TITULO PROYECTO:

Evaluación de rentabilidad y riesgos en financiamiento de Bills Médicos

INTEGRANTES:

- **Cristian Usme Córdoba**
- **María Alejandra Vargas Duque**

1. ENTENDIMIENTO DEL NEGOCIO

1.1 DESCRIPCIÓN DEL NEGOCIO

Puma Management Group es una empresa que ofrece servicios financieros y operativos especializados para proveedores de salud, con un enfoque particular en la gestión de *liens* (reclamaciones y obligaciones asociadas a servicios médicos). Su propuesta se basa en generar valor a través de un acompañamiento cercano al cliente, brindando soporte bilingüe, soluciones tecnológicas propias como su sistema **Puma TRAX**, y una filosofía centrada en relaciones de confianza y mejora continua para optimizar los procesos médicos y financieros de sus aliados.

1.2 DESCRIPCIÓN DEL PROBLEMA:

Puma Management Group enfrenta una incertidumbre significativa en la toma de decisiones para la adquisición de facturas médicas (bills). Actualmente, la empresa carece de criterios objetivos y sistemáticos que permitan evaluar de manera anticipada la rentabilidad potencial de cada transacción. Su modelo de negocio, que consiste en financiar bills asociados a casos legales de accidentes, implica que la recuperación de la inversión y la rentabilidad adicional dependen directamente del abogado, del proveedor médico, de los valores de los bills y del resultado final del caso. Esto genera un entorno de alta variabilidad y riesgo, ya que no existen parámetros claros que guíen la aceptación o el rechazo de nuevas oportunidades de inversión, lo que se manifiesta en múltiples deficiencias operativas y estratégicas.

Dentro de este problema general se identifican varias problemáticas específicas:

1. No existen criterios cuantitativos que permitan determinar la rentabilidad potencial de un bill antes de su adquisición.
2. La empresa desconoce qué combinaciones de proveedor y abogado tienden a generar mayores pérdidas o menores retornos.
3. Las decisiones de compra se realizan sin una priorización clara entre bills de alta y baja rentabilidad.
4. No hay una estimación previa del retorno esperado sobre la inversión de cada bill.

5. Se carece de indicadores objetivos para evaluar el desempeño histórico de abogados y proveedores.
6. Las decisiones erróneas en la adquisición de bills ocasionan pérdidas económicas y afectan la estabilidad financiera del negocio.
7. No se cuenta con un sistema que permita anticipar el nivel de riesgo asociado a cada transacción.

1.3 OBJETIVOS DE LA MINERÍA:

- Desarrollar un modelo predictivo que permita estimar la rentabilidad potencial de un bill antes de su adquisición, utilizando como variables explicativas el proveedor, el abogado, el valor del bill y el valor de la compra.
- Construir un modelo predictivo que identifique combinaciones de proveedor y abogado asociadas con un mayor riesgo de pérdida, a partir de patrones históricos de rentabilidad y desempeño financiero.
- Diseñar un modelo predictivo que clasifique los bills según su nivel de rentabilidad esperada, permitiendo establecer una priorización en la toma de decisiones de compra e inversión.
- Implementar un modelo predictivo que estime el retorno esperado sobre la inversión de cada bill, contribuyendo a proyectar el impacto financiero antes de concretar la adquisición.
- Desarrollar un modelo descriptivo-predictivo que evalúe el desempeño histórico de abogados y proveedores en términos de rentabilidad, generando indicadores objetivos para la gestión de relaciones estratégicas.
- Elaborar un modelo predictivo orientado a minimizar la probabilidad de decisiones erróneas en la compra de bills, reduciendo el impacto de pérdidas económicas en la estabilidad del negocio.

- Construir un modelo predictivo capaz de anticipar el nivel de riesgo asociado a cada transacción, clasificando los bills según su probabilidad de pérdida o ganancia.

1.4 DISEÑO DE SOLUCIÓN

Problema	**Tipo de Minería**	**Tipo de Aprendizaje**	**Requerimiento de Datos**	**Métodos**	**Evaluación**
No existen criterios cuantitativos que permitan determinar la rentabilidad potencial de un bill antes de su adquisición.	Minería predictiva	Supervisado	Datos históricos de bills con atributos de proveedor, abogado, valor del bill, valor de la compra y rentabilidad final (ROI o etiqueta rentable/pérdida).	Regresión logística, Árboles de decisión, Random Forest, SVM, Redes Neuronales, KNN, AdaBoost, Gradient Boosting.	Accuracy, F1-score, Matriz de confusión, ROC-AUC.
La empresa desconoce qué combinaciones de proveedor y abogado tienden a generar mayores pérdidas o menores retornos.	Minería predictiva y descriptiva	Supervisado	Dataset con registros históricos que contengan identificadores de proveedor y abogado, junto con la etiqueta de rentabilidad.	Árboles de decisión, Random Forest, Análisis de importancia de variables.	Precisión en la clasificación y análisis de importancia de características.
Las decisiones de compra se realizan sin una priorización clara entre	Minería predictiva	Supervisado	Datos con variables financieras y categóricas que permitan clasificar	Modelos de clasificación (Gradient Boosting, Random	Métricas de ranking (Precision@k), F1-score, ROC-AUC.

bills de alta y baja rentabilidad.			bills según retorno histórico.	Forest).	
No hay una estimación previa del retorno esperado sobre la inversión de cada bill.	Minería predictiva (Regresión)	Supervisado	Dataset con valores de inversión, pagos y retornos reales.	Regresión lineal, Random Forest Regressor, XGBoost Regressor.	RMSE, MAE, R^2.
Se carece de indicadores objetivos para evaluar el desempeño histórico de abogados y proveedores.	Minería descriptiva	No supervisado	Datos históricos agrupados por abogado y proveedor, con métricas de rentabilidad asociadas.	Análisis de clustering (K-Means) y estadística descriptiva.	Silhouette Score, análisis de varianza interna.
Las decisiones erróneas en la adquisición de bills ocasionan pérdidas económicas.	Minería predictiva	Supervisado	Datos históricos con etiquetas de rentabilidad (rentable/pérdida).	Árboles de decisión, AdaBoost, Redes Neuronales.	Accuracy, F1-score, matriz de confusión.

- Evaluación esperada:
  Actualmente tienen un porcentaje de perdidas que va desde el 25 al 30%.

1.5 RECURSOS PARA CREACIÓN DEL MODELO Y PARA DESPLIEGUE

- Hardware: Se entrega un modelo a través de Google Colaboratory y se despliega en un servidor local no dedicado (Ej: Un computador personal).
- Software: Se realizará un modelo a través de Google Colaboratory y su despliegue se puede realizar con Google Cloud Run y Docker.

2. ENTENDIMIENTO DE LOS DATOS

2.1 CICLO DE LOS DATOS: Generación, Almacenamiento, Modificación (ruta), Periodicidad

1. Generación de los datos

Los datos se generan principalmente cuando los agentes de operaciones ingresan información manual en el sistema TRAX, a partir de notificaciones o actualizaciones que llegan desde los abogados sobre los estados de los casos.

El contexto de generación es el seguimiento legal y operativo de los casos relacionados con la compra de bills.

Cada nuevo avance en un caso, documento recibido o comunicación con los abogados constituye un punto de generación de datos.

2. Almacenamiento de los datos

- Toda la información ingresada se registra en el sistema de manejo de casos TRAX, perteneciente a Puma Management.
- Estos datos se almacenan en una base de datos en la nube, lo que permite que estén disponibles de forma centralizada y accesible para los diferentes actores autorizados.

3. Modificación y ruta de los datos

Los 3 agentes de operaciones son quienes actualizan y modifican los datos en TRAX, siguiendo la ruta:

Abogados → Notificación/avance → Agente de operaciones → Registro en TRAX → Base de datos en la nube.

- El riesgo de errores aumenta en dos puntos clave:
- Ingreso manual de datos por parte de los agentes.
- Interpretación de la información recibida desde los abogados.
- La trazabilidad depende de cómo cada agente registra correctamente los cambios en el sistema.

4. Periodicidad de los datos:

La periodicidad no es fija, sino que depende de cuándo llegan las notificaciones de los abogados.

Sin embargo, de manera práctica:

- Se ingresan datos diariamente porque siempre hay casos en curso.
- Cada caso tiene actualizaciones eventuales, lo que significa que no siguen un calendario exacto, sino que se registran conforme avanza el proceso legal.

2.2 DICCIONARIO DE DATOS

Variable	**Descripción**	**Tipo**
Account Name	Nombre de la cuenta del abogado	Categórico
Provider Client	Nombre del proveedor	Categórico
Subject	Tema por el que se identifica el caso	Categórico
Name of Procedure	Nombre del procedimiento realizado o solicitado	Categórico
Case Stage	Estado actual del caso	Categórico
Bill Amt	Valor estimado del bill	Numérico flotante
Purch Amt	Valor de la compra del bill a los abogados	Numérico flotante
Pay Amt	Valor pagado por los abogados a la compañía cuando finaliza un caso	Numérico flotante
Funding Status	Estado de financiamiento del bill	Categórico

1.1 REGLAS DE CALIDAD DESDE EL NEGOCIO (No salen de los datos)

Variable	Quality rule

Bill Amt	Rango de [0.1,inf]
Purch Amt	Rango de [0.1,inf]
Subject	['Lien', 'Imaging', 'Pain Management'", "'Physical Therapy','Chiropractic', 'Medical Device', 'Administrative', 'Surgery']
Account Name	['Ace Law Group', 'Angulo Law Group', 'Atkinson Watkins & Hoffman Attorneys', 'BD & J Law Firm', 'Benjamin Nadig Law', 'Benson Allred Injury Law', 'Blackburn Wirth Injury Team', 'Cardenas Law Group', 'David W Fassett Personal Injury Law', 'Dimopoulos Injury Law', 'ER Injury Attorneys', 'Fuller Law Practice', 'G Dallas Horton & Associates', 'Goldberg Injury Law', 'Jacoby & Meyers CA', 'Ladah Law', 'Lalezary Law Firm CA Law Brothers', 'Law Office of Arash Khorsandi', 'Law Office of Stephen Reid', 'Law Office of Victor M Cardoza', 'Lawrence C Hill & Associates', 'Lloyd Baker Injury Attorneys', 'Mullins &

	Trenchak Law', 'Muslusky Law', 'Naqvi Injury Law', 'Neal Hyman Law', 'Nwogbe Law Group', 'Parke Law Firm', 'RRCC Firm', 'Sidell Injury Law', 'Valarie Fujii Law Offices', 'Vannah & Vannah Law Firm', 'VC2 Law', 'Willoughby Shulman Injury Lawyers', 'Wilshire Law Firm CA', 'Yan Kenyon Law']
Provider Name	['Pueblo Medical Imaging', 'LVR', 'Durango Surgery Center', 'Parkway Surgery Center', 'Suarez Physical Therapy', 'Jackson Physical Therapy', 'Simpson Chiropractic', 'Precision Medical Products', 'OpenSided MRI of Las Vegas', 'Surgical Arts Centre', 'LVC Surgery Center']

2. DATA PREPARATION

2.1 INTEGRATION

En esta fase se consolidó el conjunto de datos base para el análisis. Dado que el dataset provenía de una única fuente (Portfolio Puma MGM.xlsx), no fue necesario realizar procesos de integración entre múltiples tablas o fuentes.

El dataset fue cargado utilizando la librería pandas, lo que permitió inspec-

cionar su estructura inicial con df.info() y df.head().

```
[76] ✓ 1s
df = pd.read_excel('Portfolio Puma MGM.xlsx', sheet_name=0)
df.head(2)
```

	Case Name	Account Name	State	Rate	Provider Client	DOL	Subject	Name of Procedure	Case Stage	Stage Chg Dt	...	Inv 1 Pd Dt	Inv 1 Pd $	I
0	Sundae McGrath	Ace Law Group	NaN	NaN	Pueblo Medical Imaging	09/06/2018	SMcGrath1	MRI Lumbar Spine wo Contrast	Completed	01/21/2025	...	01/20/2025	907.5	N
1	Sumala Chuencharoewong	Ace Law Group	NaN	NaN	Pueblo Medical Imaging	NaN	SChuen1	MRI Cervical Spine wo Contrast	Completed	09/28/2023	...	05/27/2020	940.5	N

2 rows × 33 columns

```
[77] ✓ 0s
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4008 entries, 0 to 4007
Data columns (total 33 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Case Name           4008 non-null   object
 1   Account Name        4008 non-null   object
 2   State               39 non-null     object
 3   Rate                764 non-null    object
 4   Provider Client     3997 non-null   object
 5   DOL                 3137 non-null   object
 6   Subject             4008 non-null   object
 7   Name of Procedure   3483 non-null   object
 8   Case Stage          4008 non-null   object
 9   Stage Chg Dt        4002 non-null   object
 10  DOS                 3999 non-null   object
 11  Batch Dt            3987 non-null   object
 12  Bill Amt            4008 non-null   float64
 13  Purch Amt           4008 non-null   float64
 14  Purch Dt            3990 non-null   object
 15  Investor 1          4007 non-null   object
 16  Port 1              3976 non-null   object
 17  Inv 1 Purch $       3418 non-null   float64
 18  Investor 2          699 non-null    object
 19  Port 2              590 non-null    object
 20  Inv 2 Purch $       3372 non-null   float64
 21  Pay Dt              4008 non-null   object
 22  Pay Amt             3999 non-null   float64
 23  Inv 1 Pd Dt         3583 non-null   object
 24  Inv 1 Pd $          3786 non-null   float64
 25  Inv 2 Pd Dt         241 non-null    object
 26  Inv 2 Pd $          3376 non-null   float64
 27  Inv 3 Pd Dt         2 non-null      object
 28  Inv 3 Pd $          3359 non-null   float64
 29  Funding Status      3998 non-null   object
 30  Case Notes          172 non-null    object
 31  Lien Notes          98 non-null     object
 32  Case Status         2611 non-null   object
dtypes: float64(8), object(25)
memory usage: 1.0+ MB
```

Se garantizó la integridad de los datos eliminando registros sin fecha de pago (Pay Dt), para asegurar consistencia temporal en las observaciones.

```
# Antes de eliminar variables irrelevantes eliminamos los registros que no tienen fecha de pago
df = df.dropna(subset=['Pay Dt'])
```

2.2 VARIABLES SELECTION

Se efectuó una **reducción del conjunto de variables** eliminando aquellas que no aportaban valor analítico o que no eran relevantes para los objeti-

vos de predicción.
Entre las variables descartadas se encuentran:
['Case Name','Inv 1 Pd Dt','Inv 2 Pd Dt','Inv 3 Pd Dt','Case Notes','Lien Notes','Case Status','DOL','DOS','Investor 1','Port 1','Investor 2','Port 2','State','Inv 1 Purch $','Inv 2 Purch $','Inv 1 Pd $','Inv 2 Pd $','Inv 3 Pd $','Rate','Stage Chg Dt','Batch Dt','Pay Dt', 'Case Stage','Name of Procedure','Purch Dt']

El criterio de eliminación se basó en:

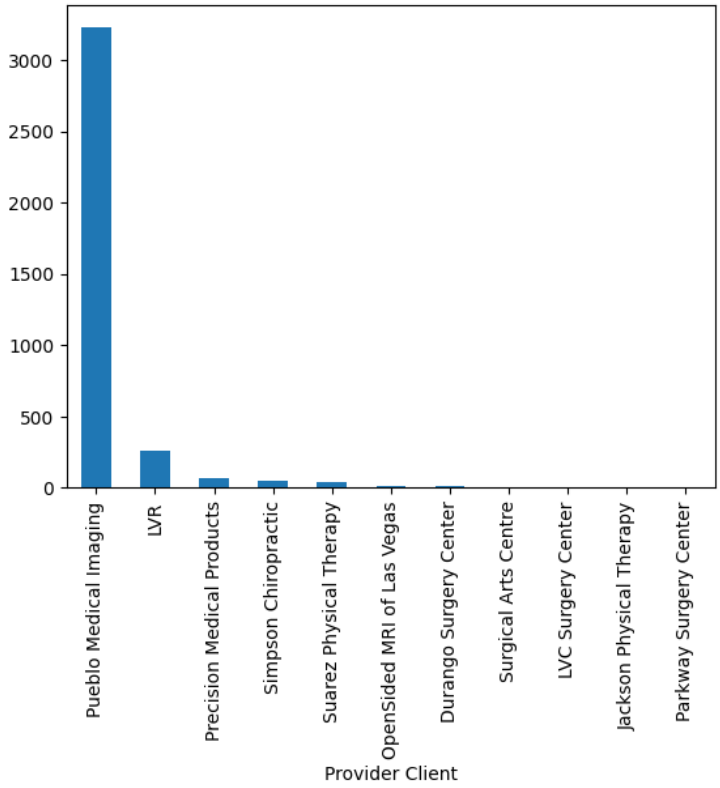
- Redundancia de información (por ejemplo, campos derivados o duplicados).
- Variables sin valor predictivo directo (identificadores, descripciones textuales).
- Campos con valores mayoritariamente nulos o sin variabilidad.

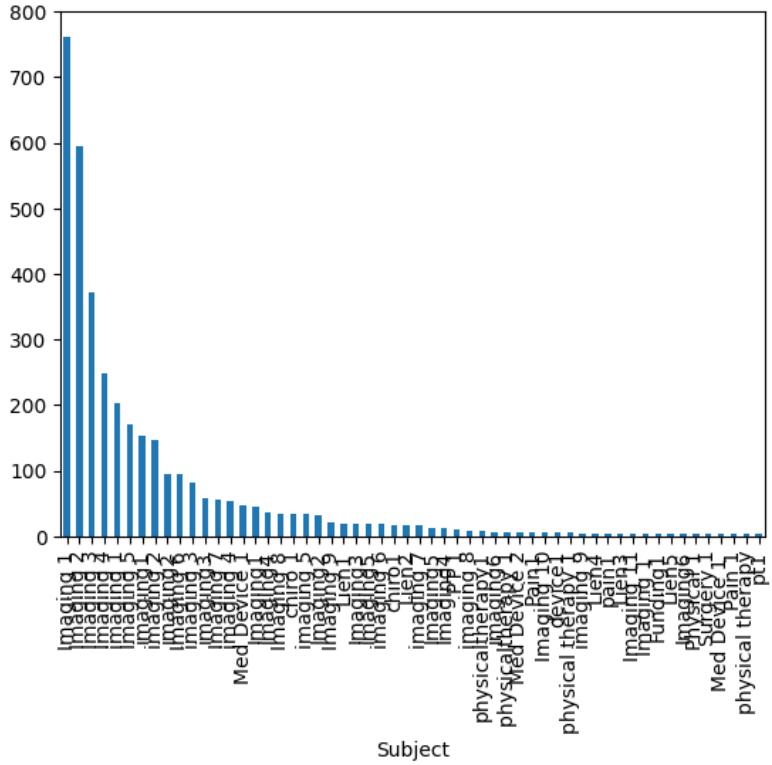
```
# Removal of irrelevant variables
irrelevant_variables = ['Case Name','Inv 1 Pd Dt','Inv 2 Pd Dt','Inv 3 Pd Dt','Case Notes','Lien Notes','Case Status','DOL','D
df = df.drop(irrelevant_variables, axis=1, errors='ignore')
df.info()
```

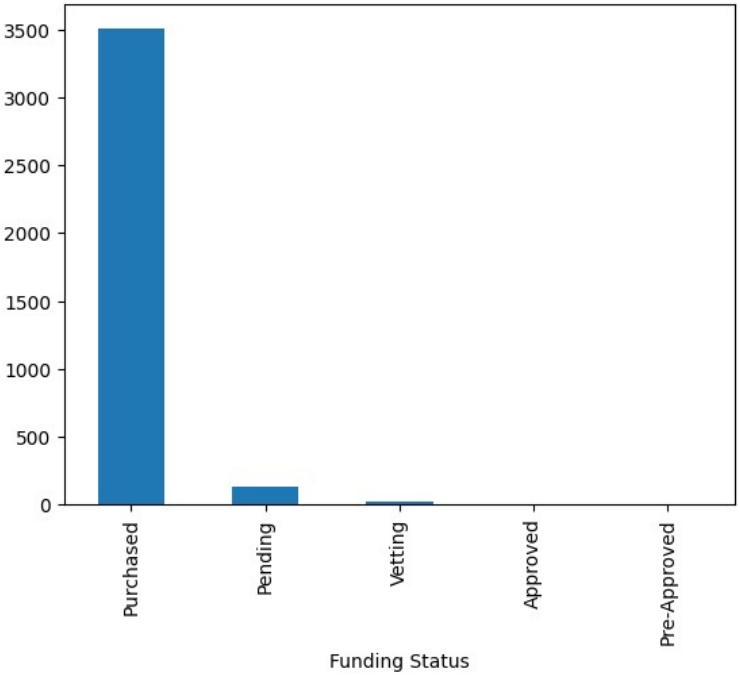
```
<class 'pandas.core.frame.DataFrame'>
Index: 1350 entries, 0 to 4004
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Account Name     1350 non-null   category
 1   Provider Client  1350 non-null   object
 2   Subject          1350 non-null   object
 3   Bill Amt         1350 non-null   float64
 4   Purch Amt        1350 non-null   float64
 5   ROI_Categoria    1350 non-null   object
dtypes: category(1), float64(2), object(3)
memory usage: 67.3+ KB
```

2.3 ESTADÍSTICA DESCRIPTIVA

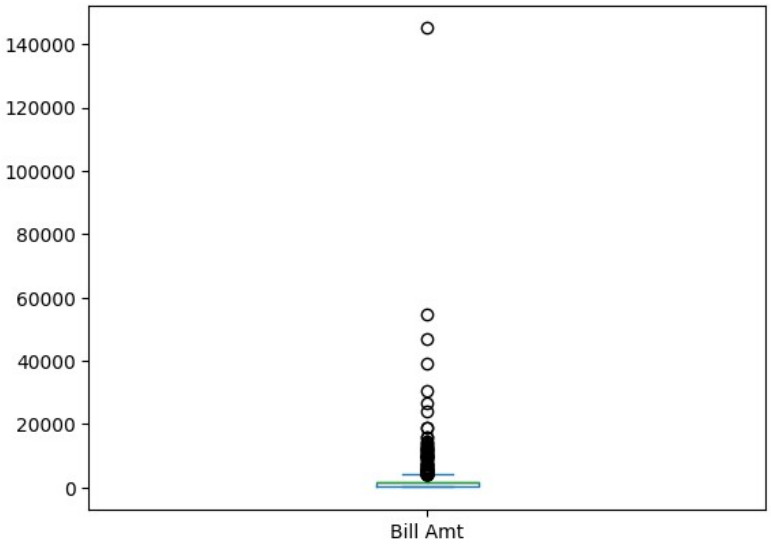
Una vez definidas las variables relevantes, se realizaron análisis descriptivos para comprender la distribución y comportamiento de los datos.
Se analizaron:
Frecuencias de variables categóricas como Provider Client, Account Name y Funding Status.

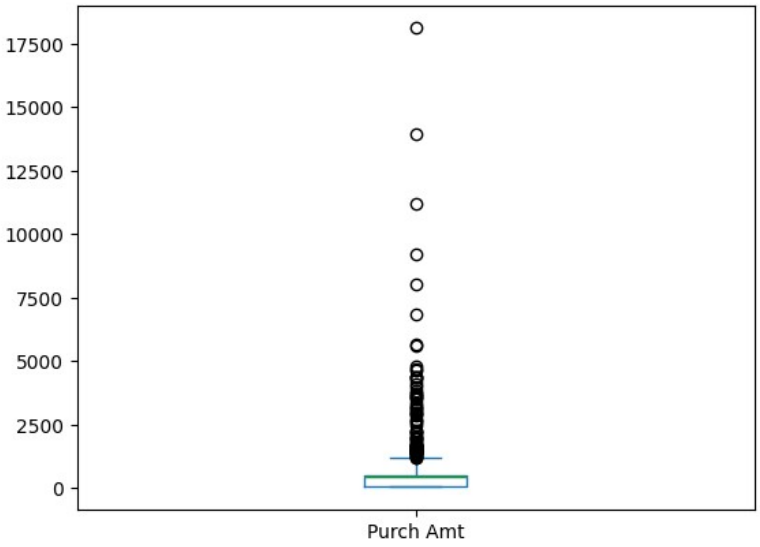

3000
2500
2000
1500
1000
500
0
Pueblo Medical Imaging
LVR
Precision Medical Products
Simpson Chiropractic
Suarez Physical Therapy
OpenSided MRI of Las Vegas
Durango Surgery Center
Surgical Arts Centre
LVC Surgery Center
Jackson Physical Therapy
Parkway Surgery Center
Provider Client

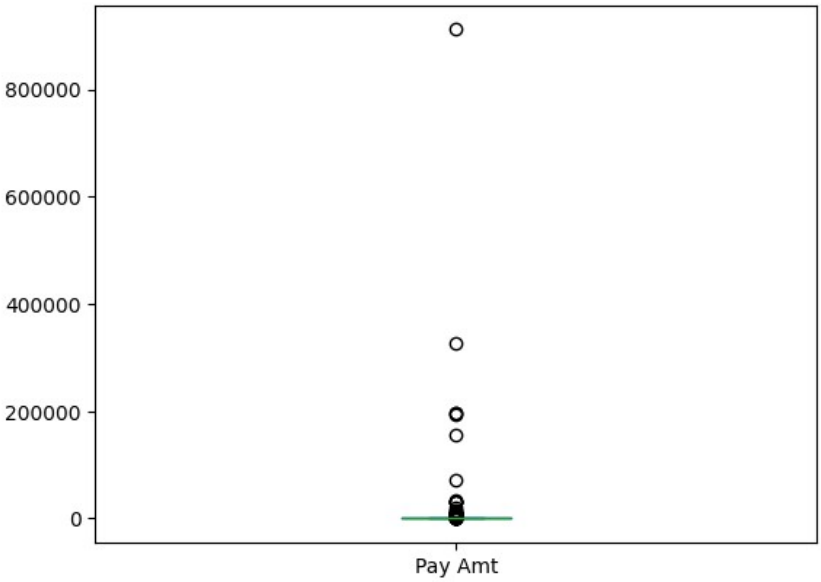

800
700
600
500
400
300
200
100
0
Subject



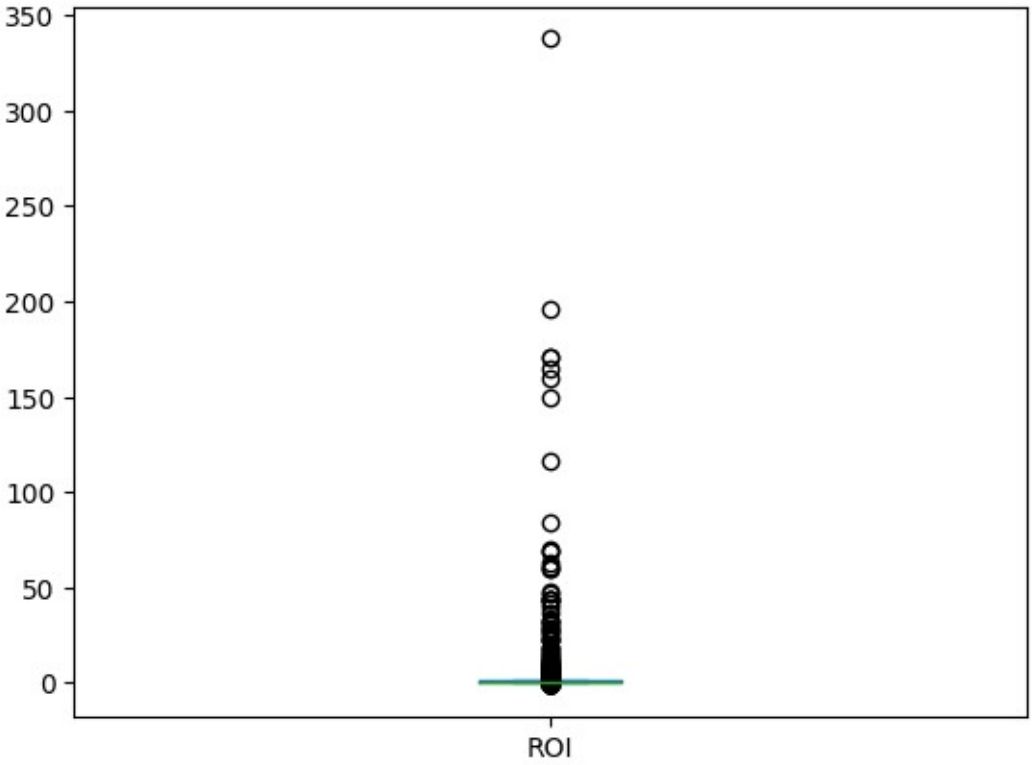
Distribuciones numéricas de montos (Bill Amt, Purch Amt, Pay Amt) mediante histogramas y diagramas de caja.



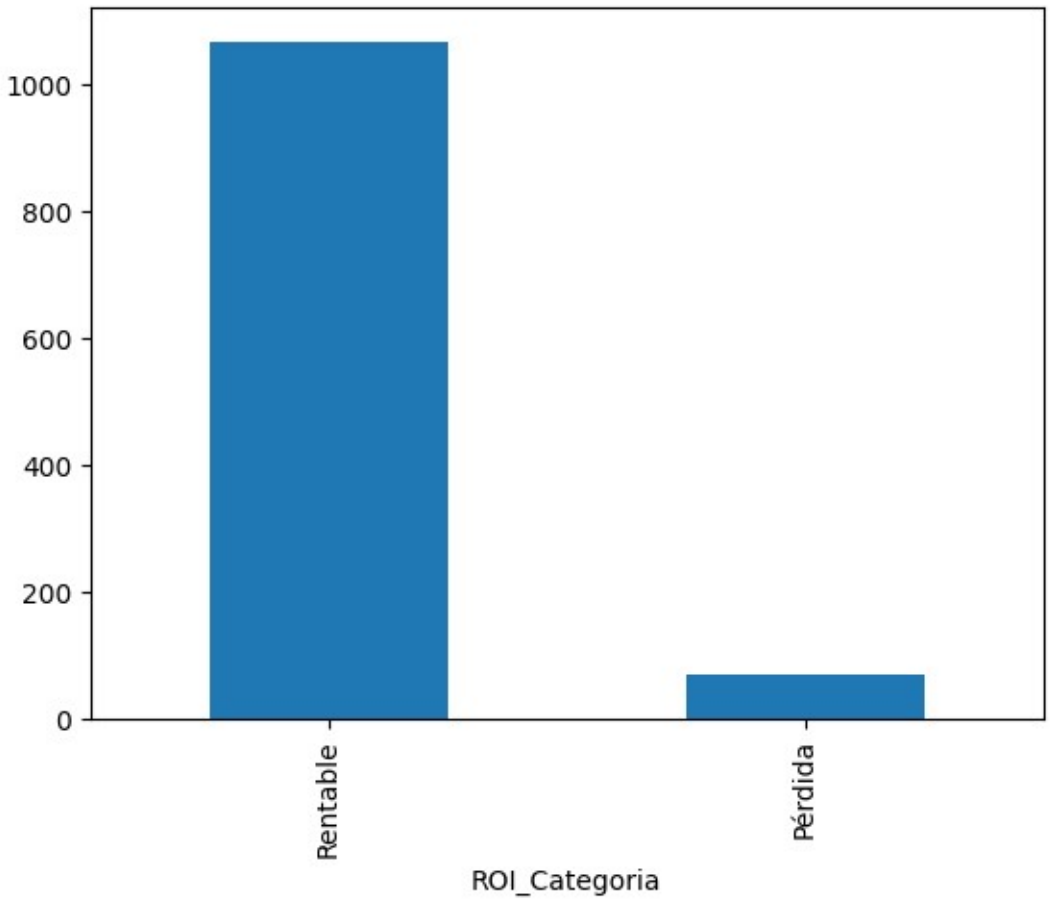




El objetivo fue identificar sesgos, dispersión y posibles valores atípicos en las variables numéricas.
En el caso específico de este dataset, tenía muchos valores atípicos pero eran valores reales que habían sucedido y habían dejado rentabilidades extremadamente altas cuando los abogados ganaron los casos por lo que para mejorar la eficacia del modelo y reducir la cantidad de atípicos se optó por la creación de una nueva variable numérica llamada ROI (Return on Investment) utilizando como fórmula df['Pay Amt'] - df['Purch Amt']) / df['Purch Amt'] (la ganancia dividido el valor de compra del bill), a pesar de haber tomado esta decisión estatégica en la calidad de los datos, seguimos obtuviendo outliers y aunque los entrenábamos con modelos de regresión para predecir el valor del ROI estimado la calidad de los modelos estaba muy por debajo.



Debido a esto se tomó la decisión de la creación de 2 variables categóricas, pérdida y rentable quedando su asignación de la siguiente manera:



2.4 LIMPIEZA DE ATÍPICOS

Posteriormente se llevó a cabo una identificación y tratamiento de outliers en las variables numéricas.

Para ello, se calcularon los rangos intercuartílicos (IQR) o se usaron visualizaciones tipo boxplot para detectar valores extremos.

Los registros con montos anómalamente altos o bajos fueron revisados y, en algunos casos, eliminados o ajustados según criterios de negocio.

```
df['Funding Status'].unique()

['Purchased', 'Pre-Approved', 'Pending', 'Approved', 'Vetting', NaN]
Categories (5, object): ['Approved', 'Pending', 'Pre-Approved', 'Purchased', 'Vetting']
```

```
# Se eliminan los registros que hayan sido declined porque no se tiene información completa de ellos y los que nos importa analizar son los comprados

indices_a_eliminar = df[df['Funding Status'] != 'Purchased'].index

# Eliminar esas filas usando drop(axis=0)
df.drop(indices_a_eliminar, axis=0, inplace=True)
```

```
df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 3667 entries, 0 to 3841
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Account Name     3667 non-null   category
 1   Provider Client  3657 non-null   category
 2   Subject          3667 non-null   category
 3   Bill Amt         3667 non-null   float64
 4   Purch Amt        3667 non-null   float64
 5   Pay Amt          3660 non-null   float64
 6   Funding Status   3667 non-null   category
dtypes: category(4), float64(3)
memory usage: 140.1 KB
```

```
# Como todos los datos pertenecen a la cateoría Purchased de Funding Status entonces eliminamos esta variable por ser redundante
df = df.drop('Funding Status', axis=1)
```

Existen procesos que a pesar de haber sido comprados aún no se han terminado porque están a la espera de que el abogado pague (Pay amt = nan) por lo que estos registros no se pueden utilizar porque se utiliza Pay amt para calcular nuestra variable objetivo

```
# Se eliminan  los registros donde Pay amt sea nula
df = df.dropna(subset=['Pay Amt'])
```

```
df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 3660 entries, 0 to 3841
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Account Name     3660 non-null   category
 1   Provider Client  3650 non-null   category
 2   Subject          3660 non-null   category
 3   Bill Amt         3660 non-null   float64
 4   Purch Amt        3660 non-null   float64
 5   Pay Amt          3660 non-null   float64
dtypes: category(3), float64(3)
memory usage: 136.1 KB
```

2.5 LIMPIEZA DE NULOS

Además del filtrado inicial de Pay Dt, se verificaron valores nulos en todas las variables mediante df.isnull().sum().

Las variables con porcentajes bajos de nulos se imputaron según su naturaleza:

Media o mediana para variables numéricas.

Moda o categoría “Desconocido” para variables categóricas.

En caso de nulos no significativos o sin posibilidad de imputación válida, se eliminaron los registros.

```python
#Limpieza de datos nulos: Imputación por la media y moda
from sklearn.impute import SimpleImputer

#Imputacion de variables numéricas: media
var_numericas = ['Bill Amt','Purch Amt']
ImpNumeros = SimpleImputer(missing_values=np.nan, strategy='mean')
df[var_numericas] = ImpNumeros.fit_transform(df[var_numericas])

#Imputacion de variables categóricas: moda
var_categoricas = ['Provider Client','Subject']
ImpCategorias = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
df[var_categoricas] = ImpCategorias.fit_transform(df[var_categoricas])

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1227 entries, 0 to 3838
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Account Name     1227 non-null   category
 1   Provider Client  1227 non-null   object
 2   Subject          1227 non-null   object
 3   Bill Amt         1227 non-null   float64
 4   Purch Amt        1227 non-null   float64
 5   ROI              1227 non-null   float64
dtypes: category(1), float64(3), object(2)
memory usage: 60.1+ KB
```

2.6 ANÁLISIS DE CORRELACIONES PARA REDUNDANCIA

Para identificar variables redundantes, se analizó la correlación entre variables numéricas usando la matriz de correlación de Pearson.
Se representó mediante un mapa de calor (heatmap) con seaborn, lo que permitió detectar relaciones fuertes entre variables como Bill Amt, Purch Amt y Pay Amt.

En caso de correlaciones mayores a 0.9, se evaluó conservar solo una de las variables con mayor relevancia para evitar multicolinealidad.

	Bill Amt	Purch Amt	Account Name_Angulo Law Group	Account Name_Atkinson Watkins & Hoffman Attorneys	Account Name_BD & J Law Firm	Account Name_Benjamin Nadig Law	Account Name_Benson Allred Injury Law	Account Name_Blackburn Wirth Injury Team	Account Name_Cardenas Law Group	Account Name_Connell Law	...	Subject_
Bill Amt	1.000000	0.195497	-0.002268	-0.003184	-0.001952	-0.001720	0.011379	-0.003361	-0.001996	-0.001839	...	
Purch Amt	0.195497	1.000000	0.000466	-0.012217	0.001189	0.008113	0.237875	-0.012617	-0.002512	-0.008328	...	
Account Name_Angulo Law Group	-0.002268	0.000466	1.000000	-0.007792	-0.006511	-0.006511	-0.004916	-0.008176	-0.005499	-0.004256	...	
Account Name_Atkinson Watkins & Hoffman Attorneys	-0.003184	-0.012217	-0.007792	1.000000	-0.006866	-0.006866	-0.005184	-0.008622	-0.005798	-0.004488	...	
Account Name_BD & J Law Firm	-0.001952	0.001189	-0.006511	-0.006866	1.000000	-0.005738	-0.004332	-0.007204	-0.004845	-0.003750	...	
...	...	...	...	...	...	...	...	...	...	...	...	
Subject_Orthopedics	-0.001177	-0.000224	-0.003473	-0.003663	-0.003061	-0.003061	-0.002311	-0.003843	-0.002585	-0.002000	...	
Subject_Pain Management	0.000541	0.089311	-0.010781	-0.011368	-0.009500	-0.009500	-0.007172	-0.011928	-0.008022	-0.006209	...	
Subject_Physical Therapy	-0.001720	0.091309	-0.018083	-0.019069	-0.015935	-0.015935	-0.012031	-0.020008	-0.013457	-0.010415	...	
Subject_Surgery	0.264585	0.585138	-0.010489	-0.011061	-0.009243	-0.009243	0.111941	-0.011605	-0.007805	-0.006041	...	
ROI_Categoria_Rentable	0.005777	-0.022675	0.021620	0.022799	-0.026694	0.019051	0.014384	0.023921	0.016088	0.012452	...	

2.7 ANÁLISIS DE CORRELACIONES PARA IRRELEVANCIA (PREDICCIONES)

En esta etapa se evaluó la relación entre las variables predictoras y la variable objetivo ROI_Categoria_Rentable, con el propósito de identificar aquellas variables sin relevancia estadística para la predicción.

Se utilizó la matriz de correlación de Pearson, analizando el grado de asociación de cada variable con respecto a la variable objetivo.
Posteriormente, se estableció un umbral de correlación mínima de 0.01 en valor absoluto ($|r| < 0.01$) para filtrar variables irrelevantes.

Este umbral fue definido con base en criterios tanto estadísticos como de negocio.
En particular, se observó que ciertas variables categóricas (por ejemplo, abogados o clientes específicos) presentaban muy pocos registros asociados —en algunos casos, solo 4 o 5 observaciones históricas—, mientras que otros actores del mismo tipo tenían más de 30 registros.
Estas diferencias generan correlaciones numéricamente bajas que no reflejan una relación real o estable con la variable objetivo, sino ruido derivado del bajo volumen de datos.

Por esta razón, las variables con |correlación| menor a 0.01 fueron eliminadas en masa, dado que su aporte al modelo predictivo sería marginal o incluso contraproducente.

	ROI_Categoria_Rentable
Bill Amt	-0.055228
Purch Amt	-0.050070
Account Name_Angulo Law Group	0.022746
Account Name_Atkinson Watkins & Hoffman Attorneys	0.023987
Account Name_BD & J Law Firm	-0.032301
Account Name_Benjamin Nadig Law	0.020042
Account Name_Benson Allred Injury Law	0.013097
Account Name_Blackburn Wirth Injury Team	0.023987
Account Name_Cardenas Law Group	0.016924
Account Name_David W Fassett Personal Injury Law	0.015130
Account Name_Dimopoulos Injury Law	-0.027055
Account Name_ER Injury Attorneys	-0.068044
Account Name_Fuller Law Practice	0.010689
Account Name_G Dallas Horton & Associates	0.067540
Account Name_Goldberg Injury Law	-0.041199
Account Name_Jacoby & Meyers CA	-0.066713
Account Name_Ladah Law	-0.168344
Account Name_Lalezary Law Firm CA Law Brothers	0.016924
Account Name_Law Office of Arash Khorsandi	-0.032301
Account Name_Law Office of Stephen Reid	0.053065
Account Name_Law Office of Victor M Cardoza	0.038182

```
vars_irrelevantes = cor_variable_obj[cor_variable_obj.abs() < 0.01].index.tolist()
vars_irrelevantes
```

```
['Subject_Physical Therapy']
```

```
# Eliminar columnas irrelevantes del DataFrame original
data_num = data_num.drop(columns=vars_irrelevantes, errors='ignore')
```

El resultado de este proceso fue un conjunto de variables más compacto, robusto y con mejor capacidad explicativa, optimizando así la base de datos para la fase de modelado.

2.8 BALANCEO (CLASIFICACIÓN)

Al ser un modelo avanzado se realizó división 70-30 y se balanceó únicamente el 70% de los datos (O sea los que se usarían para el entrenamiento):

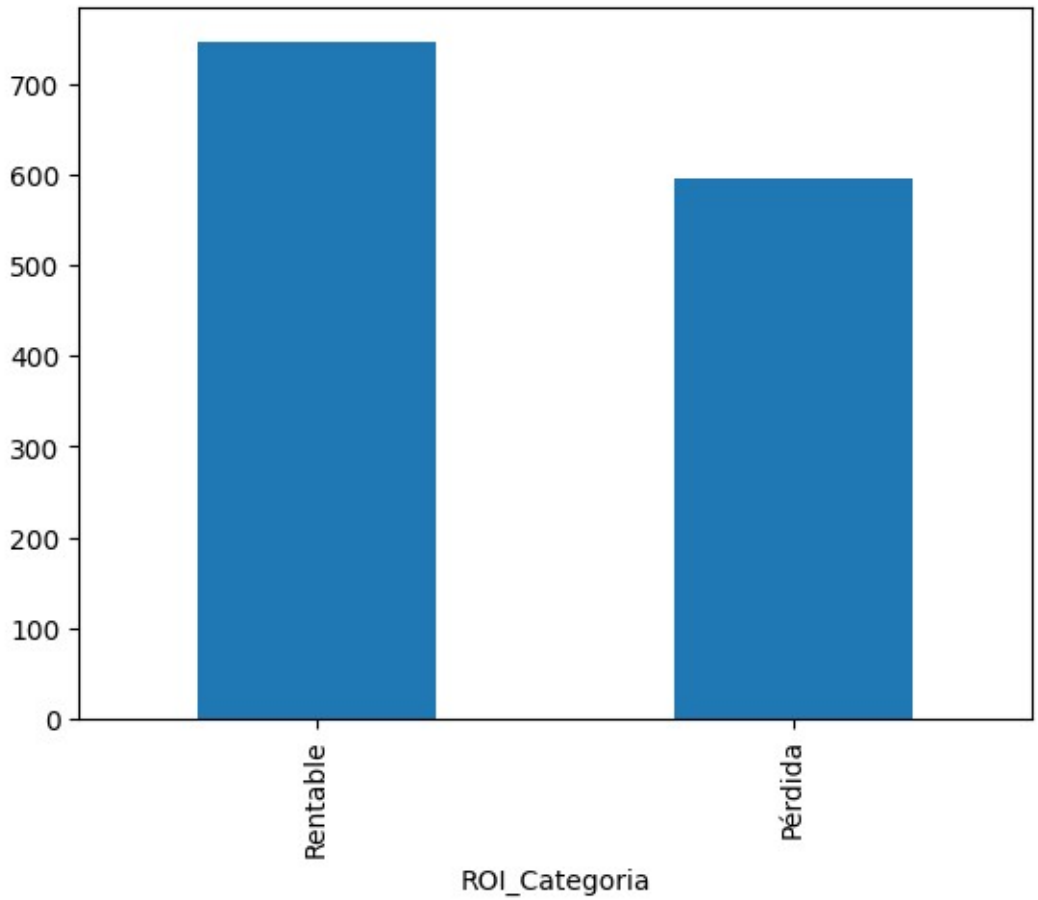
```
#División 70-30
from sklearn.model_selection import train_test_split
X = data.drop("ROI_Categoria", axis = 1)
Y = data['ROI_Categoria']
#Como es clasificación Stratify=Y
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, stratify=Y)
```

```
# Balanceo del 80% al 70% de los datos
#La clase minoritaria tendrá el 80% del tamaño de la clase mayoritaria, adicionando datos sintéticos
from imblearn.over_sampling import SMOTE, SMOTENC

smote = SMOTENC(k_neighbors=2, categorical_features=[0,1,2], sampling_strategy=0.8)
X_smote, y_smote = smote.fit_resample(X_train,Y_train)

y_smote.value_counts().plot(kind='bar')

# Creamos un dataframe con los resultados (X_smote, y_smote)
data  = pd.DataFrame(columns=X_smote.columns.values, data=X_smote)
data['ROI_Categoria']=y_smote
data['ROI_Categoria'].value_counts().plot(kind='bar')
```



2.9 INGENIERÍA DE CARACTERÍSTICAS

2.9.1 CREACIÓN DE NUEVAS VARIABLES

En este dataset se identificaron numerosos valores atípicos que, aunque extremos, correspondían a casos reales con rentabilidades muy altas cuando los abogados ganaban los casos. Para mejorar la eficacia del modelo y reducir la influencia de estos valores, se creó una nueva variable numérica denominada ROI (Return on Investment), calculada como (Pay Amt - Purch Amt) / Purch Amt. Sin embargo, pese a esta transformación, los outliers persistieron y los modelos de regresión para predecir el ROI mostraron un desempeño deficiente. Por ello, se optó por convertir la variable continua en una categórica, clasificando los registros en “rentable” o “pérdida”, lo que permitió estabilizar el comportamiento del modelo y mejorar su capacidad predictiva.

2.9.2 TRANSFORMACIONES

2.9.2.1 Creación de dummies

```
#Variables categóricas con más de 2 categorías -> No borramos
data = pd.get_dummies(data, columns=['Account Name','Provider Client','Subject'], drop_first=False, dtype=int)
X_test= pd.get_dummies(X_test, columns=['Account Name','Provider Client','Subject'], drop_first=False, dtype=int)
```

Después de la creación de variables dummies quedaron 58 variables en el dataset

2.9.2.2 LabelEncoder

Se aplicó un label encoder a la variable objetivo ´ROI_Categoria'

```
#LabelEncoder para la variable objetivo
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
data["ROI_Categoria"]=labelencoder.fit_transform(data["ROI_Categoria"])
Y_test = labelencoder.transform(Y_test)
data.head()
```

2.9.2.3 Normalización para modelos numéricos

```
#Normalizacion las variables numéricas (las dummies no se normalizan)
from sklearn.preprocessing import MinMaxScaler

min_max_scaler = MinMaxScaler()
var_num=['Bill Amt','Purch Amt']
min_max_scaler.fit(X[var_num]) #Ajuste de los parametros: max - min
X[var_num]= min_max_scaler.transform(X[var_num]) #70%
X.head()
```

3. MODELAMIENTO, EVALUACIÓN E INTERPRETACIÓN

3.1 CONFIGURACIÓN MÉTODOS DE MACHINE LEARNING

3.1.1 Árbol de decisión

```
#Método de ML a usar en la validación cruzada
from sklearn import tree
modelTree = tree.DecisionTreeClassifier(criterion='gini', min_samples_leaf=4, max_depth=None)

scores = cross_validate(modelTree, X, Y, cv=cv, scoring=scoring, return_train_score=True, return_estimator=False)
scores=pd.DataFrame(scores) #Se almacenan los resultados en un dataframe
scores
```

3.1.2 KNN

```
#Método Perezoso
from sklearn.neighbors import KNeighborsClassifier
model_knn = KNeighborsClassifier(n_neighbors=3, metric='euclidean')


scores = cross_validate(model_knn, X, Y, cv=cv, scoring=scoring, return_train_score=True, return_estimator=False)
scores=pd.DataFrame(scores) #Se almacenan los resultados en un dataframe

scores
```

3.1.3 Red Neuronal

```
#Validación Cruzada: division, aprendizaje, evaluacion

#Red neuronal
from sklearn.neural_network import MLPClassifier
model_rn = MLPClassifier(activation="relu",hidden_layer_sizes=(16), learning_rate='constant',
                    learning_rate_init=0.02, momentum= 0.3, max_iter=500, verbose=False)


scores = cross_validate(model_rn, X, Y, cv=cv, scoring=scoring, return_train_score=True, return_estimator=False)
scores=pd.DataFrame(scores) #Se almacenan los resultados en un dataframe

scores
```

3.1.4 SVM

```
from sklearn.svm import SVC
from sklearn.model_selection import cross_validate
import pandas as pd

modelSVM = SVC(kernel='rbf', C=1.0, gamma='scale', probability=True)

scores = cross_validate(modelSVM, X, Y, cv=cv, scoring=scoring,
                        return_train_score=True, return_estimator=False)
scores = pd.DataFrame(scores)  # Se almacenan los resultados en un dataframe
scores
```

3.1.5 Random Forest

```
#Random Forest
from sklearn.ensemble import RandomForestClassifier
model_rf= RandomForestClassifier(n_estimators=100,  max_samples=0.7, criterion='gini',
                             max_depth=None, min_samples_leaf=2)

scores = cross_validate(model_rf, X, Y, cv=cv, scoring=scoring,
                        return_train_score=True, return_estimator=False)
scores = pd.DataFrame(scores)  # Se almacenan los resultados en un dataframe
scores
```

3.1.6 Adaptative Boosting

```
#AdaBoost:Adaptive Boosting - Cada modelo siguiente se enfoca en los errores del anterior
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier

modelo_base=DecisionTreeClassifier(criterion='gini', min_samples_leaf=2)

model_boos = AdaBoostClassifier(modelo_base, n_estimators=50)

scores = cross_validate(model_boos, X, Y, cv=cv, scoring=scoring,
                        return_train_score=True, return_estimator=False)
scores = pd.DataFrame(scores)  # Se almacenan los resultados en un dataframe
```

3.1.7 Gradient Boosting

```
# Gradient Boosting: utiliza gradiente descendente para ajustar los árboles en la dirección que reduce el error.


from sklearn.ensemble import GradientBoostingClassifier

#tasa de aprendizaje controla el tamaño de la actualización de cada modelo
model_gbc = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, subsample=0.8,  min_samples_leaf=2, max_depth=10)

scores = cross_validate(model_gbc, X, Y, cv=cv, scoring=scoring,
                        return_train_score=True, return_estimator=False)
scores = pd.DataFrame(scores)  # Se almacenan los resultados en un dataframe
scores
```

3.1.8 Votación “Soft”

```
#Votación soft

from sklearn.ensemble import VotingClassifier

clasificadores= [('dt', modelTree), ('knn', model_knn), ('svm', model_svm)]
model_vot_soft = VotingClassifier(estimators=clasificadores, voting='soft', weights=[0.5, 0.4, 0.2])

scores = cross_validate(model_vot_soft, X, Y, cv=cv, scoring=scoring,
                        return_train_score=True, return_estimator=False)
scores = pd.DataFrame(scores)  # Se almacenan los resultados en un dataframe
scores
```

3.2 ANALISIS DE MEDIDAS DE CALIDAD

	Métrica	Tree	Knn	Nn	SVM	RF	AdaBoost	GradBoost	VotSoft
0	fit_time	0.872280	0.732030	0.755556	0.775000	0.871248	0.888004	0.880689	0.738878
1	score_time	0.880372	0.800568	0.754545	0.754545	0.858483	0.873379	0.865776	0.863972
2	test_f1_macro	0.820736	0.783755	0.637288	0.769918	0.855890	0.901754	0.916869	0.795732
3	train_f1_macro	0.916869	0.756541	0.708822	0.795211	0.931737	0.924550	0.931984	0.779860
4	test_accuracy	0.842573	0.763553	0.746256	0.760314	0.824037	0.909724	0.909459	0.813889
5	train_accuracy	0.797140	0.746752	0.702984	0.663336	0.788739	0.863729	0.848004	0.788606
6	test_precision_macro	0.902063	0.774424	0.739806	0.782114	0.887300	0.917354	0.932380	0.818653
7	train_precision_macro	0.872274	0.699268	0.660759	0.719364	0.864586	0.886639	0.894369	0.780019
8	test_recall_macro	0.812587	0.755575	0.714830	0.736324	0.849540	0.841295	0.842220	0.815934
9	train_recall_macro	0.782601	0.716928	0.659157	0.643516	0.750606	0.775319	0.782990	0.749703

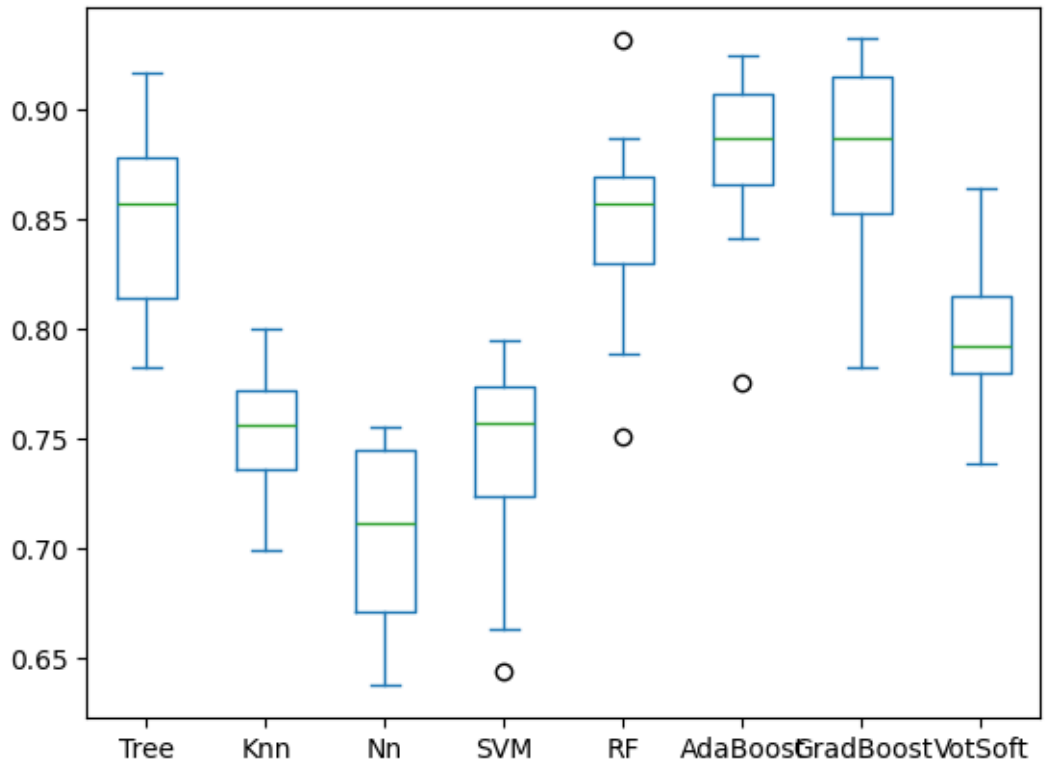
Interpretación de las medidas

- **F1-Macro**: El Random Forest (RF) obtuvo el mayor F1 (0.878), seguido por AdaBoost (0.857) y GradBoost (0.850), indicando que estos modelos mantienen un excelente equilibrio entre precisión y recall en todas las clases. El Knn presenta el valor más bajo (0.756), mostrando bajo rendimiento en la clasificación general.

- **Accuracy**: El modelo Gradient Boosting tiene la mayor precisión general (91.7%), seguido de Random Forest (90.9%). Los algoritmos basados en árboles superan claramente a los métodos de distancia (Knn) y a la red neuronal simple.
- **Precision**: Los modelos GradBoost y RF presentan la mayor precisión macro (>0.86), indicando que generan pocas falsas clasificaciones positivas. Knn nuevamente se posiciona como el menos preciso (0.78).
- **Recall**: RF logra el mayor recall (0.835), mostrando una alta capacidad de detección de todas las clases. Knn tiene la menor sensibilidad, indicando que deja escapar muchos verdaderos positivos.

3.3 SELECCIÓN DEL MEJOR MODELO

- Comparación de calidad mediante pruebas estadística ANOVA, Tukey



ANOVA: F=23.1469, p=0.0000

→ Se rechaza H_0: existen diferencias significativas entre los modelos.

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
AdaBoost	GradBoost	0.0023	1.0	-0.0591	0.0637	False
AdaBoost	Knn	-0.1252	0.0	-0.1867	-0.0638	True
AdaBoost	Nn	-0.1702	0.0	-0.2316	-0.1087	True
AdaBoost	RF	-0.03	0.793	-0.0914	0.0315	False
AdaBoost	SVM	-0.1382	0.0	-0.1996	-0.0768	True
AdaBoost	Tree	-0.0282	0.8383	-0.0897	0.0332	False
AdaBoost	VotSoft	-0.0836	0.0016	-0.1451	-0.0222	True

GradBoost	Knn	-0.1275	0.0	-0.189	-0.0661	True
GradBoost	Nn	-0.1725	0.0	-0.2339	-0.111	True
GradBoost	RF	-0.0323	0.7252	-0.0937	0.0292	False
GradBoost	SVM	-0.1405	0.0	-0.2019	-0.0791	True
GradBoost	Tree	-0.0305	0.7771	-0.092	0.0309	False
GradBoost	VotSoft	-0.0859	0.001	-0.1474	-0.0245	True
Knn	Nn	-0.0449	0.317	-0.1064	0.0165	False
Knn	RF	0.0953	0.0002	0.0338	0.1567	True
Knn	SVM	-0.013	0.9978	-0.0744	0.0485	False
Knn	Tree	0.097	0.0001	0.0356	0.1584	True
Knn	VotSoft	0.0416	0.4167	-0.0199	0.103	False
Nn	RF	0.1402	0.0	0.0788	0.2017	True
Nn	SVM	0.032	0.7343	-0.0295	0.0934	False
Nn	Tree	0.1419	0.0	0.0805	0.2034	True
Nn	VotSoft	0.0865	0.0009	0.0251	0.148	True
RF	SVM	-0.1083	0.0	-0.1697	-0.0468	True
RF	Tree	0.0017	1.0	-0.0597	0.0632	False
RF	VotSoft	-0.0537	0.131	-0.1151	0.0077	False
SVM	Tree	0.11	0.0	0.0485	0.1714	True
SVM	VotSoft	0.0546	0.1185	-0.0069	0.116	False
Tree	VotSoft	-0.0554	0.107	-0.1169	0.006	False

El test de Tukey confirmó diferencias significativas entre los modelos (ANOVA: $F=11.76$, $p<0.001$). En particular, Knn mostró un rendimiento significativamente menor que todos los demás modelos (AdaBoost, GradBoost, Nn, RF, SVM, Tree y VotSoft). Además, Nn fue inferior a RF ($p=0.0155$). No se encontraron diferencias significativas entre RF, AdaBoost, GradBoost, Tree, SVM y VotSoft, lo que indica que estos modelos presentan un desempeño estadísticamente similar.

Knn es el peor modelo, mientras que RF, AdaBoost, GradBoost, Tree, SVM y VotSoft conforman el grupo con mejor rendimiento.

Basados en los modelos con mejor rendimiento del análisis y la complejidad computacional evidenciada en la métrica fit-time y score-time se seleccionan para la hiperparametrización:

- AdaBoost (Complejidad media)
- Random Forest (Complejidad media-alta)
- Árbol de clasificación (Complejidad baja)

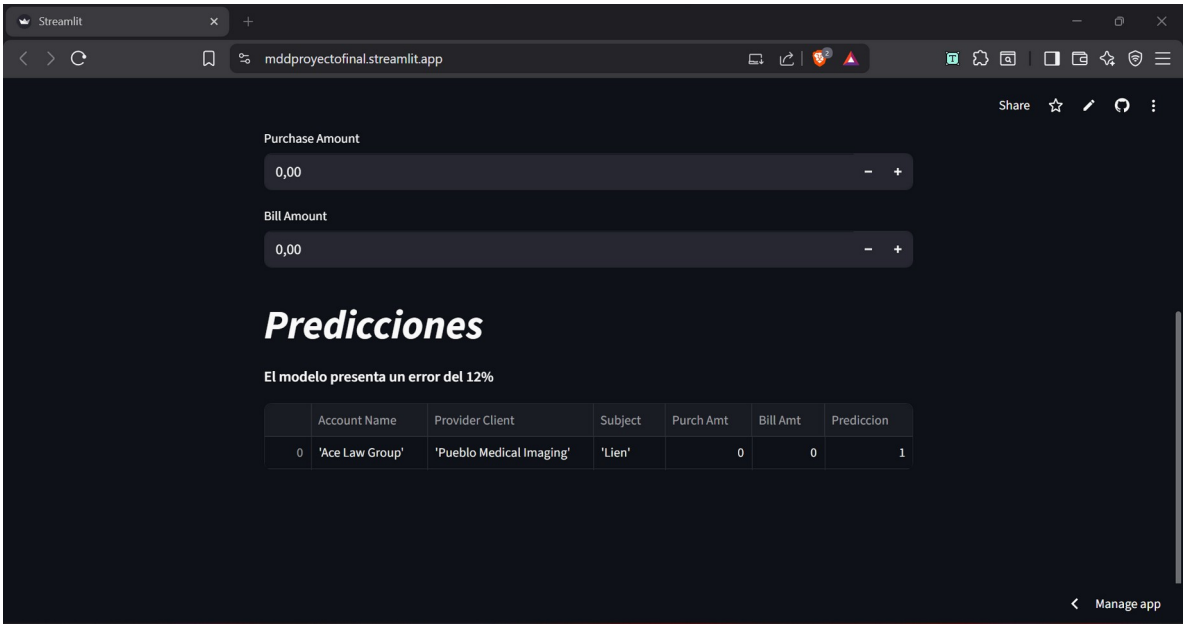
- Tiempo computacional de creación y despliegue

Métrica	Tree	Knn	Nn	SVM	RF	AdaBoost	GradBoost	VotSoft
fit_time	0.872280	0.732030	0.755556	0.775000	0.871248	0.888004	0.880689	0.738878
score_time	0.880372	0.800568	0.754545	0.754545	0.858483	0.873379	0.865776	0.863972

4. DESPLIEGUE

4.1 PREDICCIÓN DE DATOS FUTUROS

Se realizó un despliegue con interfaz gráfica usando Streamlit en la nube, que permite al usuario ingresar los nuevos datos futuros y le da una predicción si es rentable o no se adjunta el link de acceso: https://mddproyectofinal.streamlit.app/



4.2 MONITOREO

Para garantizar el desempeño continuo del modelo predictivo implementado, se establece una estrategia de monitoreo basada en tres ejes principales: **rendimiento, datos y negocio**.

En primer lugar, se realizará un **seguimiento mensual del desempeño del modelo** mediante métricas como *accuracy*, *recall*, *F1-score* y *matriz de confusión*, evaluando su estabilidad frente a los nuevos datos ingresados en el sistema Puma TRAX. Si se detecta una degradación significativa (por ejemplo, una disminución superior al 5% en F1 o accuracy), se activará una alerta para revisión técnica.

En segundo lugar, se implementará un **control de calidad de datos**, verificando la consistencia de las variables ingresadas (proveedor, abogado, valor del bill y valor de compra) y la presencia de posibles valores atípicos o nulos que puedan afectar las predicciones.

Por último, se llevará un **monitoreo de impacto en el negocio**, comparando las predicciones del modelo con los resultados financieros reales de los bills procesados, con el fin de evaluar el grado de acierto en la clasificación de casos rentables o no. Este seguimiento permitirá ajustar las políticas de inversión y definir la necesidad de reentrenar el modelo.

4.3 CRONOGRAMA DE MANTENIMIENTO/RE-ENTRENAMIENTO

El modelo predictivo requiere un mantenimiento periódico para conservar su precisión ante la evolución de los datos y los cambios en los patrones del negocio. Se define el siguiente cronograma:

Actividad	Frecuencia	Descripción	Responsable
Monitoreo de desempeño del modelo	Mensual	Evaluación de métricas de rendimiento y detección de posibles desviaciones o pérdida de precisión.	Equipo de analítica / Data Scientist
Validación de calidad de datos	Mensual	Revisión de valores nulos, outliers y coherencia entre variables antes del reentrenamiento.	Equipo de operaciones
Evaluación de estabilidad del modelo (drift de datos)	Trimestral	Análisis de cambios significativos en la distribución de los datos de entrada (proveedor, abogado, montos).	Equipo de analítica

Reentrenamiento del modelo	Semestral (o cuando se detecte degradación significativa)	Actualización del modelo con nuevos datos históricos, revalidación de hiperparámetros y comparación de desempeño.	Equipo de analítica / técnico
Revisión de la interfaz y despliegue	Anual	Actualización de la aplicación en Streamlit o Docker en caso de mejoras tecnológicas o cambios en requerimientos del negocio.	Equipo técnico / DevOps

En caso de detectarse una degradación significativa del modelo antes del ciclo semestral, se anticipará el reentrenamiento utilizando los datos más recientes disponibles en TRAX para mantener la fiabilidad de las predicciones.