DATA SCIENTIST TECHNICAL CHALLENGE

MERCADO LIBRE.

09/03/2025

Julián Cujabante Villamil.

2. Realizar un pequeño informe de la solución propuesta:

ANÁLISIS Y TRANSFORMACIONES

El dataset provisto cuenta con 150 mil filas y 17 variables entre las que se encuentra la variable objetivo: fraude.

La mayoría de las variables están anonimizadas, lo cual dificulta entender a primera vista la relación entre estas y la variable objetivo.

Se realizó un EDA (Exploratory Data analysis) del dataset para evaluar la calidad de este y evaluar las relaciones entre las variables independientes con la variable objetivo.

DESBALANCE DE DATOS.

El primer insight que sale a la luz es la particularidad presente en el mundo del fraude y es un marcado desbalance de los datos en cuanto a las etiquetas. La etiqueta 1 de interés que es fraude esta subrepresentada al ser solamente el 5% del total de las etiquetas.

Este desbalance tan marcado supone un problema para la aplicación de modelo de machine learning, pues se materializará en un problema de underfitting, en el cual el algoritmo no cuenta con suficientes casos de la etiqueta 1 para aprender a generalizar esta.

Existen varias alternativas para tratar esto:

1. Emplear modelos robustos al desbalance de clase como los modelos basados en árboles, bagging y boosting, como Xgboost y Lightgbm.
2. Tratar el problema desde un enfoque muestral, bien sea aumentando la etiqueta minoritaria de manera artificial (Oversampling) con métodos como: SMOTE, SMOTE TOMEK, ADASYN , ROS , etc. O disminuyendo la etiqueta mayoritaria (Undersampling) para que tenga una proporción similar a la etiqueta minoritaria, usando RUS, NearMiss, etc.
3. Enfoque basado en cost sensitive learning, modificando los pesos de los modelos (por ejemplo, LightGBM, XGBoost) para que el equivocarse clasificando la clase minoritaria sea muy costoso para el modelo.

CALIDAD DE DATOS.

DATOS SUPLICADOS

No se encontraron registros duplicados en el dataset.

DATOS FALTANTES.

Hay presencia severa de outliers en ciertas variables del dataset. En la variable ‘o’ la proporción de datos faltantes es de cerca del 72 %, seguido por las variables ‘b’ y ‘c’ con 8.6%.

A continuación, se presenta tabla con esta información:

Variables	Valores Faltantes	Porcentaje (%)
o	108857	72,57
b	12984	8,66
c	12984	8,66
d	365	0,24
m	365	0,24
g	194	0,13
f	11	0,01
l	11	0,01

Esta proporción de datos faltantes en algunas de las variables, cómo es el caso de ‘o’ es alarmante pues es muy elevada, haciendo que usarla sea casi imposible.

Los métodos usuales para tratar este problema nos dictan:

ELIMINACIÓN DE OBSERVACIONES O COLUMNA CON DATOS FALTANTES.

Esto conllevaría a una pérdida de información, lo cual podría ser crítico en este caso, pues si los valores faltantes de X variable se relacionan con la etiqueta 1 de la variable objetivo, eliminar estos registros ahondaría en el problema de desbalance de clases. Adicionalmente, dado que no se sabe con certeza que variable es que, en el dataset, eliminar observaciones o columnas puede llegar a ser irresponsable.

IMPUTACIÓN DE DATOS FALTANTES.

Se puede intentar imputar los valores faltantes usando valores centrales como la mediana en caso de las variables numéricas o la moda en el caso de las variables categóricas. También en este último caso se podría crear una categoría extra que corresponda a valor faltante. También se puede imputar los valores con métodos más sofisticados basados en KNN.

Este método no supone una pérdida de la información, pero también debe usarse con precaución, porque podría llegar a cambiar la distribución de la variable afectada cambiando radicalmente la relación con la variable objetivo. Por ejemplo, en el caso de la variable ‘o’ que tiene un 72% de missings, imputar un valor hará que la variable cambie totalmente.

FEATURE ENGINEERING BASADO EN PATRONES ENCONTRADOS EN LOs MISSING Y VARIABLE OBJETIVO.

Se puede asimismo explorar la relación que puede existir o no entre los valores faltantes y la variable objetivo. De existir esta, se puede crear variables que traten de capturar esta relación, como una variable binaria que indique cuando el valor de la variable original es vacío o no. Esta nueva información podría ser explotada por los modelos, que podrían aprender que existe una relación entre los valores faltantes y alguna de las etiquetas de su variable objetivo.

El enfoque que se usó en este reporte fue una combinación de las alternativas 2 y 3. Por un lado, se exploró la relación entre los valores faltantes y el fraude. Se evidenció que, en efecto en algunas variables, la tasa de fraude de los missings era más alta. Por tal motivo, se crearon variables binarias que indicaban 1 si el valor de la variable original era faltante y 0 de lo contrario. Esta hipótesis se corroboró aplicando una prueba de chi cuadrado entre las variables binarias recién creadas y la variable fraude. Mostrando que si existe una relación entre los valores faltantes y fraude.

Variables	Tasa Fraude (%) cuando es nulo	Tasa Fraude (%) cuando NO es nulo	Diferencia (%)
b	6,38	4,87	1,52
c	6,38	4,87	1,52
d	7,95	4,99	2,95
f	0,00	5,00	-5,00
g	7,73	5,00	2,74
l	0,00	5,00	-5,00
m	7,95	4,99	2,95
o	2,05	12,79	-10,74

De igual manera, se realizó una imputación basada en el tipo de la variable, con la mediana en las numéricas y creando otra categoría en las variables categóricas.

Las variables binarias creadas en este paso se incluyeron en el ejercicio de clasificación.

OUTLIERS.

Al revisar las variables numéricas se encontró presencia de valores atípicos. En algunas variables más que otras, como en ‘c’, ‘f’, ‘monto y ‘b’.

Variable	Número de Outliers	Total de Valores	Porcentaje de Outliers (%)
c	16946	137016	12,37
f	17116	149989	11,41
monto	14823	150000	9,88
b	8554	137016	6,24
e	7580	150000	5,05
h	5045	150000	3,36

m	4499	149635	3,01
l	384	149989	0,26

Estos valores atípicos pueden desorientar al modelo. Razón por la cual se sugieren tratarlos.

ELIMINACIÓN DE VALORES ATÍPICOS.

Al igual que con la alternativa de eliminar las observaciones que presentan datos faltantes, Una opción es eliminar aquellas observaciones que presentan valores extremos. No obstante, esta opción no se consideró, pues la eliminación de observaciones con datos extremos que estuvieran relacionados de alguna manera con la etiqueta 1 de fraude podría acrecentar el problema de desbalance de clase, sumado a la perdida de información que supone esto.

Truncar los valores atípicos. Se pueden truncar los valores atípicos son Winsorización, se puede cambiar por valores de tendencia central como la mediana, la moda dependiendo de tipo de variable, se podrían escalar las variables con transformaciones logarítmicas o con escaladores robustos.

FEATURE ENGINEERING BASADO EN LAS RELACIONES ENTRE OUTLIERS Y VARIABLE OBJETIVO.

Se podría explorar la relación entre los valores atípicos y la variable objetivo en caso de ser muy notoria se podría crear una variable que capturase esta información.

Para este reporte se empleó la alternativa 2 aplicando un escalador robusto en la etapa de preprocesamiento. Cuando se exploró si existía una relación entre los datos atípicos y la variable fraude, si se encontraron diferencias entre ambas poblaciones, más no se consideraron significativas.

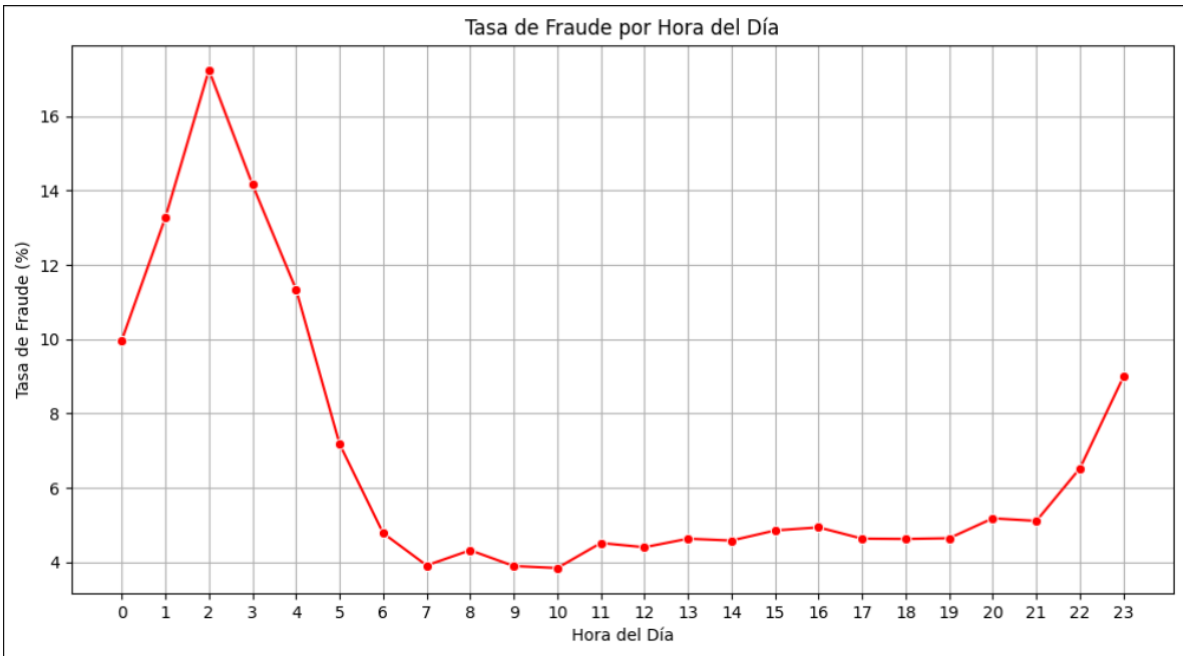
HALLAZGOS GENERALES.

De los diferentes tipos de datos presentes en el dataset (fecha, categórico numérico), se encontró lo siguiente:

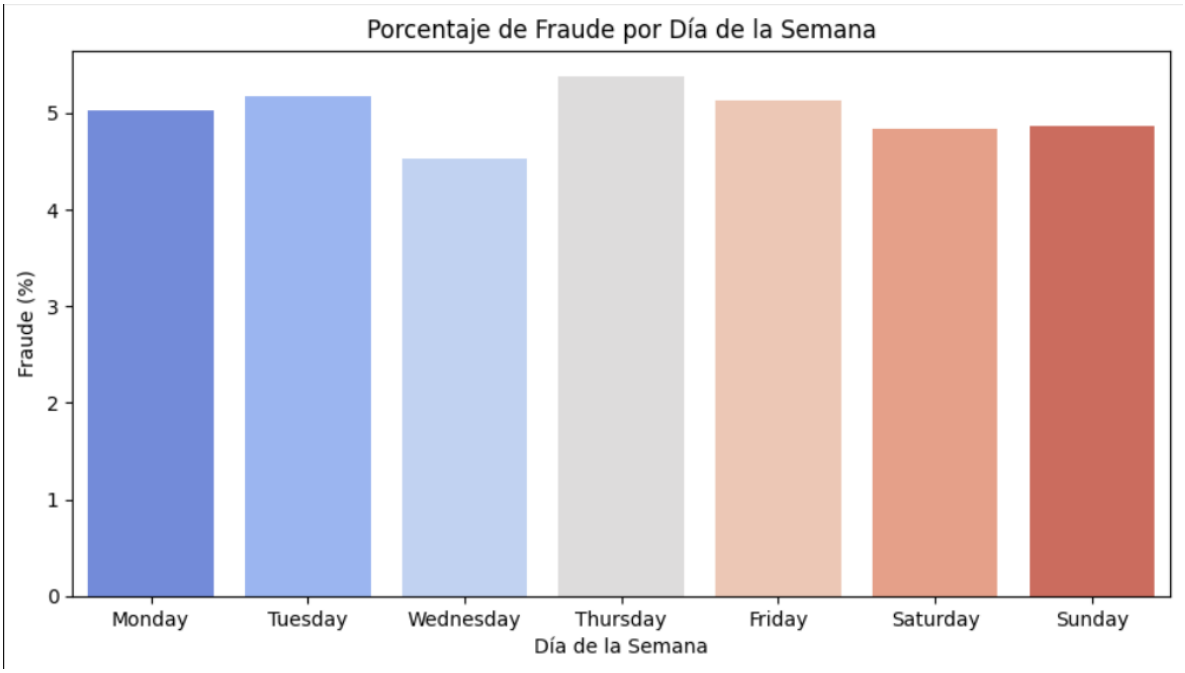
VARIABLE TIPO FECHA.

Solo hay una variable tipo fecha. Se construyeron variables adicionales en base a esta para tratar de capturar la relación entre esta variable y la variable objetivo.
Decidió no usarse esta variable en el entrenamiento del modelo, pues la alta cardinalidad de esta podría producir un efecto adverso en el algoritmo.

De esta manera, se creó una variable de hora, la cual nos dejó ver que existía una aparente relación entre las horas de la noche y madrugada con el fraude. Por tal motivo se crea una variable binaria indicando si se trata de una hora de riesgo o no.



Asimismo, se creó la variable día_semana, pues se evidencio una ligera relación entre ciertos días y el fraude.



VARIABLES NUMÉRICAS.

La variable score es enigmática, pues su nombre indica que es una especie de valoración sobre la transacción. Se puede creer que es una variable de puntaje de riesgo de la transacción, pues se puede evidenciar que su tasa de fraude es creciente a medida que los valores de la variable aumentan. Suponiendo que se trata de una variable que captura el riesgo de la transacción o del perfil del cliente se usará en el modelo.

score	tasa de fraude
(-0.001, 8.0]	2,78
(8.0, 18.0]	4,71
(18.0, 28.0]	3,06
(28.0, 38.0]	0,33
(38.0, 48.0]	0,63
(48.0, 58.0]	1,31
(58.0, 68.0]	2,14
(68.0, 78.0]	5,32
(78.0, 88.0]	8,89
(88.0, 100.0]	21,80

- Ninguna de las distribuciones de las variables numéricas sigue una normal.
- No hay correlaciones lineales fuertes entre las variables,
- Se encontró por ejemplo que la relación biserial y que las diferencias entre ambas poblaciones: fraude y no fraude de la variable ‘k’ no eran relevantes. Por tal motivo no se incluyó en el modelo.

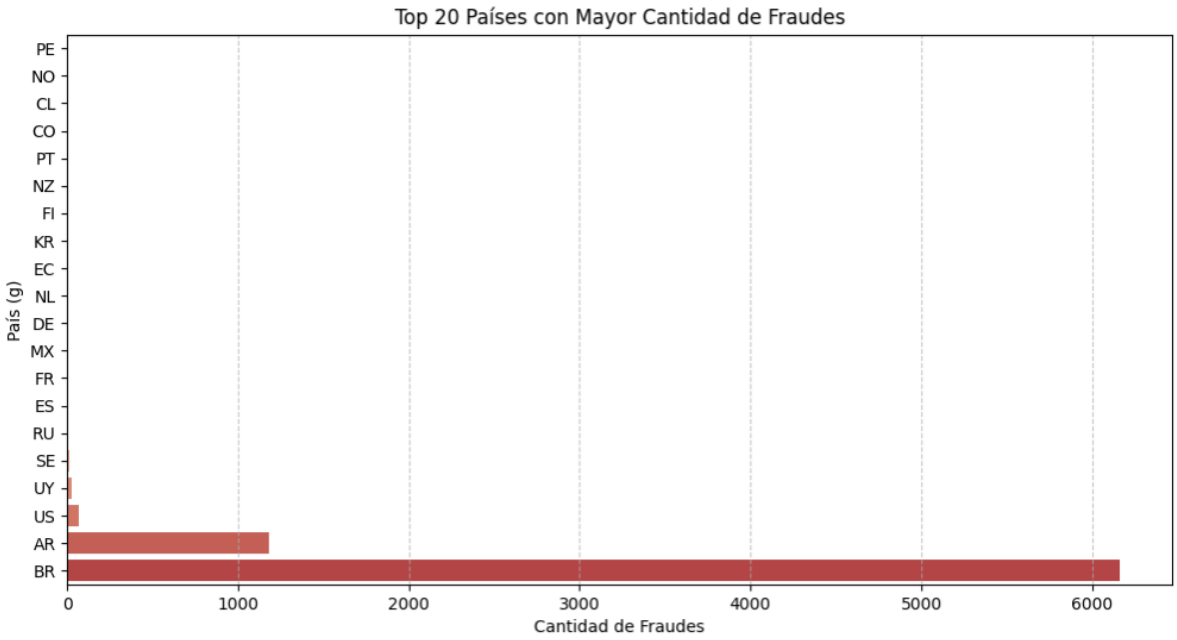
VARIABLES CATEGÓRICAS.

Dos de las variables categóricas tenían una alta cardinalidad. Una parecía referirse a productos o ítems por los valores encontrados en esta (variable ‘j’) y la otra variable estaba relacionada con países (‘g’).

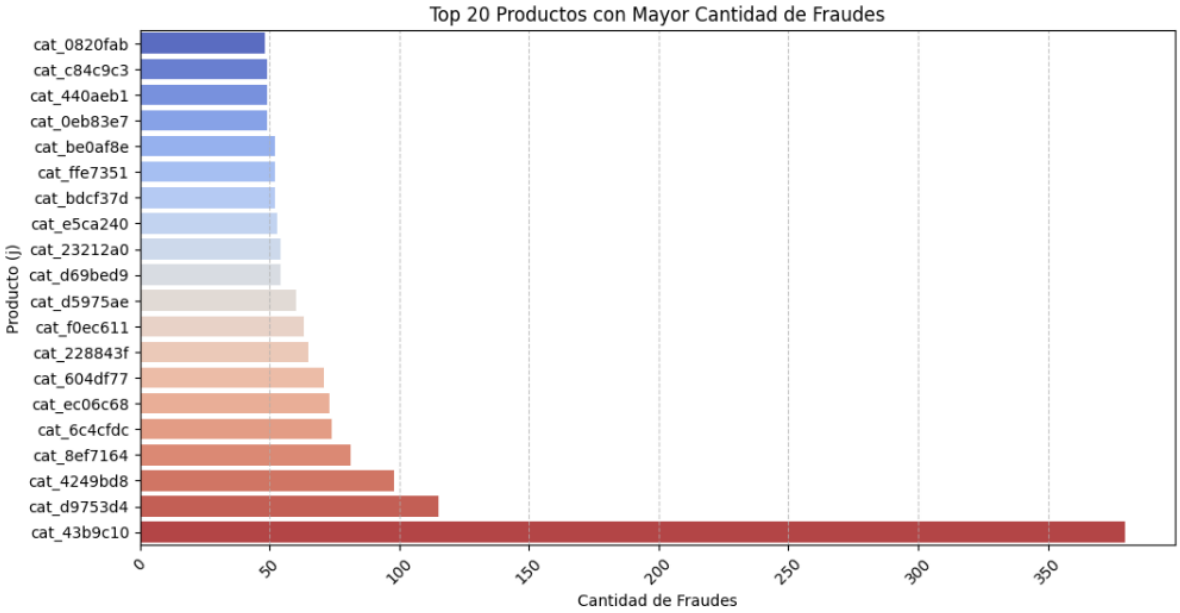
Una primera suposición sobre ‘g’ es que se trata de los países de la tarjeta de crédito. Esto en base a que, si se trata de datos de la empresa, esta no tiene operación en algunos de los países listados en la variable como Taiwán, Rusia, etc.

En ambas variables de alta cardinalidad se logró encontrar que si guardaban relación con la variable fraude.

De esta manera, en la variable ‘g’ se encontraron concentraciones de etiquetas de fraude en ciertos países.



Para la variable ‘j’, se vio que ciertos de los productos concentraban una mayor cantidad de etiquetas de fraude.



Ambas fueron evaluadas usando la tasa de fraude de sus valores y restringiendo que la tasa de fraude no se viera afectada o sesgada por pocas ocurrencias, en otras palabras, no se tuvieron en cuenta tasa de fraude de productos o países del 100% debido a que solo tenían una transacción y esa fue fraude.

Luego del análisis, se crearon dos variables nuevas, una binaria para la variable original ‘j’ la cual indica si se trata de un producto riesgoso o no

Y una para la variable ‘g’ construida mediante target encoding para reflejar la tasa de fraude de los países.

Ambas variables serán usadas en el modelo.

FEATURE ENGINEERING.

En cada uno de los apartados anteriores se fue hablando de la creación de variables que respondían a un problema de calidad de datos, de relación con la variable objetivo, o de alta cardinalidad. A continuación, se listan las nuevas variables creadas a partir de las variables originales con una breve razón del porque fueron creadas.

VARIABLE ORIGINAL	NUEVA VARIABLE	TIPO DE VARIABLE	RAZÓN
b	b_missing	Categórica	Relación datos faltantes con fraude
c	c_missing	Categórica	Relación datos faltantes con fraude
d	d_missing	Categórica	Relación datos faltantes con fraude
m	m_missing	Categórica	Relación datos faltantes con fraude
o	o_missing	Categórica	Gran proporción de missings
fecha	dia_semana	Categórica	Relación entre los días de la semana y fraude
fecha	high_risk_hour	Categórica	Relación entre ciertas horas del día y fraude
g	g_encoded	Numérica	Relación entre los países y fraude
j	high_risk_product	Categórica	Relación entre ciertos productos y fraude

EJERCICIO DE CLASIFICACIÓN.

Para desarrollar el ejercicio de clasificación supervisada, se llevó a cabo en varios pasos los cuales se listarán y explicarán a continuación:

PREPROCESAMIENTO DE LOS DATOS.

Para ejecutar este se creó un pipeline de preprocesamiento de datos para tratar los problemas de calidad de datos presentes en el dataset.

- Imputación de datos faltantes: Si la variable es numérica se utiliza la mediana. Si la variable es categórica se crea una nueva categoría missing.
- Escalador Robusto: Sólo aplica para las variables numéricas para tratar los outliers presentes y estandarizar todas las variables en una misma escala.
- Numerización de categorías: Sólo aplica a las variables categóricas. Se emplea OHE y se elimina una de las variables para evitar la trampa de la variable binaria o multicolinealidad perfecta en los modelos lineales como la regresión logística. Se empleó también para modelos como el lLghtGBM y XGBoost a pesar de que estos pueden trabajar con datos categóricos directamente, para poder aplicar SMOTE.

MODELOS UTILIZADOS.

Los modelos utilizados en este ejercicio son la regresión logística, el XGboost y el LightGBM.

Se definieron grillas de hiperparámetros de cada uno, para maximizar el aprendizaje, el rendimiento y controlar por overfitting.

Entre los hiperparámetros utilizados por modelo tenemos:

Regresión logística: c, ma_iter, penalty y solver.

XGBoost: n_estimators, max-depth, learning_rate, gamma, subsample, colsample_bytree, scale_pos_weight, etc.

LightGBM: n_estimators, max_depth, num_leaves, subsample, scale_pos_weight, etc.

HYPER PARTAMETER TUNNING

Se empleó Optuna para la optimización de hiperparámetros de los modelos. La función objetivo a maximizar sería la función de ganancia construida.

EVALUACIÓN DE LOS MODELOS.

Luego de encontrar los mejores hiperparámetros para cada modelo, se emplean estos para evaluar contra test. Esta evaluación se realiza teniendo en cuante la función de ganancia, las métricas usuales de clasificación (recall, precision, F1, Precision-Recall, etc), graficando la matriz de confusión, curvas ROC y Precison-Recall y en caso de los modelos arbóreos la importancia de las variables y Shap values.

A continuación, se listan las siguientes ejecuciones de los diferentes modelos siguiendo diferentes técnicas de balanceo de datos y pasando por optimización de hiperparámetros.

Modelos	Técnica de balanceo	Ganacia	F1	Recall	Precision	AUC PR
Regresión Logistica	Cost Sensitive Learning	238111,41	0,0053	0,0027	0,5714	0,2080
Regresión Logistica	SMOTE	201040,40	0,2628	0,7193	0,1608	0,3083
Regresión Logistica	RUS	201113,08	0,2605	0,7433	0,1580	0,3438
XGBoost	Cost Sensitive Learning	247791,67	0,3868	0,2807	0,6219	0,4368
XGBoost	SMOTE	251570,18	0.4153	0.3047	0,6519	0.4518
XGBoost	RUS	206257,53	0,3083	0,8133	0,1902	0,4522
LightGBM	Cost Sensitive Learning	250305,01	0,3871	0,2767	0,6444	0,4663
LightGBM	SMOTE	251464,27	0,4151	0,3047	0,6510	0,4625
LightGBM	RUS	206435,82	0,3097	0,8180	0,1910	0,4494

Contrastando con el valor máximo de ganancia que se puede obtener el set de prueba, realizando una clasificación perfecta de las dos clases:

Hay dos escenarios a analizar en este ejercicio, pro un lado, se obtuvo modelos que eran buenos detectando el fraude con un recall muy alto, pero que a su vez generaban muchos falsos positivos, tantos que el ahorro en fraudes se compensaba negativamente con las transacciones legitimas que fueron rechazadas pro ser consideradas como fraude cuando

no lo eran. Estos modelos fueron curiosamente los que emplearon undersampling. A pesar de haber capturado la mayor cantidad de fraudes tuvieron los niveles más bajos de ganancias.

Por otro lado, se presentó el caso contrario, modelos que tenían una ganancia superior al previo. Que te lograban esta mayor ganancia no detectando fraudes, sino reduciendo los falsos positivos. Estos modelos tenían así una precisión alta, pero dejaban pasar los fraudes. Los modelos que cayeron en esta clasificación fueron los que usaron oversampling y cost sensitive learning.

Como hallazgo general, se pudo corroborar con la importancia de características y Shap Values que muchas de las variables propuestas resultaron ser las más importantes en el modelo.

Adicional a este ejercicio se realizó otro adicional que pueden encontrar al final del notebook, en el cual se adicionalmente a lo descrito se optimizo el umbral de decisión del modelo para maximizar la ganancia. Efectivamente se maximizó esta de manera maquiavélica disminuyendo a casi cero los falsos positivos. En conclusión, se maximizó la ganancia, pero las implicaciones son muy extremas.

Estos son los resultados obtenidos en este ejercicio adicional:

Modelos	**Técnica de balanceo**	**Ganacia**	**F1**	**Recall**	**Precision**	**AUC PR**
LightGBM	Cost Sensitive Learning	287354,25	0,0769	0,0400	1,0000	0,4694

CONCLUSIÓN.

Se concluye que el ejercicio de la maximización de las ganancias de la empresa puede resultar en escenarios complejos. ¿Teóricamente se está maximizando esta, pero a que costo?

Así, tenemos casos en los cuales la maximización de la ganancia iría en línea con la métrica de Accuracy en casos de desbalance de datos. Donde seria alta por la correcta clasificación de la mayoría de las transacciones legitimas (la gran mayoría). Pero no garantizando una captura de los fraudes.

En otras palabras, es muy probable que un modelo de maximice las ganancias sea aquel que priorice clasificar la clase mayoritaria evitando que se rechace las transacciones legitimas, puesto que por el volumen de transacciones legitimas se obtenga una mayor maximización que cuando el modelo es bueno capturando los casos de fraude (escenarios donde se obtuvo recall muy altos, pero una baja ganancia). Esto está implícito en la construcción de la función de ganancia, puesto que se obtiene el 25% de las transacciones legitimas (la gran mayoría) que compensa con creces las pérdidas de las anómalas transacciones fraudulentas.

Esto lo pudimos ver en ejercicios cuando con el undersampling se lograban altos niveles de ganancia teniendo un recall pequeño. Esto mismo lo pudimos evidenciar con el modelo bonus que optimizaba el umbral de decisión en base a la ganancia. Los resultados eran

altos, pero eran basados en disminuir casi a cero los falsos positivos y dejando pasar buena parte del fraude.

En ese orden de ideas quizá la maximización de las ganancias deba ir también acompañado de otras reglas de negocio para que así no se logre esta maquiavélicamente no rechazando transacciones legitimas.

3. ¿Qué pasos puedo seguir para intentar asegurar que la performance del modelo en laboratorio será similar a la de producción?

División de datos adecuada

- Asegurarse que los datos de entrenamiento, test y validación reflejen una distribución real de los datos de producción.

Simulación del entorno de producción.

- Usar datos reales y recientes evitando usar datos históricos obsoletos que pueden presentar data drift.
- Evaluar el modelo con datos de producción no vistos antes del deployment.

Controlar Overfitting.

- Aplicar controles de overfitting en los hiperparámetros del modelo, para que este no se aprenda de memoria el dataset de entrenamiento y pueda generalizar correctamente ante nuevos datos.
- Evitar modelos muy complejos que tienden a llevar al overfitting.

Reproducibilidad del preprocesamiento utilizado.

- Aplicar el preprocesamiento tal como se aplicó en entrenamiento. Por ejemplo, si para los problemas de calidad de datos como los valores faltantes se emplearon en las variables categóricas la creación de una nueva categoría missing en entrenamiento y en producción se imputan con la moda, posiblemente las variables categóricas cambien mucho al igual que el rendimiento. Otro ejemplo puede ser, sí se aplican técnicas de balanceo de oversampling en entrenamiento y en producción de undersampling, esto afectará el rendimiento del modelo.

4. Suponiendo que la performance predictiva en producción es muy diferente a la esperada, ¿Cuáles cree que son las causas más probables?

Cambio en la distribución de datos (Data Drift o Concept Drift).

- Las características de los fraudes pueden haber cambiado. Esto es factible dado que la data con la que se entrenó el algoritmo es del 2020 y si se ‘productiviza’ a 2025, las tendencias y patrones de fraude pudieron haber evolucionado.
- Las variables predictoras ya no son igual de relevantes. Esto se relaciona con el punto anterior, pues si los patrones de fraude han cambiado en 5 años, posiblemente las variables que en el entrenamiento de este modelo fueron relevantes quizá en 2025 ya no lo sean.

Problemas de muestreo o sesgo en los datos de entrenamiento.

- Si los datos de entrenamiento no representaban correctamente los datos reales de producción, el modelo no generalizará bien.
- Si los datos con los que se entrenó este modelo presentaban un sesgo y hace que difieran mucho a los de producción, el modelo no generalizará bien.

Ruido en los datos en producción

- Si en producción hay muchos más datos incorrectos, inconsistentes o faltantes, el modelo puede fallar más. Así, por ejemplo, si en producción las variables tienen menos datos faltantes que en el dataset de entrenamiento. Por ejemplo, la variable 'o' en entrenamiento tenía cerca del 72% de su distribución como faltante, pero si en producción esta variable no contiene missings, esto afectaría la predicción, porque en entrenamiento se tomó una estrategia para tratar este problema, problema que no es presente en producción.

Desbalance en los datos de producción respecto al entrenamiento:

- Puede haber una diferencia en la proporción de fraudes en producción comparada con la de entrenamiento, afectando la sensibilidad del modelo. Suponga que en el entrenamiento teníamos que la clase minoritaria tenia apenas un 5%, pero que en producción la realidad es más desbalanceada llevando a que la etiqueta 1 tenga menos de 1%.

Overfitting al datset de entrenamiento.

- Si el modelo está demasiado ajustado a los datos de entrenamiento, puede fallar en datos nuevos y no generalizar correctamente.

Diferencias en el pipeline:

- Diferencias entre el preprocesamiento en entrenamiento y el aplicado en producción. Por ejemplo, si en entrenamiento se imputaron los valores numéricos usando la mediana, pero en producción usando la media.

5. ¿Qué pasos debería seguir para poner el nuevo modelo en producción?

Desplegar en un entorno de prueba (Shadow Deployment) y monitorearlo

- Ejecutar el modelo en paralelo con el actual sin tomar decisiones basadas en sus predicciones. Esto permitirá comparar los resultados de ambos y evaluar el rendimiento del nuevo modelo (F1, AUC-PR, Recall, ganancia, etc).
- A/B testing contra el modelo antiguo.
- Se pueden implementar alertas si las métricas bajan de un umbral definido.

Asegurar reproducibilidad y escalabilidad.

- Usar servicios en nube que faciliten el escalado y que garanticen consistencia del entorno del modelo.
- Definir tiempos de latencia aceptables.

Sistema de feedback continuo:

- Implementar un sistema que recopile feedback de los fraudes detectados y aprobados para ajustar el modelo. Esta valiosa información no solo servirá para ajustar el modelo, sino para futuros reentrenamientos.
- Detección de cambios en la distribución de los datos, en los patrones de fraude.

Reentrenamiento periódico.

- Se puede apalancar con el sistema de feedback continúo dado que se recopilaría información valiosa de como va evolucionando el fraude y se podría usar para en reentrenamiento.
- Podrían automatizarse partes del proceso usando MLOps.

Implementación de estrategias de mejora continua.

- Realización de auditorías periódicas para evaluar y asegurar que el modelo sigue siendo rentable.