Taller N°3: Preprocesamiento

Electiva II

Docente a cargo

Cristhian Alejandro Cañar Muñoz

Presentado por

Ulises ortega revelo

Alex Ramirez

Universidad autónoma del cauca

2025

1. INTRODUCCION

Se procedió a escoger un dataset de la pagina de Kaggle el cual tiene como nombre **car_price_dataset.csv** contiene un conjunto de datos sobre precios de automóviles con 10,000 registros y 10 columnas.

Columnas del Dataset

1. **Brand** (*str*): Marca del vehículo (ej. Kia, Chevrolet, Mercedes).
2. **Model** (*str*): Modelo del vehículo.
3. **Year** (*int*): Año de fabricación (2000 - 2023).
4. **Engine_Size** (*float*): Tamaño del motor en litros (1.0 - 5.0 L).
5. **Fuel_Type** (*str*): Tipo de combustible (ej. Diesel, Híbrido, Eléctrico).
6. **Transmission** (*str*): Tipo de transmisión (Manual, Automática, Semi-Automática).
7. **Mileage** (*int*): Kilometraje del vehículo (25 - 299,947 km).
8. **Doors** (*int*): Número de puertas (2 - 5).
9. **Owner_Count** (*int*): Cantidad de dueños anteriores (1 - 5).
10. **Price** (*int*): Precio del vehículo en dólares (2,000 - 18,301).

2. CODIGO Y PROCEDIMIENTO

A continuación, se explicara a detalle el código donde se llevó a cabo la limpieza del data set.

```
import pandas as pd

data_importada = pd.read_csv(r'C:\Users\Personal\Downloads\archive\car_price_dataset.csv',sep=',',decimal='.')
df_data_csv = pd.DataFrame(data_importada)
```

1. **Importar pandas**:
   Se usa pandas para trabajar con datos en forma de tablas, y se apoda pd.

2. **Cargar el archivo CSV**:
   El archivo car_price_dataset.csv se importa desde la ruta especificada.
3. **Convertir a DataFrame**:
   Los datos del archivo se convierten en una tabla (DataFrame) para poder manipularlos fácilmente.

VALORES NULOS

```
print(df_data_csv.isnull().sum()[df_data_csv.isnull().sum() > 0])
```

1. **Verificar valores nulos**:
   df_data_csv.isnull() identifica los valores faltantes (nulos) en el DataFrame.
2. **Contar valores nulos**:
   .sum() cuenta cuántos valores nulos hay en cada columna.
3. **Filtrar columnas con nulos**:
   [df_data_csv.isnull().sum() > 0] muestra solo las columnas que tienen al menos un valor nulo.

VALORES DUPLICADOS

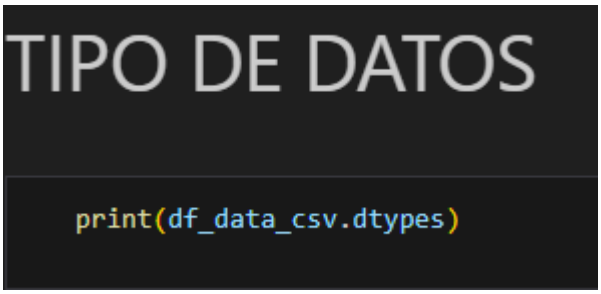
```
print(df_data_csv.duplicated().sum())
print(f"Filas duplicadas: {df_data_csv.duplicated().sum()}")
```

1. **Verificar filas duplicadas**:
   df_data_csv.duplicated() busca filas repetidas en el DataFrame.

2. **Contar filas duplicadas**:
   .sum() cuenta cuántas filas están duplicadas.

3. **Mostrar el resultado**:

   El primer print muestra solo el número de filas duplicadas.
   El segundo print muestra un mensaje más claro: "Filas duplicadas: X", donde X es el número de filas repetidas.



1. **Verificar tipos de datos**:
   df_data_csv.dtypes muestra el tipo de dato (por ejemplo, números, texto, fechas) de cada columna en el DataFrame.

2. **Mostrar el resultado**:
   El print imprime esta información, indicando qué tipo de dato contiene.

```
print(df_data_csv.dtypes)
```
[28] ✓ 0.0s

```
Brand           object
Model           object
Year             int64
Engine_Size    float64
Fuel_Type       object
Transmission    object
Mileage          int64
Doors            int64
Owner_Count      int64
Price            int64
dtype: object
```

Aquí comprobamos que están las columnas que vamos a trabajar y todo está en orden.

Después de esto se verifico si algunas columnas que deberían ser numéricas podrían estar en formato object. Para asegurarnos, sé revisa si hay valores no numéricos en columnas clave con la siguiente línea de código

```
for col in ["Year", "Engine_Size", "Mileage", "Doors", "Owner_Count", "Price"]:
    if df_data_csv[col].dtype == "object":
        print(f"Problema en {col}: valores no numericos detectados")
```

1. El código recorre una lista de columnas ("Year", "Engine_Size", etc.) para verificar su tipo de dato.
2. **Detectar valores no numéricos**:
   Si alguna de estas columnas tiene un tipo de dato "object" (generalmente texto o valores no numéricos), se imprime un mensaje indicando que hay un problema en esa columna.
3. **Mostrar el resultado**:
   El print avisa en qué columna se encontraron valores no numéricos.

```python
for col in ["Brand", "Model", "Fuel_Type", "Transmission"]:
    df_data_csv[col] = df_data_csv[col].str.strip()
```

1. **Seleccionar columnas de texto**:
   El código recorre una lista de columnas ("Brand", "Model", etc.) que contienen texto.
2. **Eliminar espacios innecesarios**:
   str.strip() elimina espacios en blanco al principio y al final de los textos en esas columnas .

Y obtenemos lo siguiente:

```
[20]  ✓ 0.0s
...                 Year   Engine_Size        Mileage         Doors   Owner_Count  \
count  10000.000000  10000.000000   10000.000000  10000.000000  10000.000000
mean    2011.543700      3.000560  149239.111800      3.497100      2.991100
std        6.897699      1.149324   86322.348957      1.110097      1.422682
min     2000.000000      1.000000      25.000000      2.000000      1.000000
25%     2006.000000      2.000000   74649.250000      3.000000      2.000000
50%     2012.000000      3.000000  149587.000000      3.000000      3.000000
75%     2017.000000      4.000000  223577.500000      4.000000      4.000000
max     2023.000000      5.000000  299947.000000      5.000000      5.000000

              Price
count  10000.00000
mean    8852.96440
std     3112.59681
min     2000.00000
25%     6646.00000
50%     8858.50000
75%    11086.50000
max    18301.00000
```

Descripción Estadística de los Datos A continuación, se presenta un resumen estadístico de las variables numéricas clave:

- **Año (Year):** Rango entre 2000 y 2023 con una media de 2011.54 y una desviación estándar de 6.89. La distribución muestra una concentración de datos alrededor de la media, sin presencia evidente de valores extremos.
- **Tamaño del motor (Engine_Size):** Rango entre 1.0 y 5.0 litros con una media de 3.00 y una desviación estándar de 1.14. Los valores siguen una distribución normal con variabilidad esperada dentro del sector automotriz.
- **Kilometraje (Mileage):** Rango entre 25 km y 299,947 km con una media de 149,239 km y una desviación estándar de 86,322 km. Se observa una alta dispersión en los datos, con valores extremos que podrían indicar autos con uso intensivo o registros erróneos.
- **Número de puertas (Doors):** Entre 2 y 5 puertas, con una media de 3.49 y una desviación estándar de 1.11. No se observan valores fuera del rango típico.
- **Cantidad de dueños previos (Owner_Count):** Entre 1 y 5 propietarios, con una media de 2.99 y una desviación estándar de 1.42. La distribución es uniforme y no presenta valores anómalos.
- **Precio (Price):** Rango entre $2,000 y $18,301 con una media de $8,852 y una desviación estándar de $3,112. Algunos valores cercanos a los extremos podrían representar ofertas inusuales o errores en el dataset.

Identificación de Valores Atípicos Para detectar valores atípicos, se utilizó el Rango Intercuartílico (IQR), el cual se define como:

Donde:

- **Q1 (Primer Cuartil):** 25% de los datos por debajo de este valor.
- **Q3 (Tercer Cuartil):** 75% de los datos por debajo de este valor.

Se consideran valores atípicos aquellos que se encuentran fuera del rango:

Valores Atípicos Identificados:

- **Kilometraje (Mileage):** Se detectaron valores atípicos en los extremos. Vehículos con kilometrajes superiores a 299,947 km podrían indicar unidades con uso extremadamente alto o errores de registro. Por otro lado, los valores más bajos (25 km) podrían corresponder a vehículos nuevos o a inconsistencias en la entrada de datos.

- Se consideran atípicos los valores inferiores a **Q1 - 1.5 * IQR = 74649.25 - 1.5(223577.5 - 74649.25) = -151790.875** (lo que es imposible en la práctica) y los valores superiores a **Q3 + 1.5 * IQR = 223577.5 + 1.5(223577.5 - 74649.25) = 450017.625**. En este caso, como el máximo en el dataset es 299,947 km, este valor se encuentra en un rango elevado, pero aún dentro de un límite realista. Sin embargo, se debe evaluar si estos autos corresponden a vehículos comerciales o con uso excesivo.

- **Precio (Price):** Los valores más bajos (cercanos a $2,000) y más altos (cercanos a $18,301) se identifican como atípicos. Estos podrían representar ofertas inusuales, vehículos en condiciones excepcionales o errores en la recolección de datos.

- Se consideran atípicos los valores inferiores a **Q1 - 1.5 * IQR = 6646 - 1.5(11086.5 - 6646) = 1863.25**, lo que implica que cualquier valor inferior a $1,863 podría ser un outlier. En este caso, el mínimo registrado es $2,000, lo cual está cerca del límite inferior, pero no representa un caso extremo. Por otro lado, los valores superiores a **Q3 + 1.5 * IQR = 11086.5 + 1.5(11086.5 - 6646) = 15969.25** se considerarían atípicos. Como el máximo registrado es $18,301, se pueden considerar como valores extremos que deben analizarse más detalladamente.

- **Año (Year), Tamaño del motor (Engine_Size), Número de puertas (Doors) y Cantidad de dueños previos (Owner_Count):** No presentan valores atípicos significativos, ya que todos los registros se encuentran dentro de rangos esperados para el mercado automotriz.

LIMPIEZA DEL DATASET

```
# Filtrar solo las columnas numericas
df_numeric = df_data_csv.select_dtypes(include=["number"])

# Calcular el rango intercuartilico (IQR)
Q1 = df_numeric.quantile(0.25)
Q3 = df_numeric.quantile(0.75)
IQR = Q3 - Q1

# Filtrar valores dentro del rango valido
df_clean = df_data_csv[~((df_numeric < (Q1 - 1.5 * IQR)) | (df_numeric > (Q3 + 1.5 * IQR))).any(axis=1)]

# Guardar el dataset limpio
df_clean.to_csv("car_price_dataset_clean.csv", index=False)
print("Valores atipicos eliminados y dataset limpio guardado.")
display(df_clean)
```

1. **Filtrar columnas numéricas**:

   Se seleccionan solo las columnas con datos numéricos (df_numeric).

2. **Calcular el rango intercuartílico (IQR)**:

   - Q1 es el primer cuartil (25% de los datos).
   - Q3 es el tercer cuartil (75% de los datos).
   - IQR es la diferencia entre Q3 y Q1.

3. **Eliminar valores atípicos**:

   Se filtran los datos que están fuera del rango válido (por debajo de Q1 - 1.5 * IQR o por encima de Q3 + 1.5 * IQR).

4. **Guardar el dataset limpio**:

   El DataFrame sin valores atípicos (df_clean) se guarda en un nuevo archivo CSV (car_price_dataset_clean.csv).

5. **Mostrar el resultado**:

   Se imprime un mensaje confirmando que se eliminaron los valores atípicos y se muestra el DataFrame limpio.

3. CONCLUSIONES

En este documento se llevó a cabo un exhaustivo proceso de limpieza del conjunto de datos car_price_dataset. csv. En primer lugar, se identificaron los valores nulos empleando la función df_data_csv. isnull(). sum(), que permitió contar la cantidad de datos faltantes en cada columna. Posteriormente, se comprobó la existencia de filas duplicadas utilizando df_data_csv. duplicated(). sum(), determinando que no había registros duplicados en el conjunto de datos.

A continuación, se revisaron los tipos de datos a través de df_data_csv. dtypes, asegurando que columnas como "Year" y "Engine_Size" fueran de tipo numérico y no contuvieran valores inválidos. También se eliminaron espacios innecesarios en columnas de texto como "Brand" y "Model" utilizando el método str. strip().

Para la detección de valores atípicos, se calculó el Rango Intercuartílico (IQR). Se obtuvieron los cuartiles Q1 (25%) y Q3 (75%) con df_numeric. quantile(), estableciendo un rango válido como [Q1 - 1. 5 * IQR, Q3 + 1. 5 * IQR]. Se consideraron atípicos los valores que se encontraban fuera de este rango y se procedió a su eliminación.

Finalmente, el conjunto de datos limpio se guardó en car_price_dataset_clean. csv, dejándolo listo para futuros análisis. Este riguroso proceso garantizó la calidad y la fiabilidad de los datos.