicción del modelo [23].

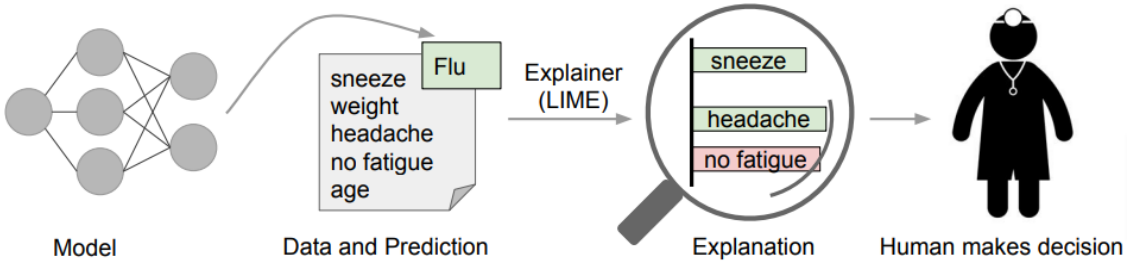

Figura 2 Explicando las predicciones individuales con LIME.

Capítulo 3: Metodología

El desarrollo de este trabajo ha sido implementado siguiendo la metodología Cross Industry Standard Process for Data Mining (CRISP-DM) representada en la Figura 3,la cual propone un proceso iterativo incremental compuesto por seis fases Figura 3.

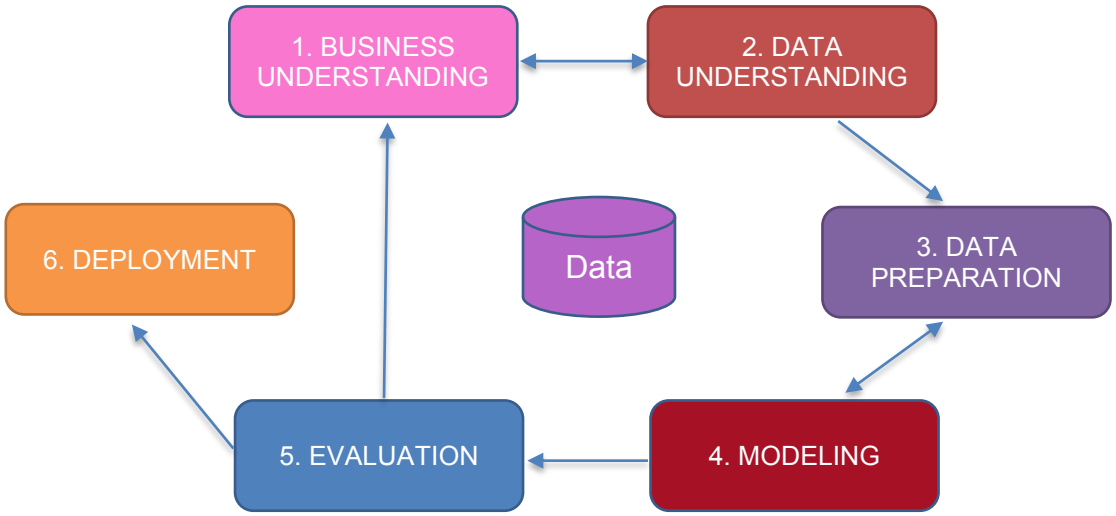

Figura 3 Metodología CRISP-DM [13],

Tal cual propone Schröer en su trabajo [13], la primera fase es Entendimiento del negocio, dónde se pretende determinar el objetivo del negocio, el tipo de problema, los criterios de éxito y un plan de proyecto obligatorio. La segunda fase es Entendimiento de los datos, para ello recopilar datos de fuentes de datos, explorarlos y describirlos y verificar la calidad de los datos son tareas esenciales en esta fase. La tercera fase es Preparación de los datos, donde se establecen criterios de inclusión y exclusión de las características que se usarán, en conjunto con un pre procesamiento y limpieza de los datos. La cuarta fase es Modelado, la cual consiste en la selección y construcción de los modelos adecuados, estableciendo sus parámetros específicos y evaluando según los criterios de éxito elegidos. Es en la quinta fase Evaluación, donde se revisan los resultados del modelo considerando los objetivos de negocio. Por último, en la fase final se encuentra el Despliegue, donde se estable una integración formal del trabajo anterior para que pueda ser usado por terceras partes.

A continuación, se describe la implementación de cada una de las fases de CRISP-DM en el presente trabajo:

- Fase 1. La comprensión del negocio comenzó con un acercamiento al jefe de innovación tecnológica de Blú Capital para una explicación de cómo funciona el proceso de factoraje, estrategias utilizadas, así como del conjunto de datos proporcionados para su análisis. Para ello se tomó como base de datos un Dataset proporcionado por Blú Capital. Estos registros corresponden a facturas que habían sido pagadas en diferentes periodos de tiempo. Dicha base de datos contiene registros que datan desde el 2016.
- Fase 2. Para la fase de comprensión de los datos se realiza un análisis exploratorio de los datos (EDA), cuyo propósito es "resumir los datos recopilados de manera significativa (…), esto incluye cuantificar pensamientos cualitativos en resúmenes de datos cuantitativos" [24]. Para ello se realizaron los siguientes pasos: categorización de variables de entrada en continuas y discretas, identificación y gestión de valores nulos, análisis de sesgo y desequilibrio de los datos, identificación de la variable objetivo.
- Fase 3. La preparación de los datos se inicia a partir del proceso EDA realizado en la fase anterior. Los valores faltantes se tratan utilizando una imputación de K-vecinos más cercanos en los datos numéricos y una imputación simple para datos categóricos. Todas las características categóricas se codifican de forma ficticia. Se eliminan las características innecesarias o que no aportan información relevante. Se eliminan los espacios en blanco al inicio y al final de las celdas de tipo cadena. Se renombran los atributos a un formato estándar, se formatean las fechas a un formato de fecha único y válido y se consolidan ambas fuentes de datos en una única base de datos. Por último, se aplica una técnica de extracción de características.

  Basado en el artículo [5], donde propone a partir de las características básicas de la factura construir nuevas características históricas. La Tabla 4, muestra todas estas características históricas y agregadas que se generan (características numeradas 6-19). Estas características proporcionan una cantidad significativa de información que se puede aprovechar para predecir el resultado de una nueva factura.

- Fase 4. En la fase de modelado se probaron los resultados usando los siguientes algoritmos: Decision Tree Classifier, XGBoost Classifier, Random Forest, LogisticRegression, AdaBoostClassifier, KNeighborsClassifier, SVM, Multinomial Naive Bayes.
- Fase 5. Para la fase de evaluación fue usado como métricas el nivel de exactitud (*Accuracy*) y el *F1 Score* para seleccionar el algoritmo de mejor rendimiento.
- Fase 6. En la fase de despliegue, una vez identificado el modelo seleccionado se exportó el mismo en formato joblib. Una aplicación implementada en Flask framework utiliza este modelo permitiendo que sea visible a través de un endpoint para que sea accedido por diferentes aplicaciones. Adicionalmente se desarrolló una aplicación web que permita al usuario interactuar con el modelo de manera visual. En ella el cliente puede introducir los valores que son requeridos para que el algoritmo pueda arrojar su predicción.

Capítulo 4: Implementación y desarrollo

En este capítulo se describe la implementación de la metodología CRISP-DM, la cual fue seleccionada para el desarrollo del trabajo. A continuación, se presenta el proceso de adquisición de la base de datos, así como las técnicas de preprocesamiento de los datos realizadas. Además, se describe la implementación de los algoritmos de ML propuestos, así como las diferentes métricas empleadas para su evaluación. Posteriormente, se detalla el proceso de despliegue del algoritmo entrenado que obtuvo los mejores resultados.

4.1. Comprensión De Los Datos

Si bien ya existen un número de bases de datos públicas sobre las cuales se han enfocado diversas investigaciones para abordar el tema de la predicción de pago de cuentas por cobrar, para el presente trabajo se usó una base de datos propiedad de Blú Capital. Para esto se tuvo en cuenta además de la necesidad de apoyar en este proceso a la empresa, la abundancia de datos con que esta contaba.

4.1.1. Descripción de los datos

La base de datos entregada consiste en un fichero en formato *csv* compuesto a su vez por **24010** registros y **30** columnas.

A continuación, se describen cada una de las columnas.

Tabla 2 Columnas de la base de datos de facturas pagadas

Atributo	Tipo de atributo	Rango de valores	Relación con otros atributos	Descripción	Es relevante para los OE del proyecto
folio_factura	String		No	Folio de la factura	No
folio_credito	String			Folio de la línea de crédito a la que pertenece	No

factura_importe	String			Importe de la factura	Si
factura_moneda	String	MXN, USD		Moneda en que se emite la factura	No
credito_fecha_inicio	String	2017-11-10 - 2023-01-23		Fecha en que se emite la factura	Si
credito_fecha_fin	String	2017-12-25 - 2023-06-09		Fecha fin para pagar la factura	Si
fecha_pago	String	2017-11-30 - 2023-01-31		Fecha de pago de la factura	Si
dias_financiar	Integer	1 - 401	credito_fecha_inicio, credito_fecha_fin	Días financiados	No
aplica_interes_en_dispersion	Boolean	False, True		Si aplica intereses	No
porcentaje_adelanto	Float	85 - 100		Porciento que hay que pagar de la factura por adelantado	No
linea_tipo	String	Blú Index, Factoraje Blú		Nombre del tipo de línea	No
contrato_marco	String			Contrato marco al que pertenece la factura	No

linea_limite	String	0 - 20000000		Monto de la línea de crédito	Si
linea_moneda_principal	String	MXN, USD		Moneda principal de la línea de crédito	No
linea_monedas_permitidas	String	MXN, USD		Lista de monedas permitidas	No
cliente_rfc	String			RFC del cliente	No
cliente_razon_social	String	Nomenclador		Razón social del cliente	Si
cliente_regimen_fiscal	String	PM, PFAE		Régimen fiscal del cliente. Si es persona física o persona moral.	Si
cliente_pais	String	MEX		País al que pertenece el cliente	No
cliente_fecha_nacimiento	String	1940-02-06 - 2022-09-27		Fecha de nacimiento del cliente o constitución de la empresa, se puede utilizar para calcular antigüedad del cliente	Si
cliente_ocupacion	String	Nomenclador		Ocupación del cliente	Si

cliente_sector	String	Privado, Micro Empresa, Rural		Sector al que pertenece el cliente	Si
codigo_scian	Float		codigo_scian_nombre	Código SCIAN que clasifica la industria del ciente	No
codigo_scian_nombre	String		codigo_scian	Nombre del código SCIAN	Si
riesgo_pld	String	Bajo, Medio		Matriz de riesgo	No
pagador_rfc	String			Código RFC del pagador	Si
pagador_nombre	String			Nombre del pagador	No
pagador_pais	String	MEX, USA		País al que pertenece el pagador	No
pagador_scian	Float		pagador_scian_nombre	Código SCIAN que clasifica la industria del pagador	No
pagador_scian_nombre	String		pagador_scian	Nombre del código SCIAN	Si

Una vez analizados los datos se encontraron una serie de problemas que necesitaban ser resueltos para poder proseguir con el siguiente paso. La Tabla 3 muestra la solución propuesta para cada uno de ellos.

Tabla 3 Verificación de los datos

Problemas de calidad identificados	**Soluciones propuestas**
El campo folio de la factura co ntiene 4 valores nulos	Se ignora, pues la columna no es relevante para la solución del problema
Fecha pago con 7 valores nulos	Eliminar los registros completos
Nombre del código scian contiene 2390 valores nulos	Se ignora, pues la columna no es relevante para la solución del problema
El campo país del pagador contiene 4768 valores nulos	Se ignora, pues la columna no es relevante para la solución del problema
Nombre del código scian del pagador contiene 4944 valores nulos	Se ignora, pues la columna no es relevante para la solución del problema
El importe de la factura contiene signo de peso, coma como separador de miles y punto como separador decimal	Eliminar los caracteres no numéricos
El campo línea limite contiene signo de peso, coma como separador de miles y punto como separador decimal	Eliminar los caracteres no numéricos

4.1.2. Exploración de los datos

Como parte del proceso EDA, se muestran una serie de análisis a partir de los datos. En la Figura 4 se muestra la distribución de los datos de acuerdo a la fecha de pago de las facturas. Es posible observar claramente cómo el conjunto de datos se encuentra desbalanceado, donde el mayor número de facturas fueron pagadas a tiempo.

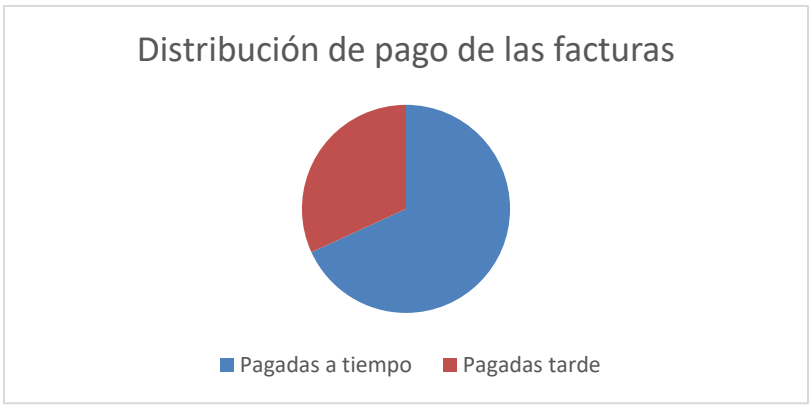

Figura 4 Distribución de pago de las facturas.

En la Figura 5 se muestra el número de facturas por clientes por años.

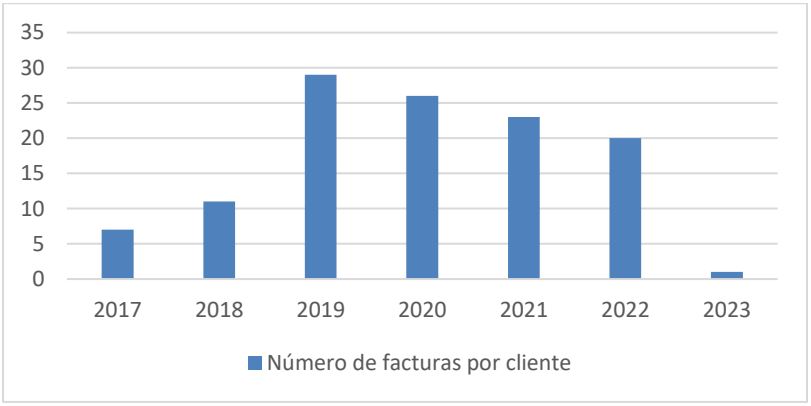

Figura 5 Número de facturas por cliente

En la Figura 6 se muestra como se ha comportado el pago de las facturas por sector.

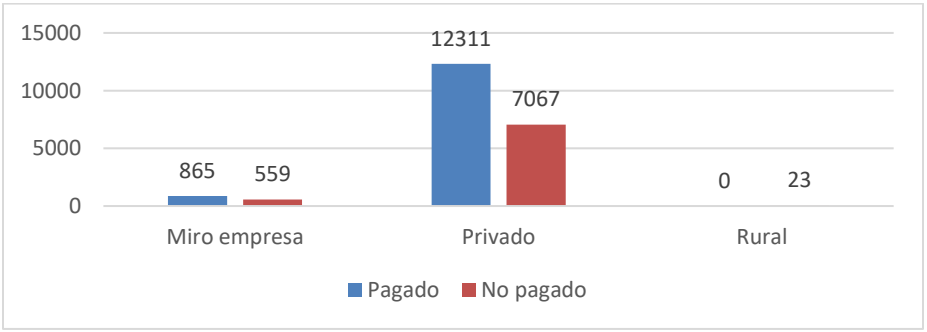

Figura 6 Relación de pago de las facturas por sector

4.2. Preparación de los datos

La preparación de los datos es una de las fases más importantes y la que más tiempo suele consumir. Permite limpiar los datos, integrarlos, hacer la ingeniería de características, así como seleccionar las variables necesarias para la construcción del modelo. Para ello fue empleado la librería Pandas de Python ampliamente utilizada hoy en día para el procesamiento de datos.

4.2.1. Eliminación de valores duplicados

Se definen las columnas *folio_factura* y *folio_crédito* en combinación como llave para identificar registros duplicados. Se obtiene como resultado ningún registro duplicado.

4.2.2. Extracción y selección de características

Este paso consiste en obtener información relevante a partir de los datos almacenados creando características que permitan mejorar la tarea de clasificación.

El uso de características agregadas aumenta significativamente la cantidad de información sobre el pago. Aunque existen algunos estudios que recomiendan qué variables usar [5], es complejo depender de enfoques similares, ya que podría darse el caso de que los conjuntos de datos utilizados para la misma tarea no compartan los mismos datos. Por lo tanto, se realizó un proceso de extracción de características específico para este caso. La Tabla 4 presenta todas las características históricas y agregadas, y su descripción, extraídas de las bases de datos de Blú Capital.

La construcción de la columna de destino (Periodo de pago) fue de la siguiente forma:

$$Días\ transcurridos = Fecha\ de\ pago - Fecha\ de\ vencimiento\ de\ la\ factura$$

Luego, clasificando el valor de la variable *Días transcurridos* en una de las siguientes categorías se obtiene el *Periodo de pago*.

1. Antes de tiempo
2. En tiempo
3. 1-7 Días tarde

4. 8-21 Días tarde
5. Más de 21 días

Tabla 4 Resumen de características

No.	Característica	Descripción
1.	Importe base de la factura	Importe de la factura
2.	Porcentaje de adelanto	Porcentaje del monto de la factura que el cliente debe pagar por adelantado
3.	Límite de la línea de crédito	Límite de la línea de crédito del cliente
4.	Edad del cliente	La edad del cliente
5.	Plazo de pago	El plazo de pago adeudado
6.	Número total de facturas pagadas	Número de facturas pagadas anteriores a la fecha de creación de una nueva factura de un cliente.
7.	Número de facturas que se pagaron con retraso	Número de facturas que se pagaron tarde antes de la fecha de creación de una nueva factura de un cliente
8.	Ratio de facturas pagadas que se retrasaron	Ratio de la característica 7. entre la 6.
9.	Suma del importe base del total de facturas pagadas	La suma del monto base de todas las facturas pagadas antes de una nueva factura para un cliente
10.	Suma del monto base de las facturas que se pagaron con retraso	La suma del monto base de todas las facturas pagadas que se retrasaron antes de una nueva factura para un cliente
11.	Relación de la suma de la cantidad base pagada que se atrasó	Ratio de la característica 10. entre la 9.
12.	Promedio de días de atraso de facturas pagadas en atraso.	Promedio de días de retraso de todas las facturas pagadas que estaban retrasadas antes de una nueva factura para un cliente
13.	Número total de facturas pendientes	Número de las facturas pendientes antes de la fecha de creación de una nueva factura de un cliente.
14.	Número de facturas pendientes que ya estaban atrasadas	Número de facturas pendientes que estaban retrasadas antes de la fecha de creación de una nueva factura de un cliente
15.	Ratio de facturas pendientes que se retrasaron	Ratio de la característica 14. entre la 13.
16.	Suma del importe base del total de facturas pendientes	La suma del monto base de todas las facturas pendientes antes de una nueva factura para un cliente
17.	Suma del importe base de las facturas pendientes que se retrasaron	La suma del monto base de todas las facturas pendientes que se retrasaron antes de una nueva

		factura para un cliente.
18.	Ratio de la suma del importe base pendiente que se atrasó	Ratio de la característica 17. entre la 16.
19.	Promedio de días de atraso de facturas pendientes de atraso.	Promedio de días de retraso de todas las facturas pendientes que estaban retrasadas antes de una nueva factura para un cliente.
20.	Periodo de pago	Columna de destino, indica el tiempo que se tardó en pagar la factura.

Después de la extracción de características, el conjunto de datos quedó compuesto por *24 003* registros, donde cada registro está formado por 20 características (las primeras 19 características de la Tabla 4) y una columna de destino (columna 20 en la misma tabla). Las características históricas están enumeradas en la misma tabla de la 6 a la 19. El número de clientes únicos en los datos es *566*.

4.3. Modelado de los algoritmos de aprendizaje de máquina

Teniendo en cuenta un requerimiento de Blú Capital al buscar modelos de ML para abordar este problema, fue que se deberían evaluar modelos que pudieran permitir, hasta cierto punto, un análisis de interpretabilidad de la lógica que sigue el modelo para producir una salida. Teniendo esto en cuenta, se comenzó con un método adecuado para este propósito: un clasificador de árboles de decisión, ya que se basa en pasos basados en reglas fácilmente rastreables. Con esta decisión, y para tener una comparación más robusta, se seleccionaron otros siete métodos, que, si bien no muestran la misma transparencia de un clasificador de árboles, se basan en estrategias similares, permitiendo incluso ejecutar algunas herramientas descriptivas como los valores de Shapley o LIME.

La implementación de los algoritmos se realizó usando la suite de ML de Scikit-Learn en Python [25] utilizando los hiperparámetros predeterminados. Se dividió el conjunto de datos completo de forma reservada utilizando el 70 % para entrenamiento y el 30 % para pruebas. Se usó la técnica de Cross Validation con un número de splits de 30, colocando aleatoriamente cada registro en la partición de entrenamiento o de prueba.

4.4. Evaluación de los algoritmos de aprendizaje de máquina

En la Figura 7, se muestra un diagrama de caja que representan la exactitud de clasificación promedio de los ocho modelos de ML analizados. Es posible observar como el mejor modelo es el creado por el algoritmo Random Forest, sorprendentemente dejando un poco atrás a XGBoost (principal competidor para datos tabulares).

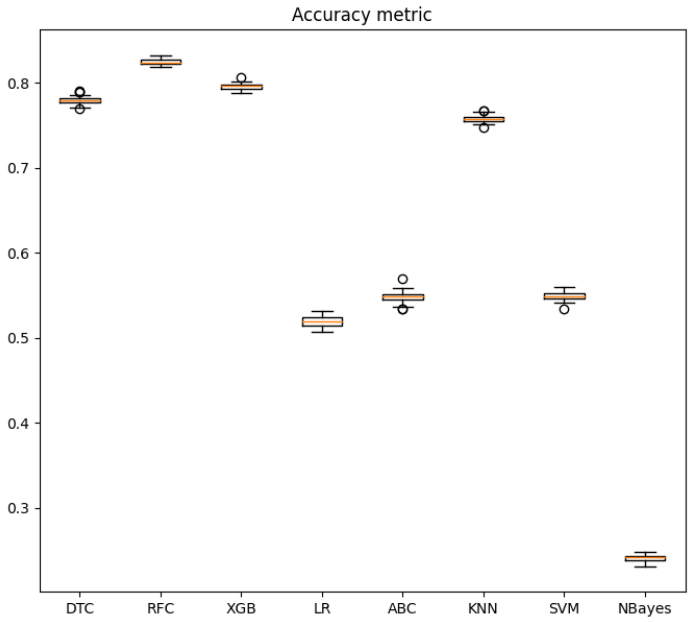

Figura 7 Rendimiento de los modelos basado en la métrica de exactitud de clasificación (Accuracy)

Dado que la distribución de las clases muestra cierto desequilibrio (ver Figura 4), teniendo más registros en la primera clase (Facturas pagadas a tiempo) y menos en la última, el Accuracy podría ser una métrica engañosa para evaluar objetivamente estos métodos. Para hacer frente a este problema, se calculó el F1 macro, que considera tanto la precisión (*Precision*) como la recuperación (*Recall*), lo que ofrece una perspectiva más sólida del rendimiento de los métodos. Para ello se presenta este resultado en la Tabla 5.

Tabla 5 Comparación de los modelos basado en la métrica F1 macro

Model	Mean (%)	Best performance (%)	Standard deviation
Decision Tree Classifier	73.31	74.63	0.00697
Random Forest	78.53	79.33	0.00487
XGB Classifier	75.03	76.44	0.00558
Logistic Regression	20.51	24.58	0.02114
AdaBoost Classifier	35.77	40.63	0.0146
KNeighbors Classifier	69.82	71.13	0.00509

SVC	25.35	26.46	0.0053
Multinomial Naive Bayes	18.86	19.6	0.00511

Es posible observar que, aunque el rendimiento del modelo disminuyó ligeramente, la conclusión sobre la mejor estrategia a utilizar es consistente con el cálculo de exactitud de clasificación observado anteriormente.

Aunque no es posible una comparación directa con la literatura, dada la diferencia en los conjuntos de datos, es informativo contrastar nuestros mejores resultados con otros trabajos informados. Si consideramos el trabajo de [3], donde reportan un Accuracy score de **0.81** obtenido por XGBoost, el cual se encuentra ligeramente por detrás del reportado en este estudio por Random Forest, el cual obtuvo un Accuracy score alrededor de **0.82**.

Para complementar el análisis de los modelos utilizados se muestra la matriz de confusión de los tres modelos con mejores resultados.

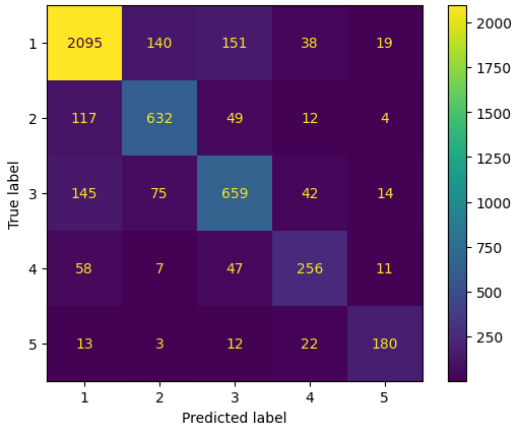

Figura 8 Matriz de confusión del modelo Decision Tree

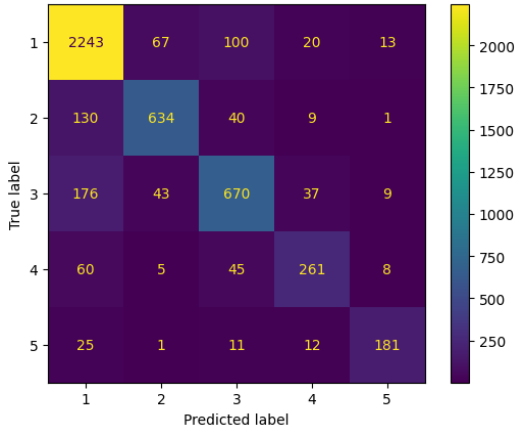

Figura 9 Matriz de confusión del modelo Random Forest

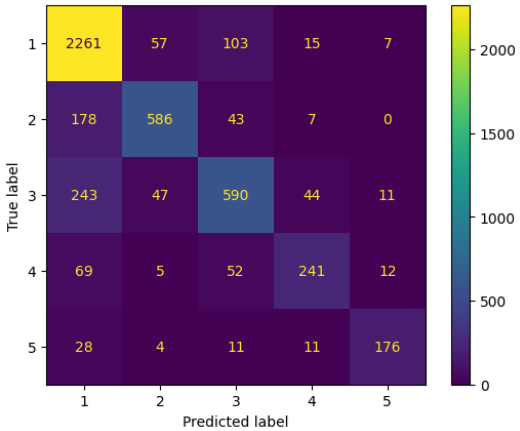

Figura 10 Matriz de confusión del modelo XGBoost Classifier

Teniendo en cuenta estas métricas se decide seleccionar el algoritmo **Random Forest** para llevar a producción.

Para ofrecer una idea del papel que desempeñan las características de entrada en el proceso de clasificación, se calcularon los valores de Shapley. A través de la escala de colores es posible observar la contribución de cada entrada del modelo para la decisión del clasificador. En este caso se presenta este análisis en la Figura 11 para el modelo antes seleccionado.

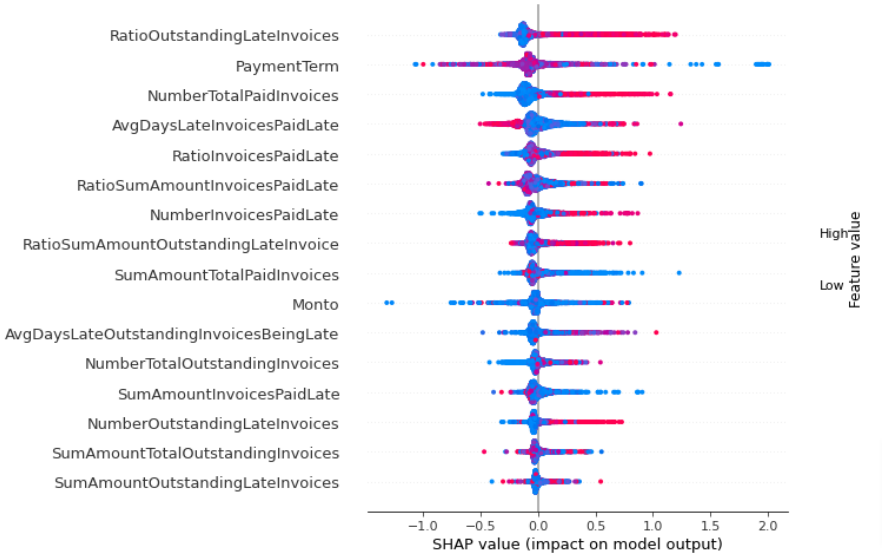

Figura 11 Valores SHAP relacionados con el Random Forests Classifier

Un aspecto a destacar de este análisis es cómo las variables Ratio de facturas pendientes de pago que se atrasaron, Plazo de pago y Número de facturas totales pagadas son más importantes para el modelo, mientras que las variables Suma del importe base del total de facturas pendientes de pago y Suma del importe base de las facturas pendientes que se retrasaron son las menos importantes. Se puede observar además como la variable ratio de facturas atrasadas pendientes (RatioOutstandingLateInvoices) cuando sus valores son altos existe una mayor probabilidad de que esa factura se pague tarde. También es posible distinguir que la variable término de pago (PaymentTerm) tiene un rango de impacto en el modelo más amplio y por eso se encuentra entre las más importantes. Al apreciar el comportamiento de la variable número de facturas pagadas tardes (NumberInvoicesPaidLate) se puede comprobar cómo mientras mayor sea este número contribuye de manera positiva a que la factura en cuestión también se pague tarde lo cual tiene bastante sentido. Este análisis permitirá refinar el modelo en próximas rondas de evaluaciones.

4.5. Despliegue

Para la concepción de la solución se tuvieron en cuenta principalmente dos arquitecturas, las cuales analizaremos a continuación.

4.5.1. Arquitectura de Machine Learning en la nube de AWS.

Inicialmente por requerimiento del cliente, se seleccionó la infraestructura en la nube de Amazon Web Service (AWS) para alojar el proyecto. Esta plataforma ofrece una amplia gama de diferentes productos globales basados en la nube para fines comerciales, incluidos los relacionados con la inteligencia artificial. Se usó Amazon SageMaker, una herramienta que permite construir, entrenar e implementar modelos de Machine Learning a cualquier escala. Para su desarrollo se siguieron las mejores prácticas de Machine Learning Well-Architected [23] diseñadas por el equipo de AWS.

En la figura Figura 12 se puede observar su diseño, cómo interactúan los actores con cada artefacto y la relación entre ellos.

1. Los datos se cargan en un bucket de **Amazon S3** para usarlos para entrenamiento y pruebas.
2. **Amazon SageMaker** proporciona instancias de Jupyter Notebook para que los científicos de datos preparen los datos y entrenen los modelos.
3. Todos los Notebooks y artefactos generados se guardan en un repositorio Git.
4. Al seleccionar el algoritmo deseado, se entrena en **Amazon SageMaker**.
5. Una vez que el modelo está listo, los artefactos generados se guardan en un bucket de **Amazon S3** y el modelo se despliega a producción mediante un endpoint al modelo de **Amazon SageMaker**.
6. El usuario ingresa la información relacionada con la factura que desea predecir en la App y espera por la respuesta.
7. La aplicación interactúa con el endpoint de **Amazon API Gateway** a través del método POST.
8. Los datos se envían a una función de **Amazon Lambda** para su transformación.

9. **Amazon Lambda** recibe los datos que se utilizarán para las inferencias y los transforma en un formato adecuado. Invoca el endpoint del modelo y recibe la predicción. Este resultado se propaga de vuelta al usuario, donde se presenta en un formato legible.

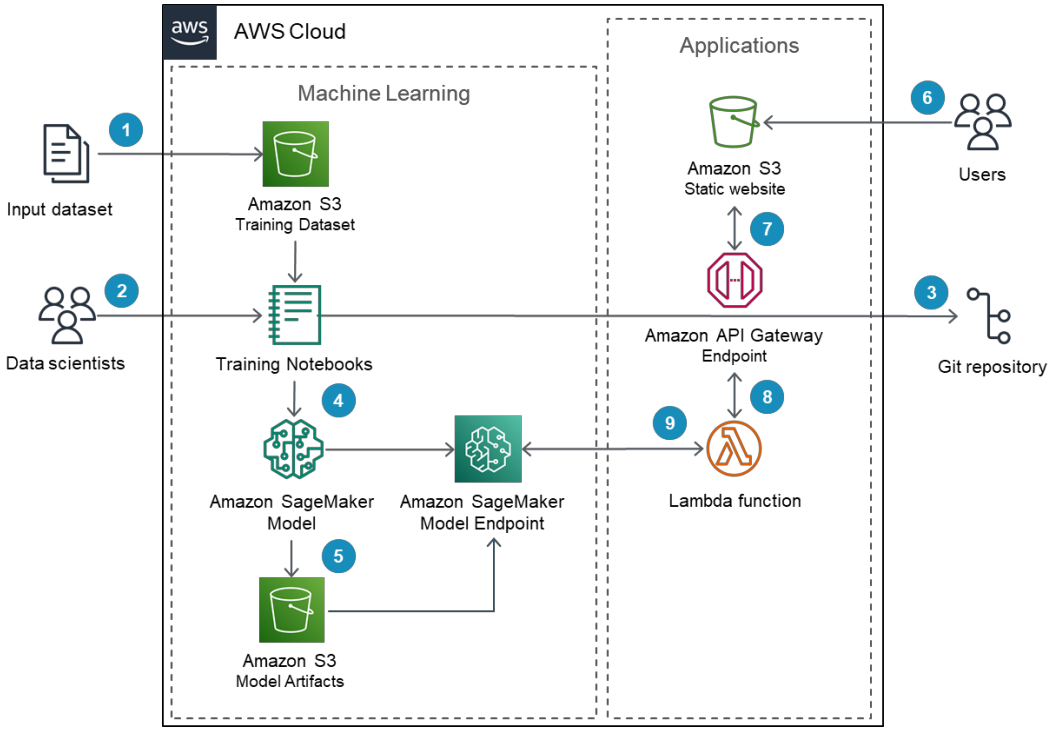

Figura 12 Diagrama arquitectónico usando la infraestructura de AWS

4.5.2. Arquitectura híbrida para el desarrollo de modelos de Machine Learning

Una alternativa frente a una arquitectura donde todos los recursos se encuentren en la nube puede ser una híbrida. Para el desarrollo de este tipo de solución, representado en la Figura 13, se aprovechó la infraestructura en perteneciente a Google Cloud, donde haciendo uso de la capa gratuita que proveen fue posible la construcción y entrenamiento del modelo. Esto nos permitió hacer uso del poder de cómputo que ofrecen las plataformas en la nube para obtener un modelo de ML robusto a la vez que desplegábamos dicho modelo en un servidor de la empresa. Para su desarrollo se siguieron las mejores prácticas de Machine Learning Well-Architected [26].

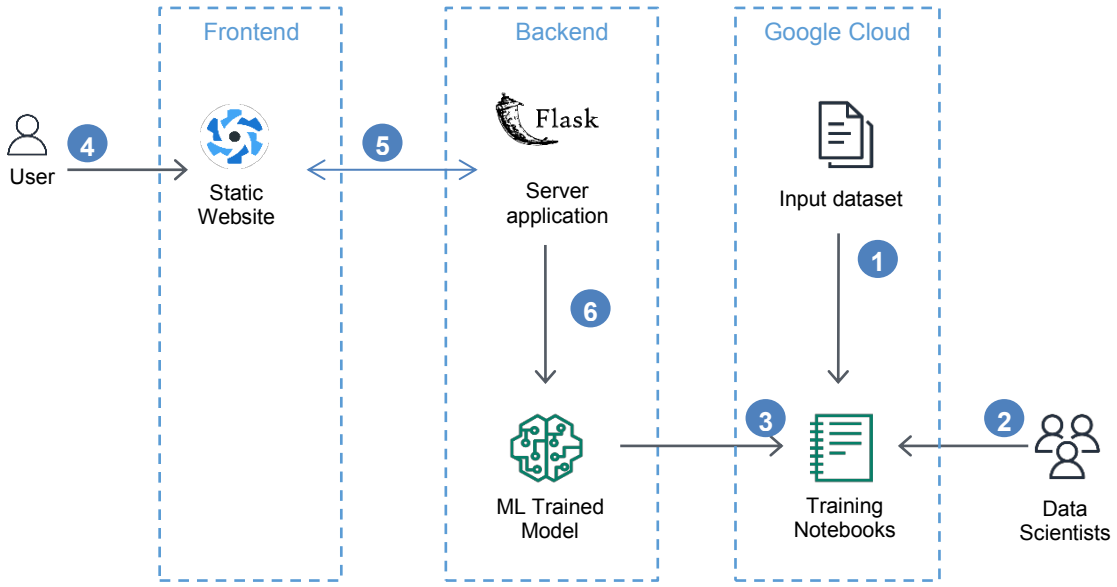

Figura 13 Diagrama arquitectónico

Descripción de cada paso:

1. Los datos se cargan en **Google drive** para ser usados en el entrenamiento y prueba de los modelos.
2. **Google Colaboratory** proporciona instancias de Jupyter Notebook para que los científicos de datos preparen los datos y entrenen los modelos.
3. Una vez que el modelo está listo, los artefactos generados se guardan en **Google Drive** y el modelo se compila en formato joblib y se despliega al servidor de producción propuesto por Blú Capital.
4. El usuario ingresa la información relacionada con la factura que desea predecir en la App y espera por la respuesta.
5. La aplicación envía la información al servidor web vía un HTTP usando el método POST.
6. El servidor recibe los datos y transforma la petición a un formato que se entendible para el modelo. Se invoca al modelo y se recibe el resultado de la

predicción. Esta respuesta se envía al usuario donde es presentada en un formato legible.

Finalmente, ante estas dos alternativas se optó por la segunda, una alternativa híbrida. Si bien la plataforma de AWS permite tener todo integrado en un solo lugar, provee un sinfín de herramientas que facilitan el desarrollo de modelos de ML, al mismo tiempo que ofrece características superiores, desde escalabilidad hasta seguridad, esta decisión se basó principalmente en los costos en que se incurría cuando se intentaban hacer operaciones que requerían un poder de cómputo superior.

Dando seguimiento a las buenas prácticas para el desarrollo de software, todo el código fuente de la aplicación se encuentra versionado usando Git [27] como sistema de control de versiones. El mismo está alojado en los repositorios propiedad de la empresa Blú Capital. Los notebooks empleados para la preparación de los datos, entrenamiento y evaluación de los modelos también se versionaron en dichos repositorios.

Como resultado final, en la Figura 14 se muestra la interfaz de usuario en forma de formulario. Esta es capaz de recolectar toda la información en un formato legible para ser enviada al modelo y esperar por la respuesta de la predicción. También se incluye una sección dónde haciendo uso de la técnica de LIME una manera para interpretar cómo el modelo tomó la decisión de clasificación. Esto puede ser muy útil para el cliente final pues apoya el proceso de toma de decisión.

Descripción de la interfaz de usuario:

1. Área donde el resultado de la predicción es presentado al usuario. Esta respuesta es convertida en un formato legible para las personas.
2. Área de formulario. La información requerida para construir los datos de entrada para el modelo es ingresada en esta sección.
3. Área explicativa. Muestra como el modelo justifica su decisión. Los gráficos son generados usando LIME.

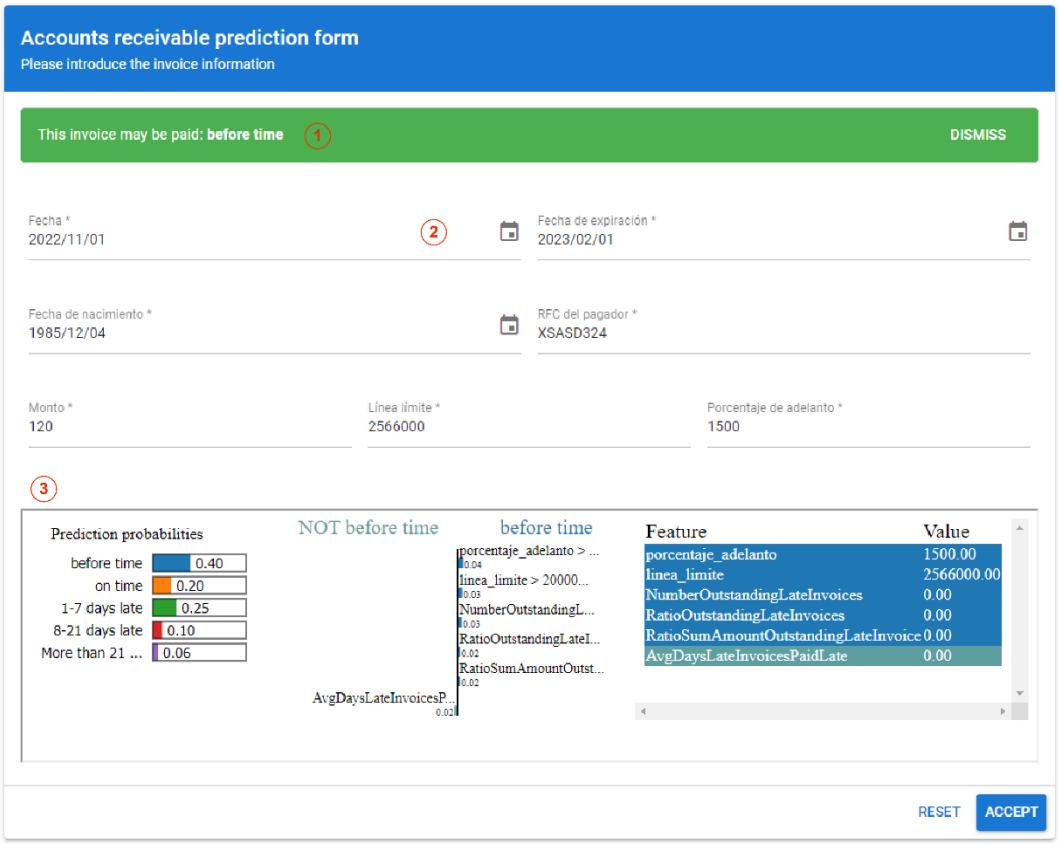

Figura 14 Interfaz de usuario para la captura de datos

Capítulo 5: Conclusiones

Durante la realización de este trabajo se realizó un estudio del estado del arte sobre diversos enfoques para abordar la predicción de cuentas por cobrar haciendo uso de técnicas de ML. Si bien es un tema que ya ha sido abordado en diferentes investigaciones el tratamiento de este problema siempre lleva un estudio personalizado teniendo en cuenta la naturaleza de los datos sobre los cuales se van a trabajar. Otro factor importante a considerar es como abordar el problema para clientes nuevos de los cuales no se tiene información previa.

Teniendo en cuenta toda la información proveniente de la bibliografía, donde la gran mayoría coincidía con el poder predictivo de los atributos históricas de la factura. Se creó un conjunto de características agregadas para capturar el comportamiento de pago histórico de cada cliente.

Unos de los problemas encontrados durante la etapa de preparación de los datos fue que en los atributos de *monto de la factura* y *monto límite de la línea* que los mismos se encontraban en diferentes monedas. Este es un problema que no fue descrito en la literatura estudiada y al cual se le da solución en este trabajo.

A través de un enfoque de aprendizaje supervisado usando modelos de clasificación multiclase se abordó el problema de previsión de AR. Se probaron ocho algoritmos de ML para pronosticar las AR y clasificarlas por categoría. Se analizaron los resultados obtenidos a través de estos modelos para diferentes datos de entrenamiento y prueba usando como métricas el *accuracy* y *F1-score*. De este análisis, se concluyó que el Random Forest Classifier se desempeña mejor que el resto de los modelos evaluados. Por lo que se cuenta con un modelo de ML robusto el cual alcanza niveles de exactitud (0.82) comparables con los de la literatura estudiada. Este modelo es capaz de cubrir los escenarios de predicción tanto para clientes recurrentes como para clientes nuevos, alcanzando mejores resultados con clientes ya existentes, pues su principal fortaleza reside en el valor de los datos históricos de la factura.

Con el objetivo de mejorar el desempeño del modelo se agregó una capa de interpretabilidad. En ella se hallaron los valores Shapley, los cuales ayudaron a entender la contribución de cada variable a la predicción del modelo. Además, se hizo uso de LIME para permitir que las predicciones locales pudieran ser interpretables y presentadas de manera visual al usuario.

Como parte de la solución se construyó además una aplicación web, la cual permite a los funcionarios de Blú Capital la entrada de información de las facturas. Dicha información es procesada por el sistema para poder realizar la predicción haciendo uso del modelo previamente entrenado. Dotando al cliente de una herramienta que lo ayudará a tomar acciones preventivas para poder identificar a clientes morosos y de esta forma disminuir el impacto negativo que pudiera causar el atraso en el pago de factura.

Por lo anteriormente expresado se considera que se da respuesta a cada una de las preguntas de la investigación. Cumpliéndose de esta manera con el objetivo general del trabajo.

Como trabajo futuro, se propone el desarrollo de un modelo más robusto que nos permita predecir con mejores resultados facturas de nuevos clientes. Aplicar métodos de ajuste de hiperparámetros para lograr un mejor resultado de clasificación. Además de integrar esta herramienta en el ecosistema de software de Blú Capital.

Referencias

[1] G. A. Bunich, Y. A. Rovenskiy, and L. P. Dashkov, “Factoring development: Theory and practice,” *Espacios*, vol. 39, no. 19, 2018.

[2] G. Pérez-Elizundia, J. A. Delgado-Guzmán, and J. F. Lampón, “Commercial banking as a key factor for SMEs development in Mexico through factoring: A qualitative approach,” *Eur. Res. Manag. Bus. Econ.*, vol. 26, no. 3, pp. 155–163, 2020, doi: 10.1016/j.iedeen.2020.06.001.

[3] A. P. Appel, G. L. Malfatti, R. L. de F. Cunha, B. Lima, and R. de Paula, “Predicting Account Receivables with Machine Learning,” Aug. 2020, [Online]. Available: http://arxiv.org/abs/2008.07363

[4] P. Kapadia, B. Kadhiwala, T. Bahurupi, H. Dalal, S. Jariwala, and K. Naik, “A Novel Approach for Forecasting Account Receivables,” in *Lecture Notes in Networks and Systems*, 2022, vol. 321, pp. 797–806. doi: 10.1007/978-981-16-5987-4_79.

[5] S. Zeng, P. Melville, C. A. Lang, I. Boier-Martin, and C. Murphy, “Using predictive analysis to improve invoice-to-cash collection,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, no. August, pp. 1043–1050, 2008, doi: 10.1145/1401890.1402014.

[6] R. D. Bailey *et al.*, “Providian Financial Corporation: Collections Strategy,” in *Systems Engineering Capstone Conference, University of Virginia*, 1999, pp. 23–30.

[7] H. S. Shah, “Customer Payment Prediction in Account Receivable,” *Int. J. Sci. Res. Index Copernicus Value*, vol. 8, no. 1, pp. 7–296, 2016, [Online]. Available: https://halobi.com/blog/descriptive-

[8] H. Peiguang, “Predicting and Improving Invoice-to-Cash Collection Through Machine Learning,” *Master Thesis*, pp. 1–92, 2015, [Online]. Available: http://dspace.mit.edu/bitstream/handle/1721.1/99584/925473704-MIT.pdf?sequence=1

[9] P. Chuqui and M. Cecilia, “Comparativa de técnicas Machine Learning sobre comportamiento de pago de clientes con cuentas por cobrar,” p. 91, 2019, [Online]. Available: https://reunir.unir.net/handle/123456789/9734

[10] M. Ramirez Quiceno and A. Medina Baéz, “Aplicación de técnicas de Machine Learning para la predicción del riesgo de default de un cliente en una compañía de filipinas,” *Univ. Antioquia*, pp. 1–47, 2022.

[11] J. J. Espinosa Zúñiga, “Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito,” *Ing. Investig. y Tecnol.*, vol. 21, no. 3, pp. 1–16, 2020, doi: 10.22201/fi.25940732e.2020.21.3.022.

[12] Martín Fernando Calero Pérez, “Aplicación de Machine Learning a través de la metodología CRISP-DM para la predicción de pago por acuerdo en una empresa de cobro de deudas.,” 2022.

[13] C. Schröer, F. Kruse, and J. M. Gómez, “A systematic literature review on applying

CRISP-DM process model," *Procedia Comput. Sci.*, vol. 181, no. 2019, pp. 526–534, 2021, doi: 10.1016/j.procs.2021.01.199.

[14] P. Dönmez, "Introduction to Machine Learning, 2nd ed., by Ethem Alpaydın. Cambridge, MA: The MIT Press2010. ISBN: 978-0-262-01243-0. $54/£ 39.95 + 584 pages.," *Nat. Lang. Eng.*, vol. 19, no. 2, pp. 285–288, 2013, doi: 10.1017/s1351324912000290.

[15] J. Bobadilla, "Machine Learning y Deep Learning: Usando Python, Scikit y Keras," p. 322, 2020, [Online]. Available: https://elibro.net/es/lc/uguayaquil/titulos/222698

[16] Ethem Alpaydin, *Machine Learning New AI*, vol. 91. 2017. [Online]. Available: https://www.wook.pt/livro/machine-learning-ethem-alpaydin/17932580

[17] S. Malik, R. Harode, and A. Singh Kunwar, "XGBoost: A Deep Dive into Boosting," Burnaby, BC, Canada., 2022. doi: 10.13140/RG.2.2.15243.64803.

[18] V. K. Vemuri, "The Hundred-Page Machine Learning Book," *J. Inf. Technol. Case Appl. Res.*, vol. 22, no. 2, pp. 136–138, 2020, doi: 10.1080/15228053.2020.1766224.

[19] R. Wang, "AdaBoost for Feature Selection, Classification and Its Relation with SVM, A Review," *Phys. Procedia*, vol. 25, pp. 800–807, 2012, doi: 10.1016/j.phpro.2012.03.160.

[20] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Section 2, pp. 4766–4775, 2017.

[21] J. Yang, "Fast TreeSHAP: Accelerating SHAP Value Computation for Trees," 2021, [Online]. Available: http://arxiv.org/abs/2109.09847

[22] S. Chen, "Interpretation of multi-label classification models using shapley values," pp. 1–12, 2021, [Online]. Available: http://arxiv.org/abs/2104.10505

[23] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?' Explaining the Predictions of Any Classifier," *NAACL-HLT 2016 - 2016 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Proc. Demonstr. Sess.*, pp. 97–101, 2016, doi: 10.18653/v1/n16-3020.

[24] L. Metcalf and W. Casey, "Introduction to data analysis," *Cybersecurity Appl. Math.*, pp. 43–65, Jan. 2016, doi: 10.1016/B978-0-12-804452-0.00004-X.

[25] F. G. V. Pedregosa, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 127, no. 9, pp. 2825–2830, 2019, doi: 10.1289/EHP4713.

[26] Amazon, "Well-Architected machine learning - Machine Learning Lens." https://docs.aws.amazon.com/wellarchitected/latest/machine-learning-lens/well-architected-machine-learning.html (accessed Jan. 31, 2023).

[27] U. S. Smart, "Manual de usuario de git," vol. 1, 2014.