1. EDA – Exploración y evaluación de datos

a. Descripción inicial

- **Volumen total**: ~10 millones de transacciones
- Variables clave: `user_id`, `transaction_date`, `transaction_amount`, `account_number`, `subsidiary`, `transaction_type`
- Rango de fechas: incluye múltiples días y horas
- Alta variabilidad en montos (media ≈ 191, mediana ≈ 107)

b. Patrones observados

- Gran parte de los usuarios realiza **1 sola transacción**
- La distribución de montos está **sesgada a la derecha**
- La mayoría de las transacciones pequeñas (≤ $250) ocurren entre 8:00 a.m. y 6:00 p.m.
- Pocos usuarios acumulan muchas transacciones en un mismo día

c. Hipótesis

"Los usuarios que realizan **múltiples transacciones pequeñas en un día**, dirigidas a pocas cuentas, podrían estar fraccionando montos intencionalmente."

Esto motivó el uso de **ventanas móviles por usuario**, para evaluar patrones por período móvil de 24 horas. Además, que, al analizar la distribución de las transacciones por hora, se puede observar que la mayoría se realizan entre las 8 y las 6 pm como se indicó por lo cual al dejar ventanas diarias se tiene en cuenta todos los posibles casos de fraccionamiento. De Ser menos tiempo de ventana se tendría el mismo resultado mientras se capture las horas pico. Y de ser mayor se podría estar clasificando fraccionamiento casos que no lo son.

2. Modelo Analítico

a. Flujo de datos

1. **Carga de datos con Polars**
2. **Rolling window de 24 horas por usuario**
    - Se computan: número de transacciones, monto total y número de cuentas distintas en cada ventana
3. **Etiquetado supervisado (`sospechoso_24h`)**
    - Regla final:

```
sospechoso_24h = (
    (n_tx_24h >= 2) &
    (total_24h >= 100) &
    (n_cuentas_distintas_24h <= 2)
)
```

4. **Ingeniería de características**
    - Variables agregadas: `n_tx_total`, `small_tx_ratio`, `std_monto_usuario`, `tx_same_hour_flag`, etc.
5. **Modelado supervisado**
    - Algoritmo: `XGBoostClassifier`
    - Balanceo de clases con `SMOTE`
    - Validación cruzada estratificada (10 folds)

b. Evaluación del modelo

- **Métrica principal**: F1-score para clase positiva (sospechosa)
- **Resultados promedio (CV):**
    - F1-score ≈ 0.955
    - Precision ≈ 0.951
    - Recall ≈ 0.959
    - ROC AUC ≈ 0.999

c. Aplicación sobre muestra representativa

- Muestra de 150.000 transacciones (estratificada)
- Resultado:
    - **310 usuarios** fueron clasificados como sospechosos
    - **314 transacciones** identificadas como fraccionadas

4. Frecuencia de actualización recomendada

Se recomienda aplicar este modelo **diariamente**, al finalizar cada jornada, para analizar las transacciones del último día en ventanas móviles.

Ventajas:

- Permite detección casi en tiempo real
- Puede ser usado para alertas tempranas o revisión operativa

5. Arquitectura sugerida (opcional)

- **Ingesta**: conexión a base transaccional (BigQuery, Redshift, etc.)
- **Preprocesamiento**: script en Python con Polars (escala bien)
- **Modelado**: `.pkl` en servidor local o contenedor Docker
- **Despliegue**: batch diario o pipeline en Airflow
- **Almacenamiento de resultados**: tabla resumen de usuarios sospechosos