PRUEBA
CIENTÍFICO DE DATOS

Prueba de habilidades y aptitudes Científico de datos Jr.

círculo de crédito

Dirección de Analítica
Círculo de crédito

círculo de crédito

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?
   - Solicitud: El cliente presenta una solicitud de crédito a la entidad
   - Evaluación de crédito: La entidad se encarga de realizar un análisis para verificar si el cliente es viable para para el crédito, a partir de revisar el historial crediticio de este. Además de la realización de un estudio de ingresos y egresos del solicitante para verificar su capacidad de pago.
   - Aprobación: Después de haber decidido la aprobación del crédito, se acuerdan entre las dos partes las condiciones y se firma un contrato.
   - Desembolso: La entidad desembolsa al solicitante la cantidad pactada.
   - Monitoreo y administración: Se realiza una revisión de los pagos en tiempo y en forma para detectar en su caso la mora.
   - Cierre y liquidación
   - Evaluación
2. Menciona algún indicador de riesgo de crédito.

   La más común es mediante el *scoring* que es un conjunto de modelos estadísticos que pronostican si se pagara o no el crédito a partir del historial crediticio.

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.

círculo de crédito

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%

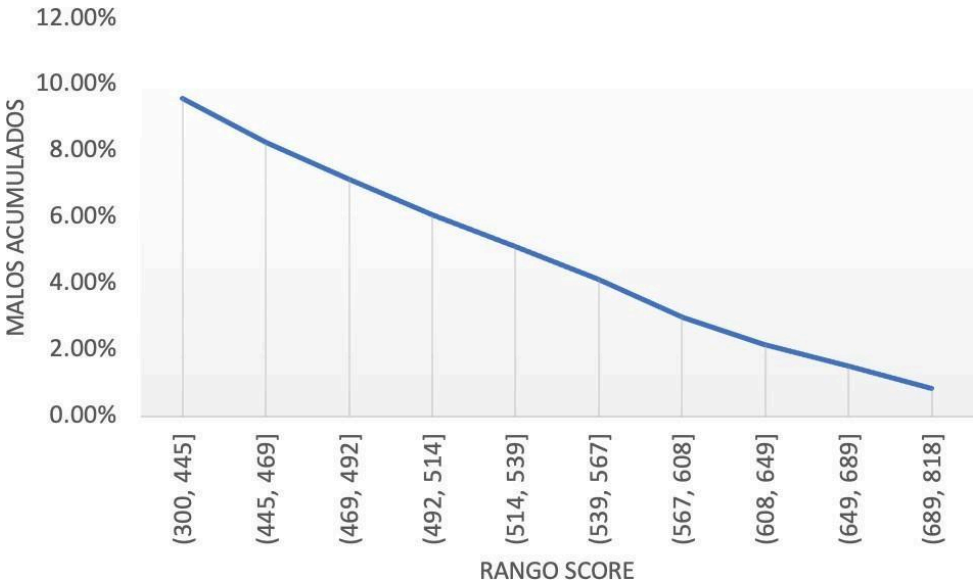
Decil	Rango Score	Personas	Buenos	Malos	Tasa de Malos	Malos Acumulados
1	(300 , 445]	20,000	15,673	4,327	21.64%	9.91%
2	(445 , 469]	20,000	17,178	2,822	14.11%	7.74%
3	(469 , 492]	20,000	16,634	3,366	16.83%	6.33%
4	(492 , 514]	20,000	15,998	4,002	20.01%	4.65%
5	(514 , 539]	20,000	17,999	2,001	10.01%	2.65%
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.

El primer modelo es el que funciona correctamente, ya que aunque los dos cuentan con medidas correctas en el campo de Tasa de Malos, el modelo segundo en el campo de Malos acumulados comete error al calcular y en consecuencia el monto general de este se ve errado.

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.



Considerar todos aquellos clientes que se encuentren en los rangos cuyo porcentaje de malos acumulados no rebase el cinco por ciento, es decir, hasta el rango (539,567).

Así, el porcentaje de posibles deudores acumulados, es decir, aquellos se veran en posibilidad de no pagar en todos los intervalos será de máximo 4.24% aproximadamente como lo indica el modelo.

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?

   Para la módulo pandas de python:

   ```
   na_totales_por_variable = df.isna().sum()
   ```

   Para pySpark:

   ```
   na_totales_df = df.select([count(when(isnan(c), c)).alias(c) for c in df.columns]).show()
   ```

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'.

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

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

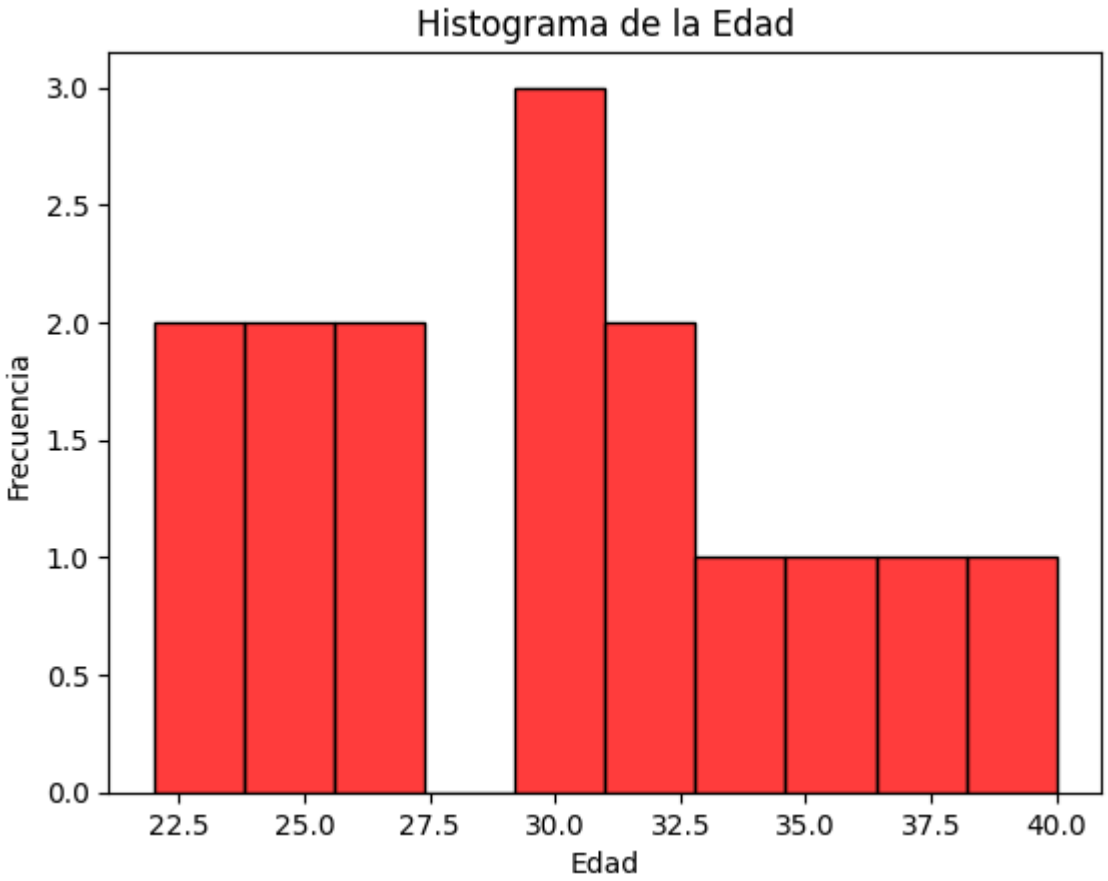
También en pySpark:

```
# Calcular el promedio de las columnas 'edad' y 'nivel_socieco'
mean_vals = df.select(mean(col('edad')).alias('mean_edad'), mean(col('nivel_socieco')).alias('mean_nivel_socieco')).collect()

# Extraer los valores medios
mean_edad = mean_vals[0]['mean_edad']
mean_nivel_socieco = mean_vals[0]['mean_nivel_socieco']
# Imputar los valores perdidos con el promedio
df = df.fillna({'edad': mean_edad, 'nivel_socieco': mean_nivel_socieco})
```

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

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns


# CCreamos un dataframe de prueba para graficar
data = {
    'id': range(1, 16),
    'edad': [25, 30, 22, 35, np.nan, 40, 23, 27, 31, np.nan, 33, 26, 24, 38, 32]
}

df = pd.DataFrame(data)

# Imputamos los valores nulos
df['edad'].fillna(df['edad'].mean(), inplace=True)

# Creamos el histograma usando seaborn
sns.histplot(df['edad'], kde=False, bins=10, color='red')
#Los ejes x y y correspondel a la edad y a la frecuencia de aparicion de esta.
plt.xlabel('Edad')
plt.ylabel('Frecuencia')
plt.title('Histograma de la Edad')
plt.show()
```

Output:



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.

```
# Dividir los datos en conjuntos de entrenamiento y prueba.
train_df, test_df = df.randomSplit([0.7, 0.3]
```

se usa pyspark

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

```
#Importamos los modulos necesarios
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split

#es un dataset de prueba
from sklearn.datasets import load_iris

# Cargar un conjunto de datos de ejemplo (en este caso, el conjunto de
datos Iris)
data = load_iris()
X_set = data.data
y_set = data.target

# Dividir los datos en conjuntos de entrenamiento y prueba
```

```
#0.7 para el entrenamiento y 0.3 para la prueba
x_train, x_test, y_train, y_test = train_test_split(X_set, y_set,
test_size=0.3, random_state=42)

# Inicializar el modelo de Random Forest con 10 arboles y profundidad 2
rf = RandomForestClassifier(n_estimators=10, max_depth=2, random_state=42)

# Entrenar el modelo
rf.fit(x_train, y_train)
accuracy = rf.score(x_test, y_test)
print(f'Accuracy: {accuracy}')
```

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?

   Podría usarse scikit-learn para una implementación más directa:

```
roc_auc_score(Y_test, model.predict(xtest))
```

   Podría usar pySpark:

```
# ROC AUC Score
evaluator = BinaryClassificationEvaluator(labelCol="is_fraud",
rawPredictionCol="rawPrediction", metricName="areaUnderROC")
roc_auc = evaluator.evaluate(predictions)
```

   Recordemos que la curva ROC (Receiving Operating Characteristic) es una herramienta que nos ayuda a medir la relación entre la sensibilidad y la especificidad (detectar verdaderos y falsos positivos en un problema de clasificación). Asociada típicamente a que valores cercanos al 50% serán casos cuando el modelo detecta en casi la misma proporción verdaderos y falsos negativos, lo que implicaría directamente que el modelo no está detectando correctamente ningún patrón para la calcificación, y en consecuencia carece de correcta funcionalidad.

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

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

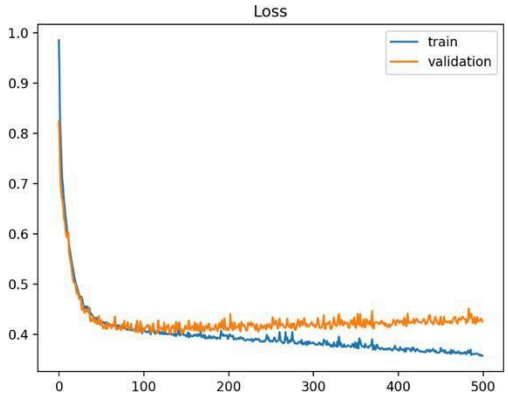
Tabla dos		
Año	Trim	Monto
2012	3	3.2
2013	4	5.6
2014	1	6.7

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

   SELECT SUM(MONTO*PRECIO) FROM TABLA UNO AS T1 INNER JOIN TABLA DOS AS T2 ON T1.AÑO = T2.AÑO AND T1.TRIM = T2.TRIM

C. Conocimientos de Ciencia de Datos

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



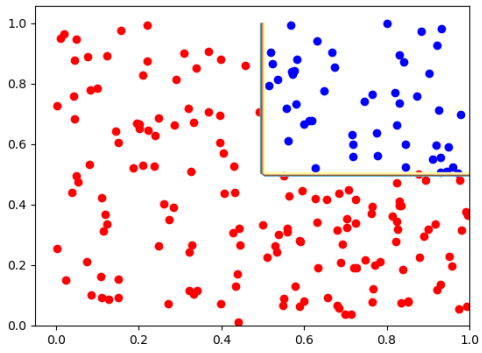
a) El modelo está sobre ajustado (overfit).
b) El modelo está sub ajustado (underfit).
c) El modelo está bien entrenado.
d) Ninguna de las anteriores.

El modelo está bien entrenado pues la curva de validación se ajusta lo suficiente a los datos de entrenamiento.

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

   Podar el conjunto de características, de manera que el modelo consuma solo aquellas que representen utilidad para la salida. Además de aumentar la cantidad de datos que el modelo consume para añadir variedad.

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.

   Se trata de una serie de técnicas de aprendizaje automatizado en la que los modelos intentan aprender a partir de patrones de los datos que reciben y no directamente de etiquetas o categorías predefinidas dadas a cada muestra.

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?

   En este contexto, el bad rate se refiere a la cantidad de fallos o predicciones incorrectas entre la cantidad de predicciones totales. En este sentido, puede referirse a muchos factores, entre estos el modelo no está clasificando bien la cantidad suficiente de muestras en el conjunto de pruebas siendo provocado por que el modelo no esté bien ajustado o los datos de entrenamiento no sean suficientes o no estén correctamente balanceados (exista mayor cantidad de muestras asociadas a una clase o etiqueta).

   Al realizar un submuestreo de todos aquellos registros de la clase dominante para balancear el dataset y escoger un modelo correcto. En este caso optaría por un árbol de decisión o un random forest para mejorar la robustez del modelo y por ende de la predicción.

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

   AUC ROC nos proporciona una visión completa y resumida del rendimiento del modelo. Que esta medida sea de 60% significa que el modelo es un poco mejor que la aleatoriedad (50%), hecho que nos hace pensar que el modelo funciona, sin embargo es aun bajo. Por tanto, es mejor optar por un 90% para este tipo de medida para asegurarnos que el modelo funciona de manera correcta.

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

La implementación del modelo se puede ver en el archivo *model_fraud.ipynb*

Selección de punto de corte e implementación en producción

Para la implementación del problema, seleccionaremos un punto de corte (threshold) de 0.5 basado en el equilibrio entre precisión y recall observados en la curva ROC y el reporte de clasificación. Pues es un buen balance para maximizar la detección de fraudes y para minimizar los falsos positivos. Es decir, tratamos de encontrar el punto de cruce para que ninguna de estas dos características predomine en la predicción del modelo.

Para implementar y monitorear este modelo en producción, continuaremos evaluando el rendimiento del modelo y ajustando el umbral según sea necesario para adaptarnos a cambios en los patrones de fraude. También seria recomendable realizar experimentación con diversos hyperparametros integrando nuevos datos al conjunto de entrenamiento del modelo.