EXPLORATORY DATA ANALYSIS

Exploracion inicial

`.head()` y `.info()`

Para variables categoricas, ver categorias y cuenta `.value_counts()`

`.describe()`

Ver variables numéricas con histogramas

Data Validation

Para ver únicamente los tipos de datos → `books.dtypes`

Convertimos al tipo deseado

Para comprobar → `books['col'].isin(['valor_A', 'valor_B'])`

→ Por cada fila retorna true si el valor está en la lista.

Para ver solo las columnas numéricas → `books.select_dtypes('number')`

Metiendo datos en los nulos

Para ver la cantidad de valores nulos en cada columna → `df.isna().sum()`

Para encontrar el nivel de valores máximo por columna nulo →

```
threshold = len(df) * 0.05
cols_to_drop = df.columns[df.isna().sum() <= threshold]
df.dropna(subset = cols_to_drop, inplace=True)
```

Convirtiendo y analizando datos categóricos

Para comprobar que una fila contiene esto o lo otro `df['col'].str.contains("esto|lo otro")`

Para encontrar multiples frases →

```
categorias = ['A', 'B', 'C'....]
conditions = [(df['col'].str.contains(frase_A)),
              - -        -  ..
             ]
df['New_col'] = np.select(conditions, categorias, default='Other')
```

Deben estar en el mismo orden

Trabajando los outliers

Preguntamos → ¿Porque está?
↘ ¿Están bien los datos?

EXPLORATORY DATA ANALYSIS

Patrones temporales

Para extraer datos de un datetime →

```
df['col'].dt.month
  "       .dt.year
  "       .dt.day
```

Correlacion

Usa el coeficiente de Pearson

```
df.corr()
```

Consideraciones para datos categoricos

Para ver el porcentaje de cada categoria → `df['col'].value_counts(normalize=True)`

Para agregar valores usando crosstab → `pd.crosstab(planes['Stick'], planes['Destino'], values=planes['Price'], aggfunc='median')`