PRUEBA
CIENTÍFICO DE DATOS

Prueba de habilidades y aptitudes Científico de datos Jr.

Tabla de Contenido

Introducción

El objetivo de esta prueba es demostrar tus habilidades desde los siguientes puntos de vista:

- Herramientas como R/PYTHON (a elección)
- Análisis de Datos
- Desarrollo de Modelos
- Argumentación
- Curiosidad e Investigación

Contesta este examen de manera individual. Puedes buscar información en sitios web si así lo requieres. Nos gustaría conocer cómo abordas los problemas, por favor documenta qué herramientas usaste.

A. Conocimientos de Riesgo de Crédito.

1. ¿Cuáles son las etapas en el ciclo del crédito?

Después de investigar brevemente en internet (y por experiencia personal), esto fue lo que pude sintetizar:

El ciclo de un crédito se compone de varias etapas que aseguran el cumplimiento del préstamo de ambas partes, algunas de estas etapas son:

- Solicitud del crédito: en esta etapa el solicitante presenta una solicitud junto con documentos básicos de información personal para su pre-aprobación.
- Aprobación o rechazo: después de hacer una evaluación, como revisar su historial crediticio, el crédito se aprueba o se rechaza de acuerdo con los hallazgos y se presenta el plazo, la tasa de interés, garantías y lo necesario para firmar el contrato.
- Desembolso: se realiza el desembolso por la cantidad solicitada para que el cliente haga uso de los fondos
- Pago del crédito: el cliente realiza los pagos según los términos acordados
- Seguimiento y control: a la par con la etapa anterior, la entidad que otorgó el crédito se asegura y verifica que los pagos se estén realizando en tiempo y forma, de igual manera podrían analizar una reestructuración del crédito o un aumento del mismo si el cliente lo requiere.
- Recuperación o cierre del crédito: se da una vez que el cliente ha pagado el préstamos en su totalidad y la entidad financiera libera el bien asociado al crédito o la garantía dejada por el mismo.

2. Menciona algún indicador de riesgo de crédito.

Mencionaré brevemente los 3 principales que aprendí que son:

- ECL (Expected Credit Loss)
- VaR (Value at Risk)
- ES (Expected Shortfall)

ECL: El Expected Credit Loss está dado por la fórmula PD * EAD * LGD, donde:

- PD: Probability of Default, lo cual es probabilidad de incumplimiento por parte del cliente.
- EAD: Exposure at Default, lo cual es la cantidad monetaria que está pendiente de saldar al momento de un incumplimiento.
- LGD: Loss Given Default, lo cual en pocas palabras es la pérdida real (cuánto termino perdiendo) en porcentaje del crédito una vez que se hayan cobrado los bienes dejados en garantía del cliente.

Este indicador de riesgo nos dirá la cantidad estimada que se puede llegar a perder en un crédito o en un portafolio de créditos. También los ayuda a cobrar una tasa de interés adecuada, es decir, que cubra con estas pérdidas esperadas calculadas.

VaR: El Credit VaR se refiere, en pocas palabras, a la máxima perdida esperada de acuerdo a un nivel de confianza (usualmente 99%).

ES: Es una medida similar al VaR, solamente un poco más pesimista, es decir, obtiene el valor esperado de las pérdidas mayores o iguales al VaR.

Para contestar esta pregunta consulté mis notas de una clase llamada "admin. de riesgos".

3. Supongamos que un banco quiere desarrollar un modelo de score que le indique qué personas que solicitan un crédito van a caer en impago y cuáles no. Dicho banco desarrolló dos modelos, el ordenamiento de cada modelo se muestra en las siguientes tablas:

Definiciones:

Campos tabla de odds	Definición
Rango Score	Los puntajes se dividen en intervalos del 10% de la población y se disponen en rangos de orden ascendente con los puntajes bajos en la parte superior y los altos en la inferior. Los puntajes altos indican un menor riesgo.
Personas	Muestra el número de expedientes (personas) en cada intervalo de puntajes.
Buenos	Muestra el número de expedientes (personas) que no cayeron en impago en cada intervalo de puntajes.
Malos	Muestra el número de expedientes (personas) que cayeron en impago en cada intervalo de puntajes.
Tasa de malos	Muestra el porcentaje de expedientes (personas) que cayeron en impago con respecto al total en cada intervalo de puntaje.
Malos acumulados	Muestra el porcentaje de expedientes (personas) que cayeron en impago con respecto al total en cada intervalo de puntajes mayores o iguales al del rango de puntaje considerado.

Modelo 1

	Personas	Buenos	Malos	Tasa de malos
Cartera	200,000	180,797	19,203	9.60%

Decil	Rango Score	Personas	Buenos	Malos	Tasa de Malos	Malos Acumulados
1	(300 , 445]	20000	15,673	4,327	21.64%	9.60%
2	(445 , 469]	20000	16,578	3,422	17.11%	7.44%
3	(469 , 492]	20000	17,034	2,966	14.83%	5.73%
4	(492 , 514]	20000	17,652	2,348	11.74%	4.24%

5	(514 , 539]	20000	17,999	2,001	10.01%	3.07%
6	(539 , 567]	20000	18,257	1,743	8.72%	2.07%
7	(567, 608]	20000	18,902	1,098	5.49%	1.20%
8	(608 , 649]	20000	19,302	698	3.49%	0.65%
9	(649 , 689]	20000	19,569	431	2.16%	0.30%
10	(689 , 818]	20000	19,834	169	0.85%	0.08%

Modelo 2:

	Personas	Buenos	Malos	Tasa de malos
Cartera	200,000	180,797	19,203	9.60%

Decil	Rango Score	Personas	Buenos	Malos	Tasa de Malos	Malos Acumulados
1	(300 , 445]	20,000	15,673	4,327	21.64%	9.91%
2	(445 , 469]	20,000	17,178	2,822	14.11%	7.74%
3	(469 , 492]	20,000	16,634	3,366	16.83%	6.33%
4	(492 , 514]	20,000	15,998	4,002	20.01%	4.65%
5	(514 , 539]	20,000	17,999	2,001	10.01%	2.65%
6	(539 , 567]	20,000	18,600	1,400	7.00%	1.65%
7	(567, 608]	20,000	19,600	400	2.00%	0.95%
8	(608 , 649]	20,000	19,200	800	4.00%	0.75%
9	(649 , 689]	20,000	19,569	431	2.16%	0.35%
10	(689 , 818]	20,000	19,734	266	1.33%	0.13%

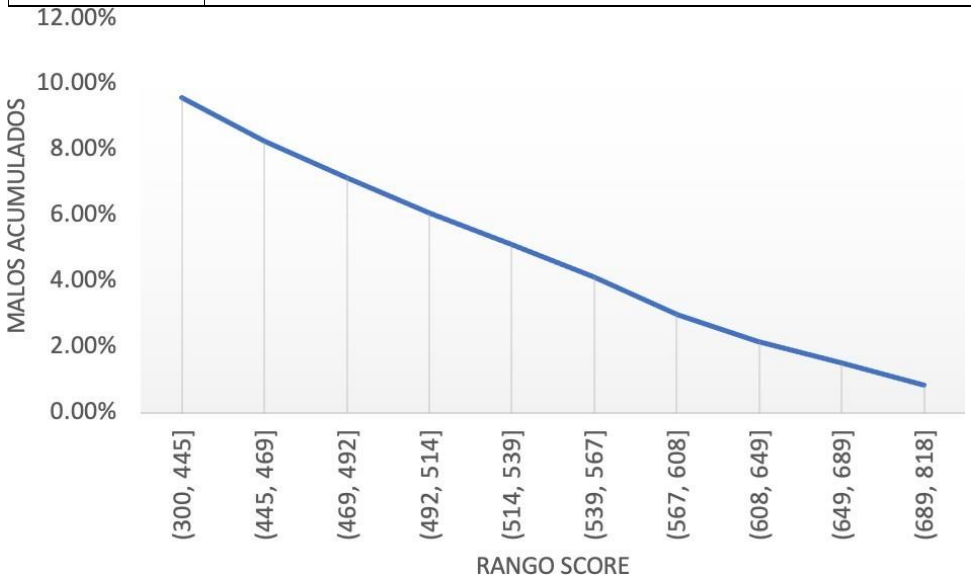
Justifica cuál de los dos modelos funciona correctamente.

Al ver estas tablas por primera vez no entendí bien lo que era o qué representaban, después me di cuenta que es un tipo de análisis de deciles lo cual he escuchado en algunas ocasiones. Investigué un poco y entendí lo más que pude. De esta manera llegué a la conclusión de que el modelo 1 es mejor o funciona correctamente.

Lo primero que se me ocurrió fue ver si la cantidad de malos de la tabla coincidía con el total de malos general, al analizar esto, me di cuenta que al sumar la cantidad de malos de la tabla del modelo 2 no da el total de malos, lo cual ya me indicaba que podría ser incorrecto. Posteriormente, analicé las tablas a más profundidad y llegué a la conclusión de que el modelo 1 es el correcto debido a la tasa de malos. Si la analizamos detenidamente, vemos que la tasa de malos de la tabla del modelo 1 disminuye constantemente en cada decil, a medida que avanza el decil la tasa de malos siempre disminuye, esto no sucede en la tabla del modelo 2, ya que en el decil 3 y 4 la tasa de malos aumenta con respecto a las anteriores. Esto indicaría que el modelo 1 es correcto ya que, a medida que aumenta el score crediticio, tiene sentido que la tasa de malos o impagos disminuya y no que aumente.

4. Considerando la gráfica que se muestra a continuación, ¿qué estrategia de aceptación puede implementar el banco si su apetito de riesgo es que no más del 5% de su cartera caiga en impago?

Concepto	Definición
Malos acumulados	Muestra el porcentaje de expedientes (personas) que cayeron en impago con respecto al total en cada intervalo de puntajes mayores o iguales al del rango de puntaje considerado.



Observando la gráfica, la estrategia a implementar por el banco sería aceptar créditos solamente a personas con un score superior a 514. Esto debido a que, más o menos en este rango, el porcentaje de malos acumulados cae por debajo del 5%, que equivale al apetito de riesgo del banco.

B. Conocimientos de Python y SQL

1. En un dataframe (df), ¿cuál es la sintaxis para obtener el total de na's para cada variable?

   nans = df.isna().sum()
   nans

   O también usando una comprensión de listas/diccionarios (más lento):

   {col: df[col].isna().sum() for col in df.columns}

2. En ese mismo df, ¿cuál es la sintaxis para imputar los valores perdidos na con el promedio? Supón que los valores perdidos se encuentran en la variable 'edad' y 'nivel_socieco'.

   La manera más sencilla es la siguiente, solamente que reemplazan los nan de todas las columnas con su respectivo promedio:

```
df.fillna(df.mean(), inplace = True)
df
```

Si solamente queremos reemplazar los nan de las variables especificadas, se puede hacer así:

```
df[['edad', 'nivel_socieco']] = df[['edad', 'nivel_socieco']].fillna(df[['edad',
'nivel_socieco']].mean(), inplace=True)
df
```

De igual manera, se puede hacer con una comprensión de listas de la siguiente manera (más lento por ciclo for):

```
[df[col].fillna(df[col].mean(), inplace=True) for col in df.columns];
df
```

Si queremos solamente ciertas columnas, hay que modificar el ciclo for:

```
[df[col].fillna(df[col].mean(), inplace=True) for col in ['edad', 'nivel_socieco']];
df
```

3. Crea un histograma a través de la librería seaborn para la variable edad.

```
plt.figure(figsize=(10, 6))
sns.histplot(data = df, x = "edad", bins = 5)
plt.title("Distribución de Edad", fontsize=12)
plt.xlabel("Edad")
plt.ylabel("Frecuencia")
plt.show()
```

4. Estás a punto de preparar un modelo y para ello quieres separar en entrenamiento y validación. ¿Con qué sintaxis separarías tu base en un 70% para entrenamiento y el restante para validación? ¿qué librería usarías? Esto con una semilla aleatoria.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.3,
random_state=4)
```

5. Si quisieras implementar un Random Forest 10 árboles y una máxima profundidad de 2. ¿Cuál sería la sintaxis? Puedes apoyarte de lo sig:

```
# Librería a importar
from sklearn import ….

# Inicializar el modelo
rf = …..(n_estimators = …..)
```

```
# Entrenar el modelo en tu base de entrenamiento: x_train, y_train
rf.fit(…. , ….)
```

```
from sklearn.ensemble import RandomForestClassifier

forest = RandomForestClassifier(n_estimators=10, max_depth = 2)

forest.fit(X_train, y_train)
```

6. Si quisieras verificar algunas métricas sobre tu modelo de Random Forest, ¿qué librería y sintaxis usarías para graficar la curva ROC? Supón que obtienes un valor de 0.53 de AUC, ¿este valor te diría que tu modelo es preciso?

```
from sklearn import metrics

y_pred_proba = forest.predict_proba(X_test)[:,1]
fpr, tpr, _ = metrics.roc_curve(y_test, y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, label = "AUC = " + str(round(auc, 4)))
plt.plot([0,1], [0,1], 'r--')
plt.legend(loc = 'best')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

En este caso, al tener un AUC de 0.53, de acuerdo con la teoría, debido a que este valor se encuentra dentro del rango de 0.5 y 1, podríamos inferir que el modelo tiene un desempeño aceptable. Sin embargo, al estar tan cerca del valor de 0.5, quiere decir que este desempeño está muy cercano al azar, por lo que la mejor opción sería mejorar el modelo.

7. Supón que tienes la sig información en las tablas uno y dos:

Tabla uno		
Año	Trim	PrecioMonto
2012	3	1200
2013	4	4000
2014	1	100

Tabla dos		
Año	Trim	Monto
2012	3	3.2
2013	4	5.6
2014	1	6.7

Escribe código sql para hacer un inner join de las tablas y seleccionar la suma de monto * precio por año.

```
SELECT
   u.ano,
   u.preciomonto * d.monto
FROM tabla_uno u JOIN tabla_dos d
ON u.ano = d.ano;
```

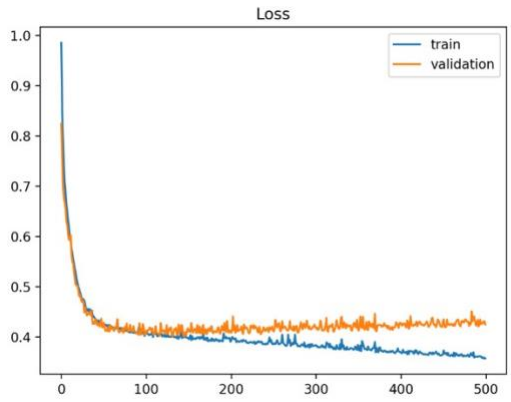
Lo anterior funciona ya que ningún año se repite, en caso de que se repitiera (y esto sería lo más adecuado):

```
SELECT
    u.ano,
    SUM(u.preciomonto * d.monto) AS total
FROM tabla_uno u JOIN tabla_dos d
ON u.ano = d.ano
GROUP BY u.ano;
```

Consulté las notas del curso que tomé para recordar este código

C. Conocimientos de Ciencia de Datos

1. ¿Qué significan las siguientes curvas de aprendizaje? Justifica tu respuesta.



a) El modelo está sobre ajustado (overfit).
b) El modelo está sub ajustado (underfit).
c) El modelo está bien entrenado.
d) Ninguna de las anteriores.

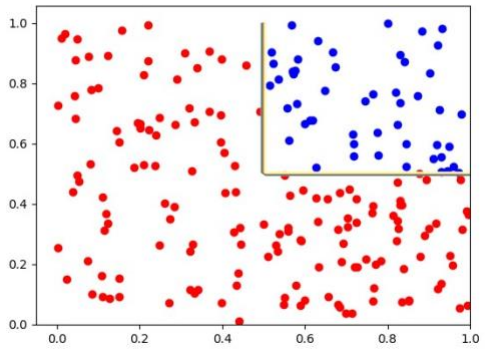
A manera general, las siguientes curvas de aprendizaje se pueden interpretar como, mientras menos errores tenga/haga el modelo, menor será la pérdida. Viendo la gráfica, vemos que no hay underfitting ya que la pérdida del entrenamiento es baja. Tampoco diría que el modelo está bien entrenado, ya que la curva de validación parecería presentar un pequeño problema.

Al avanzar en las épocas del modelo, vemos que la pérdida del entrenamiento sigue disminuyendo, mientras que la pérdida de la validación parecería tener una ligera tendencia a aumentar. Por lo tanto, el modelo parece tener un ligero problema de overfitting y, siguiendo la tendencia, si aumentamos las épocas este problema se vería más claramente en la gráfica.

2. Supón que tu modelo está sobre ajustado. ¿Qué harías para mejorar eso?

   La primera opción que consideraría, la cual aplicaría un poco para el caso anterior, sería utilizar un early stopping para detener el entrenamiento de los datos cuando el rendimiento del modelo empieza a empeorar causando un sobreajuste, por ejemplo, en la gráfica anterior se podría detener el modelo alrededor de la época 100 para tener un menor sobreajuste. Además, otra técnica que se podría utilizar es la de revisar las características del modelo para ver que sean relevantes y no redundantes, viendo la importancia de las variables (en un modelo de random forest por ejemplo), haciendo un análisis de correlaciones o utilizando técnicas como PCA por ejemplo. Existen otros métodos y técnicas como una validación cruzada, simplificar el modelo, etc., que también se pueden utilizar para resolver este problema.

3. ¿Qué tipo de algoritmo realiza una separación como la que se muestra en la figura?



a) Máquina de Soporte Vectorial con Kernel Lineal
b) Regresión Logística
c) Árbol de Decisión
d) Random Forest

4. ¿Qué es el aprendizaje no supervisado? Menciona un ejemplo.

   En mis palabras, aprendizaje no supervisado se refiere a cuando no tenemos una variable de salida/objetivo (y), es decir, no se conoce el resultado de la clasificación. Por ejemplo, en el caso de un problema de fraude, se tienen las variables de entrada, sin embargo no se conoce el resultado de estas, es decir, no se conoce si la transacción fue fraude o no. Esto quiere decir que el modelo lo determinará por sí solo (sin conocer el resultado) tratando de identificar patrones o anomalías.

5. Si se te pidiera elaborar un modelo de machine learning de clasificación para un problema de fraude que tiene una bad rate del 2.0%, ¿con qué tipo de problema te podrías estar enfrentando? ¿cómo lo solucionarías? ¿qué tipo de modelo de ML usarías?

   Primero que nada, estamos lidiando con un problema de desbalance de datos, teniendo muchas más instancias de una clase (no fraudes) que de otra clase minoritaria (fraudes). Esto supone un problema al crear un modelo sin resolver este problema, ya que podríamos no detectar la clase minoritaria. Por ejemplo, si usamos la métrica de rendimiento “accuracy”, la cual mide el porcentaje de aciertos, y resulta ser casi perfecta, estos aciertos podrían ser más que nada de transacciones no fraudulentas (la clase mayoritaria). Esto quiere decir que, en efecto, tenemos un accuracy alto, pero el

modelo podría estar únicamente prediciendo operaciones no fraudulentas, siendo inútil al predecir/detectar una operación fraudulenta.

En cuanto a soluciones hay varios métodos como lo pueden ser over-sampling y under-sampling. Estas dos son las opciones más sencillas y las que más he utilizado, sin embargo, ambas tienen sus ventajas y desventajas como podrían ser crear datos artificiales en el caso de over-sampling, así como la pérdida de información en under-sampling. Existe otra alternativa llamada SMOTE la cual, en pocas palabras, es una combinación de ambas, sin embargo no tengo tanta experiencia usando este método.

Finalmente, el tipo de modelo que usaría dependería un poco de si solucioné el desbalance con algunos de los métodos anteriormente mencionados o no. En caso de que no, utilizaría algún modelo que sea mejor en detectar/manejar el desbalance de los datos como lo podría ser Random Forest o XGBoost, esto también podría ser parte de la solución. Ahora en caso de que haya aplicado alguno de los métodos anteriormente mencionados (sampling), podría utilizar los mismos modelos o una regresión logística ya que su costo computacional es menor, tarda menos, es fácil de interpretar y muchas veces obtiene resultados similares a otros modelos más complejos.

6. ¿Cuál consideras que es una buena métrica de performance para un problema desbalanceado, por decir, el 60% de AUCROC es una métrica buena o mala y porqué?

   Primero que nada, la métrica que no consideraría buena por ser engañosa y por lo anteriormente explicado (detectar únicamente instancias de la clase mayoritaria) es el accuracy. Un 60% de AUCROC es aceptable, teniendo en cuenta que está dentro de dicho rango aceptable, además, si no se lidió con el problema del desbalance de datos y se creó el modelo con los datos desbalanceados, un 60% de AUCROC no estaría tan mal debido a la complejidad de los datos/del problema del desbalance. Sin embargo, un 60% es relativamente bajo, por lo que de todas maneras buscaría intentar mejorar el modelo.

   Además, no solamente consideraría el AUCROC para evaluar el modelo, lo complementaría con otras métricas tales como “Precisión”, “Recall” y “F1 Score”, las cuales ayudarían a complementar y entender el performance del modelo de manera más detallada y evaluar si se requiere mejorar.

D. Problema en Python:

La empresa necesita realizar un modelo de fraude. Tu papel como DS es dar una solución de modelo y mitigar los riesgos de fraude basada en datos. Los datos para realizar el modelo son datos_fraud.csv .

1. Una vez que obtengas los datos, utiliza todos tus conocimientos y todos los pasos que creas necesarios para poder entregar un modelo funcional para predecir si un cliente debería de ser aprobado o no. Entre los pasos que esperamos ver están:

    - EDA
    - Pre-procesamiento de los datos
    - Entrenamiento del modelo
    - Testing de modelo
    - Una explicación de cómo pondrías este modelo en producción y que tendrías que estarle cuidando con el tiempo
2. Una vez entrenado tu modelo esperamos recibir 3 archivos específicos:
    - Un Jupyter Notebook que explique todo su proceso de entrenamiento. Aquí mismo es donde vas a incluir los distintos pasos descritos arriba bien documentados para poder entender cómo fuiste generando el modelo.
    - CSV de predicciones de tu modelo en testing data. El CSV nada más debe de incluir el ID de solicitante y su score del modelo.
    - Un PDF explicando qué punto de corte seleccionarías y por qué.

Recuerda que en un contexto de trabajo en equipo, las personas que leerán tu código puede que no estuvieron involucradas en su desarrollo pero igual tendrán que entendenderlo y/o mantenerlo.

Datos: **id:** Identificador único de la transacción
timestamp: momento en el que ocurre la transacción **amount:** monto de la transacción
variables_01 a variable_32: features de la trasacción
fraud: Flag de flaude 1 es transacción fraudulente y 0 no lo es