Exploración y visualización de datos en Python

Ejercicio de Aplicación Autónoma: Análisis Detallado de Transacciones de E-commerce

Resultado de Aprendizaje de la Semana: Desarrolla habilidades y competencias para ser un experto en el uso de librerías Python en procesos de análisis exploratorio, limpieza, procesamiento e ingeniería de datos.

Objetivo y alcance del trabajo

El objetivo de este componente práctico es explorar brevemente y familiarizarse con el entorno de Python para manipulación y visualización de datos. Para esto, se recomienda utilizar Google Colab que permite escribir y ejecutar código Python en la nube, disponible mediante el navegador web, e integrarlo con bloques de texto en formato Markdown para documentación complementaria. Se debe presentar un solo documento tipo notebook en formato “ipynb” y adjuntar los archivos que se soliciten en el desarrollo. El nombre del archivo a entregar debe tener el siguiente formato: “AMGD_CP_S1_G#”, en donde ‘#’ es el número de grupo.

Contexto del Problema:

Eres un analista de datos junior en una empresa de E-commerce. Tu equipo necesita comprender mejor el comportamiento de compra de los clientes para optimizar las estrategias de marketing y el inventario. Se te ha asignado la tarea de realizar un análisis exploratorio de un dataset de transacciones para identificar patrones clave, limpiar los datos y preparar algunas métricas importantes.

Dataset Sugerido:

Un dataset simulado, aquí te describo el tipo de columnas y desafíos que podría tener:

- **ID_Transaccion**: Identificador único de cada transacción.
- **ID_Cliente**: Identificador único del cliente.
- **Fecha_Transaccion**: Fecha y hora en que se realizó la compra (puede venir como string "yyyy-mm-dd HH:MM:SS" o "dd/mm/yyyy").
- **Producto_SKU**: SKU (Stock Keeping Unit) del producto (ej. "PROD123", "ITEM456").
- **Nombre_Producto**: Descripción del producto (ej. "Auriculares Bluetooth", "Camiseta Algodón").
- **Cantidad**: Número de unidades del producto compradas en esa transacción.
- **Precio_Unitario**: Precio de una unidad del producto.
- **Valor_Total_Transaccion**: Precio total de la línea de producto (Cantidad * Precio_Unitario). (Puede ser que esta columna **falte** y debas calcularla).

- **Categoria_Producto**: Categoría a la que pertenece el producto (ej. "Electrónica", "Ropa", "Hogar").
- **Metodo_Pago**: ("Tarjeta", "PayPal", "Efectivo", etc.).
- **Ciudad_Envio**: Ciudad de destino del pedido. (Puede contener valores nulos o errores tipográficos).
- **Calificacion_Producto**: Calificación que el cliente dio al producto (ej. 1 a 5 estrellas). (Puede contener muchos valores nulos).

Desafíos y Dificultades Potenciales del Datos (para que el estudiante los identifique y resuelva):

1. **Valores Nulos:** Especialmente en Ciudad_Envio y Calificacion_Producto. ¿Cómo manejarlos (eliminar, imputar, etc.)?
2. **Tipos de Datos Incorrectos:** Fecha_Transaccion como object (string) en lugar de datetime. Cantidad o Precio_Unitario como object por caracteres inapropiados.
3. **Outliers/Valores Anómalos:** Cantidad o Precio_Unitario con valores cero, negativos o irrealmente altos.
4. **Inconsistencia en Cadenas de Texto:** Múltiples formas de escribir la misma Ciudad_Envio (ej. "Madrid", "madrid", "MAD").
5. **Cálculo de Nuevas Columnas (Ingeniería de Datos Básica):** Si Valor_Total_Transaccion no existe, calcularla. Extraer año, mes, día de la semana de Fecha_Transaccion.

Tarea: Desarrolla un Jupyter Notebook que realice un comprehensive Análisis Exploratorio de Datos (EDA) y preprocesamiento del dataset de transacciones de E-commerce.

Pasos Detallados (Guía, NO un script paso a paso):

1. **Generación del Dataset:**
    - Ejecuta el notebook base
2. **Exploración Inicial de Datos:**
    - Muestra las primeras filas del DataFrame (head()).
    - Obtén un resumen conciso del DataFrame, incluyendo tipos de datos y valores no nulos (info()).
    - Calcula estadísticas descriptivas para las columnas numéricas (describe()).
    - Identifica y cuantifica la presencia de valores nulos por columna (isna().sum()).

- Explora la cantidad de valores únicos y las categorías más frecuentes para las columnas categóricas.

3. **Limpieza y Preprocesamiento de Datos:**
   - **Manejo de Nulos:** Implementa una estrategia para gestionar los valores nulos (e.g., rellenar con la media/mediana, la moda, o eliminar filas/columnas si la cantidad de nulos es significativa y justificada). Justifica tu elección.
   - **Conversión de Tipos:** Asegúrate de que Fecha_Transaccion sea de tipo datetime. Convierte otras columnas a tipos correctos si es necesario.
   - **Validación de Datos Numéricos:** Identifica y, si es apropiado, gestiona valores anómalos o atípicos en Cantidad y Precio_Unitario (o al menos documenta su presencia).
   - **Normalización de Texto (Opcional, pero valorado):** Si hay inconsistencias en columnas de texto como Ciudad_Envio, aplica técnicas básicas para estandarizarlas (e.g., convertir a minúsculas, arreglar errores tipográficos simples).
4. **Ingeniería de Características (Básica):**
   - Crea una nueva columna Valor_Total_Transaccion si no existe, calculándola como Cantidad * Precio_Unitario.
   - Extrae el Año, Mes, y Día_Semana de la columna Fecha_Transaccion y crea nuevas columnas para cada uno.
5. **Análisis y Agregación de Datos:**
   - Calcula:
     - Las ventas totales por Categoria_Producto.
     - El número de transacciones por Metodo_Pago.
     - Los TOP 10 productos más vendidos (por Cantidad o Valor_Total_Transaccion).
     - El número de transacciones y el valor total por mes (utilizando la columna Mes creada).
     - La calificación promedio por Categoria_Producto.
   - Utiliza groupby() y agg() de Pandas para realizar estas agregaciones.

6. **Visualización de Datos:**

- Crea al menos **cinco visualizaciones** significativas utilizando Matplotlib y/or Seaborn que respondan a las preguntas planteadas en el paso 5. Algunos ejemplos:
  - Un gráfico de barras de ventas totales por categoría de producto.
  - Un gráfico de líneas de las ventas totales a lo largo del tiempo (por mes o año).
  - Un histograma de la distribución de Precio_Unitario o Cantidad.
  - Un boxplot de Valor_Total_Transaccion por Metodo_Pago.
  - Un scatter plot que relacione Cantidad y Calificacion_Producto (si hay suficientes datos).
- Asegúrate de que cada gráfico tenga: un título claro, etiquetas de los ejes (xlabel, ylabel), y una leyenda si es necesario.
- Utiliza el estilo de Matplotlib orientado a objetos (fig, ax = plt.subplots()).
- Guarda al menos una de las figuras generadas en un archivo de imagen (plt.savefig()).

7. **Conclusiones y Reflexión:**
   - En una celda de Markdown al final del notebook, escribe un breve resumen de tus hallazgos más interesantes.
   - Reflexiona sobre los desafíos encontrados durante la limpieza y el preprocesamiento de los datos.
   - Menciona las preguntas adicionales que te gustaría explorar con este dataset si tuvieras más tiempo o herramientas.

Criterios de Evaluación:

- **Funcionalidad:** El código se ejecuta sin errores y produce los resultados esperados.
- **Claridad del Código:** Código bien estructurado, legible y con comentarios adecuados donde sea necesario.
- **Calidad del Análisis:** Se realizan los pasos de EDA y preprocesamiento de manera efectiva, abordando los desafíos del dataset.
- **Calidad de las Visualizaciones:** Gráficos informativos, claros, bien etiquetados y visualmente atractivos.
- **Pensamiento Crítico:** Las conclusiones y reflexiones demuestran una comprensión profunda del proceso y de los datos.
- **Uso de Librerías:** Correcta aplicación de Python (fundamentos), NumPy (si es el caso), Pandas, Matplotlib y Seaborn.

- **Autonomía:** Demuestra capacidad para investigar y resolver problemas de forma independiente.

Entrega: Un único archivo Jupyter Notebook (.ipynb).

¿Por qué este ejercicio tiene una dificultad mayor?

- **Realismo:** Simula un escenario de datos "desordenados" que requieren decisiones sobre valores nulos, tipos de datos y outliers.
- **Autonomía:** Fomenta la búsqueda de soluciones a problemas no directamente "recetados" en las clases (ej. cómo manejar un tipo de outlier específico, cómo estandarizar texto).
- **Ingeniería de Características:** Requiere la creación de nuevas columnas a partir de las existentes (Valor_Total_Transaccion, Mes, Día_Semana), lo que es un primer paso en la ingeniería de características.
- **Análisis Multidimensional:** Implica agregar y visualizar datos a través de múltiples dimensiones (ej. ventas por categoría y mes).
- **Integración de Librerías:** Los estudiantes deben combinar de manera fluida Python base, Pandas (para manipulación y agregación) y Matplotlib/Seaborn (para visualización).
- **Reflexión Crítica:** La sección de conclusiones pide un análisis no solo de los resultados, sino del *proceso*, un aspecto clave para un futuro científico de datos.