**Guia**

Este documento expone los pasos a seguir para la elección de los conjuntos de datos, y el objetivo del análisis.

**Objetivo del Analisis**

**Elección del conjunto**

**Proceso de EDA**

**1. Lectura de archivos**

Se importaron las librerías básicas, pandas y numpy, se leyeron los archivos CSV y se convirtieron en DataFrames para trabajar con ellos mediante el método de pandas read\_csv

circuits = pd.read\_csv('./00\_data\_raw/circuits.csv', index\_col=0)

constructor\_results = pd.read\_csv('./00\_data\_raw/constructor\_results.csv', index\_col=0)

# ...se leen todos los CSVs hasta status.csv

status = pd.read\_csv('./00\_data\_raw/status.csv', index\_col=0)

```

**2. Limpieza inicial y formateo**

Para cada DataFrame se aplicaron:

- Eliminación de columnas innecesarias (ej. 'url', 'positionText', 'milliseconds').

- Sustitución de '\N' por valores NaN.

- Formateo de textos (capitalizar 'circuitRef', estandarizar nacionalidades, etc.).

Ejemplo:

```python

circuits.drop(columns=['url'], inplace=True)

circuits.replace('\\N', np.nan, inplace=True)

circuits['circuitRef'] = circuits['circuitRef'].str.capitalize()

```

**3. Revisión de tipos y calidad de datos**

Se utilizó `DataFrame.info()` para verificar tipos de datos y contar valores nulos en cada columna. Esto permitió identificar dónde aplicar conversiones de tipo y limpieza adicional.

**4. Transformaciones adicionales**

A continuación, las transformaciones propuestas antes de generar los CSV finales:

**\*\*4.1 Conversión de fechas a datetime\*\***

```python

races['date'] = pd.to\_datetime(races['date'], errors='coerce')

races['quali\_date'] = pd.to\_datetime(races['quali\_date'], errors='coerce')

races['sprint\_date'] = pd.to\_datetime(races['sprint\_date'], errors='coerce')

drivers['dob'] = pd.to\_datetime(drivers['dob'], errors='coerce')

```

**\*\*4.2 Conversión de duraciones y tiempos\*\***

```python

pit\_stops['duration\_s'] = pd.to\_timedelta(pit\_stops['duration'], unit='s')

lap\_times['lap\_time'] = pd.to\_timedelta(lap\_times['time'])

for df in [results, sprint\_results]:

df['fastestLapTime\_td'] = pd.to\_timedelta(df['fastestLapTime'])

```

**\*\*4.3 Limpieza y estandarización de textos\*\***

```python

constructors['nationality'] = constructors['nationality'].str.title()

status['status'] = status['status'].str.lower().str.strip()

```

**\*\*4.4 Tratamiento de nulos y duplicados\*\***

Se revisó el porcentaje de nulos por columna e indices duplicados para eliminar o imputar según conveniencia.

**\*\*4.5 Ingeniería de características\*\***

- Edad de piloto en cada carrera.

- Velocidad media por vuelta (distancia/tiempo).

- Agregados: puntos totales, número de victorias, podios por temporada.

**Comprobaciones de calidad y relaciones referenciales**

Se verificó la integridad de claves foráneas (existencia de circuitId, driverId, constructorId en sus tablas maestras).

**Generación de Tablas Finales**

Se generaron los siguientes DataFrames finales y se exportaron a CSV para carga en Power BI:

Resultados completos uniendo Results + Drivers + Constructors + Races + Circuits

Puntos por piloto y escudería por temporada

Tabla de pit stops limpia

Resumen de temporada

**Exportar a CSV**

*merged\_results.to\_csv('./02\_data\_cleaned/merged\_results.csv', index=False)*

*driver\_points.to\_csv('./02\_data\_cleaned/driver\_points.csv', index=False)*

*constructor\_points.to\_csv('./02\_data\_cleaned/constructor\_points.csv', index=False)*

*pit\_stops\_clean.to\_csv('./02\_data\_cleaned/pit\_stops\_clean.csv', index=False)*

*lap\_times\_std.to\_csv('./02\_data\_cleaned/lap\_times\_std.csv', index=False)*

*season\_summary.to\_csv('./02\_data\_cleaned/season\_summary.csv', index=False)*