EJERCICIO PRÁCTICO DE MINERÍA DE DATOS

1. Definir el Problema y Objetivos

Objetivo:
Decide qué quieres descubrir o predecir. Por ejemplo, podrías utilizar un dataset de estudiantes para predecir si aprobarán o no un examen, o analizar factores que influyen en sus calificaciones.

Ejemplo de enunciado:
"Determinar cuáles variables (horas de estudio, asistencia, actividades extracurriculares, etc.) influyen más en la probabilidad de aprobar un examen."

---

2. Búsqueda y Adquisición de Datos

Dónde buscar datos:

- **Kaggle:** Plataforma con datasets en diversas áreas.
  Ejemplo: Busca "student performance" en Kaggle.
- **UCI Machine Learning Repository:** Tiene conjuntos de datos clásicos y bien documentados.
- **Datos abiertos de gobiernos o instituciones:** Muchos países ofrecen portales de datos abiertos.

Acción práctica:
Descarga un dataset, por ejemplo, el dataset "Student Performance" disponible en Kaggle o UCI. Asegúrate de que el archivo esté en formato CSV, Excel u otro formato legible.

---

3. Preparar el Entorno de Trabajo

Instalar Python y librerías necesarias:
Puedes usar Anaconda para tener todo el entorno de ciencia de datos. Las librerías comunes son:

- **Pandas:** Para manipulación de datos.
- **NumPy:** Para operaciones numéricas.
- **Matplotlib / Seaborn:** Para visualización.
- **scikit-learn:** Para modelado y evaluación.

Ejemplo de instalación (desde la terminal):

```
pip install pandas numpy matplotlib seaborn scikit-learn
```

Configuración en un Jupyter Notebook:
Crea un nuevo Notebook y carga las librerías:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

---

4. Exploración de Datos (Data Exploration)

Carga de datos:
Utiliza Pandas para leer el archivo CSV:

```
df = pd.read_csv("ruta/al/archivo/student_performance.csv")
df.head()  # Visualiza las primeras filas
```

Análisis descriptivo:
Obtén información general:

```
df.info()
df.describe()
```

Visualización inicial:

- **Distribuciones:** Usa histogramas para ver la distribución de variables numéricas.
- **Relaciones:** Usa gráficos de dispersión o boxplots para ver relaciones entre variables.

```
sns.histplot(df['horas_estudio'])
plt.title("Distribución de Horas de Estudio")
plt.show()

sns.boxplot(x='aprobado', y='horas_estudio', data=df)
plt.title("Horas de estudio según aprobación")
plt.show()
```

---

5. Preprocesamiento de Datos

Limpieza:

- Revisa valores nulos o inconsistentes.
- Ejemplo:

  ```
  df.isnull().sum()
  df.dropna(inplace=True)  # O aplicar otras técnicas de imputación
  ```

Transformación:

- Convierte variables categóricas a numéricas (codificación).

```
df['genero'] = df['genero'].map({'M': 0, 'F': 1})
```

- Normaliza o estandariza variables si es necesario.

Selección de características:

- Escoge las columnas relevantes para tu análisis.

  Por ejemplo, si vas a predecir la aprobación:

```
features = ['horas_estudio', 'asistencia', 'participacion', 'genero']
target = 'aprobado'
X = df[features]
y = df[target]
```

---

6. División del Conjunto de Datos

División en entrenamiento y prueba:
Para evaluar el modelo, separa los datos:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

---

7. Modelado

Selección del algoritmo:
En este ejemplo, utilizaremos una regresión logística para un problema de clasificación.

Entrenamiento del modelo:

```
model = LogisticRegression()
model.fit(X_train, y_train)
```

---

8. Evaluación del Modelo

Predicciones y análisis de resultados:

```
y_pred = model.predict(X_test)
print("Precisión del modelo:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Matriz de confusión:

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.title("Matriz de Confusión")
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.show()
```

---

9. Interpretación y Conclusiones

Analiza los resultados:

- ¿Cuál es la precisión del modelo?
- ¿Qué variables parecen tener mayor influencia según el modelo?
- Discute las limitaciones y posibles mejoras (más datos, otros algoritmos, etc.).

Documentación:
Asegúrate de anotar cada paso, las decisiones tomadas y las conclusiones obtenidas. Esto es esencial para la reproducibilidad y para aprender de la experiencia.

---

10. Presentación de Resultados

Visualizaciones finales:
Prepara gráficos que resuman el comportamiento del modelo y los hallazgos principales.
Informe:
Elabora un reporte o presentación en la que expliques:

- El objetivo del ejercicio.
- La metodología seguida.
- Los resultados y su interpretación.
- Posibles recomendaciones o acciones basadas en los hallazgos.

Recursos Adicionales

- Tutoriales en línea:
  - Kaggle Learn ofrece cursos prácticos sobre análisis de datos.
- Libros:
  - *"Python for Data Analysis"* de Wes McKinney, que es excelente para aprender a manejar datos con Pandas.
- Comunidades:
  - Participa en foros y grupos de estudio en Stack Overflow, Reddit o comunidades de Data Science en español.

Esta guía te ayudará a montar un ejercicio práctico de minería de datos desde cero. La clave está en definir claramente el problema, trabajar ordenadamente desde la exploración de datos hasta el modelado y la interpretación, y utilizar las herramientas y recursos adecuados para cada fase del proceso. ¿Te gustaría profundizar en algún paso o ver un ejemplo de código más detallado en algún apartado?