

**Benemérita Universidad Autónoma de
Facultad de Ciencias de la Computación
Asignatura: Introducción a la ciencia de
datos**

Profesor: Jaime Alejandro Romero Sierra

Alumno: Carlos Alexander Castillo Gomez

Sección:001

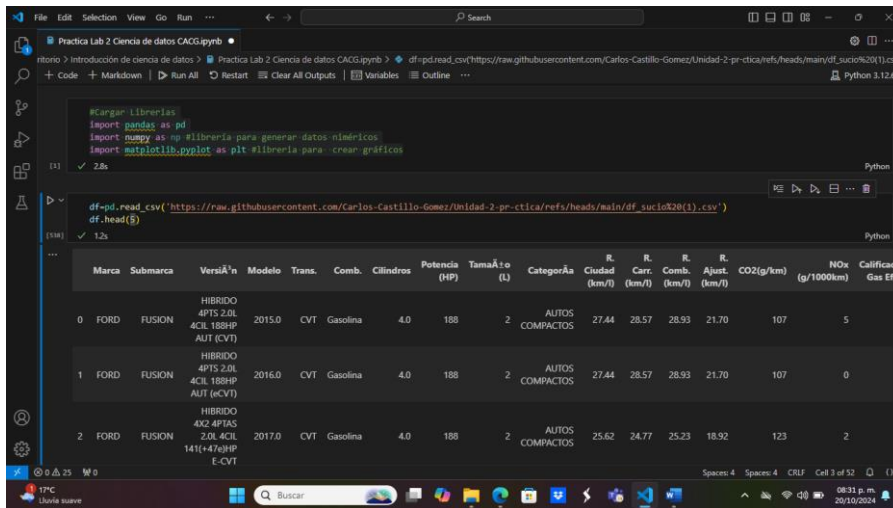
**Actividad: Práctica de Laboratorio. Limpieza
de una base de datos ensuciada**

**Objetivo: Desarrollar habilidades en el
preprocesamiento de datos, incluyendo la
identificación y tratamiento de valores
faltantes, datos duplicados, y formatos
inconsistentes en una base de datos,**

Fecha de entrega: 20/10/2024

¿Qué pasos realice para limpiar la base de datos?

1. Comencé con cargar las librerías, las que ocupe fue pandas y matplotlib, ya después de eso cargue en Visual code el enlace de la base de datos que subí a GitHub



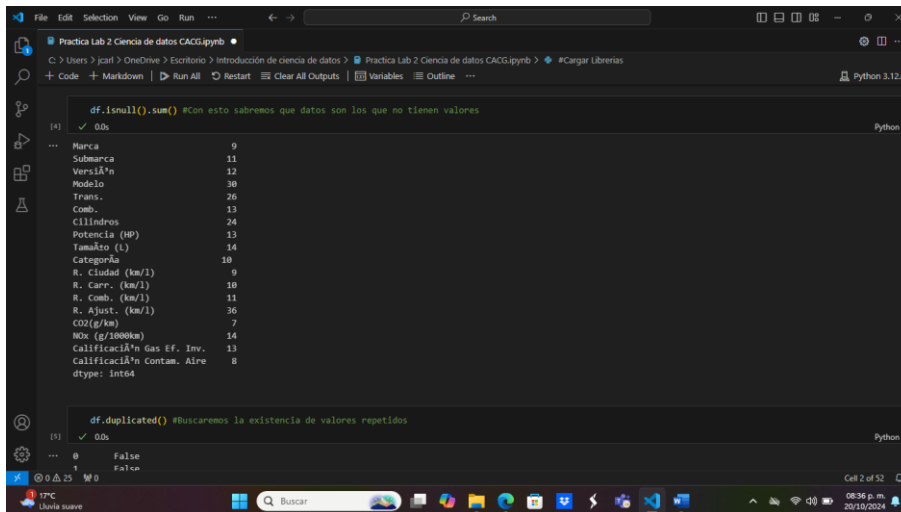
The screenshot shows a Jupyter Notebook with the following code and output:

```
#Cargar librerías
import pandas as pd
import numpy as np #librería para generar datos numéricos
import matplotlib.pyplot as plt #librería para crear gráficos
```

```
df=pd.read_csv('https://raw.githubusercontent.com/Carlos-Castillo-Gomez/Unidad-2-prctica/refs/heads/main/df_sucio%281).csv')
df.head(5)
```

	Marca	Submarca	Versión	Modelo	Trans.	Comb.	Cilindros	Potencia (HP)	Tamaño (L)	Categoría	R. Ciudad (km/l)	R. Carr. (km/l)	R. Comb. (km/l)	R. Ajust. (km/l)	CO2(g/km)	NOx (g/1000km)	Calificación Gas Ef.
0	FORD	FUSION	HIBRIDO 4PTS 2.0L 4CIL 188HP AUT (CVT)	2015.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107		5
1	FORD	FUSION	HIBRIDO 4PTS 2.0L 4CIL 188HP AUT (eCVT)	2016.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107		0
2	FORD	FUSION	HIBRIDO 4X2 4PTAS 2.0L 4CIL 141(+4)188HP E-CVT	2017.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	25.62	24.77	25.23	18.92	123		2

2. Ya comenzando el análisis de las principales funciones que ocupe fue el “isnull (). sum ()” y “duplicated” esto para saber cuántos datos no tenían valores y para buscar la existencia de valores repetidos



The screenshot shows a Jupyter Notebook with the following code and output:

```
df.isnull().sum() #Con esto sabremos que datos son los que no tienen valores
```

```
df.duplicated() #Buscaremos la existencia de valores repetidos
```

```
df.info()
```

3. También ocupe la función “duplicated (). sum ()” para saber los valores duplicados y finalmente el “info ()” para saber más acerca de cada columna y principalmente que tipo de datos tenía cada una

```

df.duplicated().sum() #Los valores duplicados
Out[14]: 0
np.int64(1806)

df.info() #Saber mas acerca de cada columna, como el tipo de datos que nos presenta y si es de acuerdo a lo que buscamos y queremos
Out[17]:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5799 entries, 0 to 5798
Data columns (total 18 columns):
 #   Column              Non-Null Count  Dtype
---  ---
 0   Marca               5799 non-null  object
 1   Submarca            5798 non-null  object
 2   VersiÃ³n            5787 non-null  object
 3   Modelo             5769 non-null  float64
 4   Trans.              5773 non-null  object
 5   Comb.               5786 non-null  object
 6   Cilindros           5775 non-null  float64
 7   Potencia (HP)       5785 non-null  object
 8   Tanakto (l)         5785 non-null  object
 9   Categoría           5769 non-null  object
10  R. Ciudad (km/l)    5798 non-null  object
11  R. Carr. (km/l)     5789 non-null  object
12  R. Comb. (km/l)     5788 non-null  object
13  R. Ajust. (km/l)    5763 non-null  float64
14  CO2(g/km)          5792 non-null  object

```

- Con funciones como “unique” e incluso la creación de gráficas pude saber mas acerca de que datos eran los que existían en cada columna, de ahí pude saber que datos que no concordaban era “nan” y “aaaaa”.


```

df['Trans.'].unique()
Out[18]:
array(['CVT', 'AUT', 'MAN', 'DSG', 'TIP', 'TIPTRONIC', 'MS', 'MS', nan,
       'DIALOGIC', 'TIP', 'MULTITRONIC', 'MULTI', 'STRONIC', 'S TRONIC',
       'aaaaa', 'S TRONIC 2', 'MAN 6', 'POK', 'DCT', 'G TRONIC', 'TIP B',
       'R TRONIC', 'ZF SPEED QUICKSHIFT', 'DSG', 'AS6'], dtype=object)

#Saber más sobre los datos de cada columna
gender_counts=df['Comb.'].value_counts()

#Gráfico de pastel
plt.figure(figsize=(4,4))
plt.pie(gender_counts, labels=gender_counts.index, autopct='%1.1f%%', startangle=90, colors=['blue', 'green', 'red'])
plt.title('Datos en la columna Combustible')
plt.show()

```



- Por lo que hice eso con cada columna de acuerdo con cuantos datos presentaba era la función que ocupé y al final saber cuántos “aaaaa” existían en cada columna

```

df['Calificación Contam. Aire'].unique()
Out[24]:
array(['G', nan, 'B', '6', '3', '7', '5', '4', '2', '3', 'aaaaa', '1'],
      dtype=object)

for j in df.columns:
    print(f'En las columnas {j} los aaaaa: {df[df[j]=="aaaaa"].shape[0]}')

```

```

En las columnas Marca los aaaaa: 15
En las columnas Submarca los aaaaa: 19
En las columnas VersiÃ³n los aaaaa: 15
En las columnas Modelo los aaaaa: 0
En las columnas Trans. los aaaaa: 16
En las columnas Comb. los aaaaa: 16
En las columnas Cilindros los aaaaa: 0
En las columnas Potencia (HP) los aaaaa: 17
En las columnas Tanakto (l) los aaaaa: 23
En las columnas Categoría los aaaaa: 18
En las columnas R. Ciudad (km/l) los aaaaa: 18
En las columnas R. Carr. (km/l) los aaaaa: 18
En las columnas R. Comb. (km/l) los aaaaa: 19
En las columnas R. Ajust. (km/l) los aaaaa: 0
En las columnas CO2(g/km) los aaaaa: 20
En las columnas NOx (g/1000km) los aaaaa: 15
En las columnas Calificación Gas ff. Inv. los aaaaa: 24
En las columnas Calificación Contam. Aire los aaaaa: 20

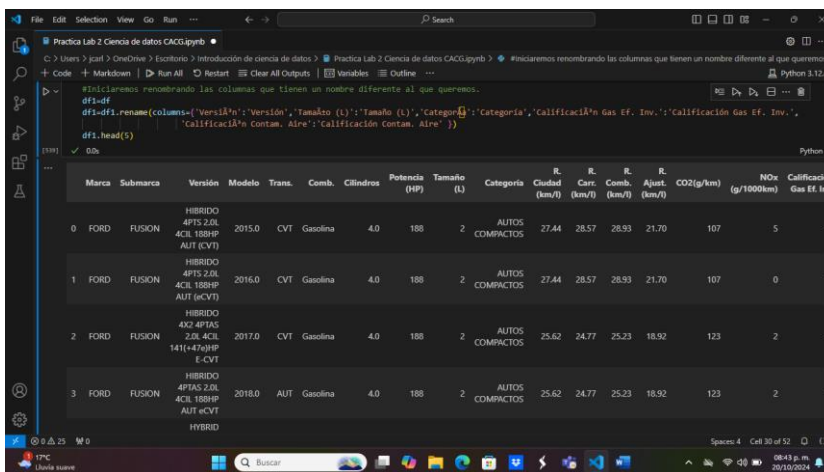
```

6. Ya después de analizar los “errores” realizar la conclusión

Conclusión

1. Como primeros cambios que note sería la corrección del nombre de las columnas, ya que tienen caracteres que simplemente hacen que no se entienda de manera eficaz en que consiste cada columna.
2. Con la función “isnull().sum()” note que en todas las columnas existen datos que no tienen valores, por lo que será necesario realizar algo para que no nos afecten en un futuro.
3. Con la función “duplicated()” encontramos que existen valores repetidos, por lo que será necesario quitarlos.
4. Con la función “info()” podemos observar que la mayoría de las columnas los Dtype están en object, lo cual en unas sí es necesario pero en otras donde lo que más se presentan son números, nos sería de mucha ayuda el que estén en tipo “int” o “float” de acuerdo al caso.
5. Al realizar las gráficas para apreciar más a detalle cuáles son los datos que tenemos se puede notar que existe un dato “aaaaa” lo cual no nos dice nada por lo que lo mejor sería removerlo.
6. El primer error con las gráficas es que algunas no se pudieron graficar por el simple hecho de que son de tipo “str”, por lo que fue necesario ocupar la función “unique()”, con esto nos dimos cuenta que si existe el dato “aaaaa” que tendremos que reemplazar además de que en unas columnas hay datos “nan” los cuales tendremos que realizar algo con ellos.

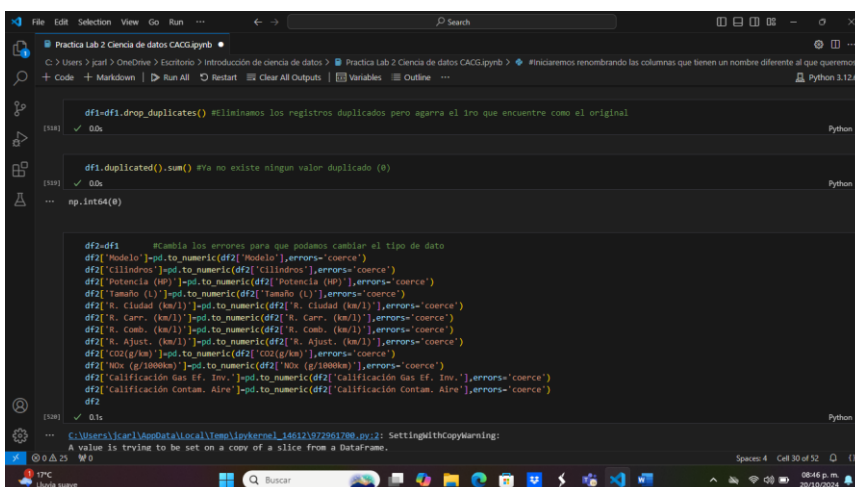
7. Inicie renombrando las columnas que tenían un nombre que no correspondía, esto con el comando rename.



```
#Iniciamos renombrando las columnas que tienen un nombre diferente al que queremos.
df1=df1.rename(columns={'Versión':'Modelo','Tamaño (l)':'Tamaño (L)','Categoría':'Categoría','Calificación Gas Ef. Inv.':'Calificación Gas Ef. Inv.',
                        'Calificación Contam. Aire':'Calificación Contam. Aire'})
df1.head(5)
```

	Marca	Submarca	Versión	Modelo	Trans.	Comb.	Cilindros	Potencia (HP)	Tamaño (L)	Categoría	R. Ciudad (km/l)	R. Carr. (km/l)	R. Comb. (km/l)	R. Ajust. (km/l)	CO2(g/km)	NOx (g/1000km)	Calificación Gas Ef. Inv.	Calificación Contam. Aire
0	FORD	FUSION	HIBRIDO 4P1S 2.0L ACIL 188HP AUT (CVT)	2015.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107	5		
1	FORD	FUSION	HIBRIDO 4P1S 2.0L ACIL 188HP AUT (CVT)	2016.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107	0		
2	FORD	FUSION	HIBRIDO 4X2 4P1AS 2.0L ACIL 141(+4)HP 1-CVT	2017.0	CVT	Gasolina	4.0	188	2	AUTOS COMPACTOS	25.62	24.77	25.23	18.92	123	2		
3	FORD	FUSION	HIBRIDO 4P1AS 2.0L ACIL 188HP AUT eCVT	2018.0	AUT	Gasolina	4.0	188	2	AUTOS COMPACTOS	25.62	24.77	25.23	18.92	123	2		

8. Con el comando “drop_duplicates ()” eliminamos los registros duplicados, para que con “duplicated (). sum ()” nos aparezca el que ya no existe ningún valor duplicado. Ahora, con la función “pd.to_numeric [], errors=’coerce’” cambiaremos los errores por algo “numérico” y así poder cambiar el tipo de dato, esto solo funciona en las columnas que sus datos eran de puros números, nada de texto.



```
df1=df1.drop_duplicates() #Eliminamos los registros duplicados pero agarra el 1ro que encuentre como el original
df1.duplicated().sum() #Ya no existe ningún valor duplicado (0)
np.int64(0)

df2=df1 #cambia los errores para que podamos cambiar el tipo de dato
df2['Modelo']=pd.to_numeric(df2['Modelo'],errors='coerce')
df2['Cilindros']=pd.to_numeric(df2['Cilindros'],errors='coerce')
df2['Potencia (HP)']=pd.to_numeric(df2['Potencia (HP)'],errors='coerce')
df2['Tamaño (l)']=pd.to_numeric(df2['Tamaño (l)'],errors='coerce')
df2['R. Ciudad (km/l)']=pd.to_numeric(df2['R. Ciudad (km/l)'],errors='coerce')
df2['R. Carr. (km/l)']=pd.to_numeric(df2['R. Carr. (km/l)'],errors='coerce')
df2['R. Comb. (km/l)']=pd.to_numeric(df2['R. Comb. (km/l)'],errors='coerce')
df2['R. Ajust. (km/l)']=pd.to_numeric(df2['R. Ajust. (km/l)'],errors='coerce')
df2['CO2(g/km)']=pd.to_numeric(df2['CO2(g/km)'],errors='coerce')
df2['NOx (g/1000km)']=pd.to_numeric(df2['NOx (g/1000km)'],errors='coerce')
df2['Calificación Gas Ef. Inv.']=pd.to_numeric(df2['Calificación Gas Ef. Inv.'],errors='coerce')
df2['Calificación Contam. Aire']=pd.to_numeric(df2['Calificación Contam. Aire'],errors='coerce')
df2
```

9. Ya que cambiamos nuestros datos en las columnas de datos numéricos a algo numérico valga la redundancia, utilice el comando “fillna (). mean (), inplace=True” para que sustituya los ‘nan’ por el promedio de cada columna a la cual escribamos.

```
# Sustituye los Nan por el promedio en cada columna que pongamos
df2['Modelo'].fillna(df2['Modelo'].mean(), inplace=True)
df2['cilindros'].fillna(df2['cilindros'].mean(), inplace=True)
df2['Potencia (HP)'].fillna(df2['Potencia (HP)'].mean(), inplace=True)
df2['Tamaño (L)'].fillna(df2['Tamaño (L)'].mean(), inplace=True)
df2['R. Ciudad (km/l)'].fillna(df2['R. Ciudad (km/l)'].mean(), inplace=True)
df2['R. Carr. (km/l)'].fillna(df2['R. Carr. (km/l)'].mean(), inplace=True)
df2['R. Comb. (km/l)'].fillna(df2['R. Comb. (km/l)'].mean(), inplace=True)
df2['R. Ajust. (km/l)'].fillna(df2['R. Ajust. (km/l)'].mean(), inplace=True)
df2['CO2(g/km)'].fillna(df2['CO2(g/km)'].mean(), inplace=True)
df2['NOx (g/100km)'].fillna(df2['NOx (g/100km)'].mean(), inplace=True)
df2['Calificación Gas Ef. Inv.'].fillna(df2['Calificación Gas Ef. Inv.'].mean(), inplace=True)
df2['Calificación Contam. Aire'].fillna(df2['Calificación Contam. Aire'].mean(), inplace=True)
```

10. Ahora, con respecto a las columnas las cuales sus datos son textos, lo que utilice para eliminar los nan fue primero utilizar la función “fillna (0)” para cambiar los ‘nan’ por 0.

```
#Cambiamos los nan por 0
df4['Marca']=df4['Marca'].fillna(0)
df4['Submarca']=df4['Submarca'].fillna(0)
df4['Versión']=df4['Versión'].fillna(0)
df4['Trans.']=df4['Trans.'].fillna(0)
df4['Comb.']=df4['Comb.'].fillna(0)
df4['Categoría']=df4['Categoría'].fillna(0)
```

11. Utilizando la función “replace” cambiaremos los 0 que anteriormente habíamos puesto con el “fillna (0)” por la palabra ‘Sin_información’, esto para que en un futuro no nos genere confusión que entre vario texto existan 0.

```
#Reemplazaremos esos 0 por la palabra "Sin_información"
df4['Marca']=df4['Marca'].replace({0:'Sin_información'})
df4['Submarca']=df4['Submarca'].replace({0:'Sin_información'})
df4['Versión']=df4['Versión'].replace({0:'Sin_información'})
df4['Trans.']=df4['Trans.'].replace({0:'Sin_información'})
df4['Comb.']=df4['Comb.'].replace({0:'Sin_información'})
df4['Categoría']=df4['Categoría'].replace({0:'Sin_información'})
```

12. ¡Como recordemos anteriormente supimos de la existencia de un dato “aaaaa” por lo que decidí eliminarlo, para eso ocuparemos la función “!= ‘aaaaa’” para que elimine ese dato en cada columna, en este caso solo será en las de texto. Ya después con un para haremos un resumen para que ahora sí, ya no existe ese dato en nuestra base

```
df4=df4[df4['Marca'] !='aaaaa'] #Con esto eliminaremos los 'aaaaa' que estan en nuestra base de datos
df4=df4[df4['Submarca'] !='aaaaa']
df4=df4[df4['Trans.'] !='aaaaa']
df4=df4[df4['Comb.'] !='aaaaa']
df4=df4[df4['Versión'] !='aaaaa']
df4=df4[df4['Categoría'] !='aaaaa']

for j in df4.columns:
    print (f'En las columnas {j} los aaaaa: (df4[df4[j]!='aaaaa'].shape[0])")

En las columnas Marca los aaaaa: 0
En las columnas Submarca los aaaaa: 0
En las columnas Versión los aaaaa: 0
En las columnas Modelo los aaaaa: 0
En las columnas Trans. los aaaaa: 0
En las columnas Comb. los aaaaa: 0
En las columnas Cilindros los aaaaa: 0
En las columnas Potencia (HP) los aaaaa: 0
En las columnas Tamaño (L) los aaaaa: 0
En las columnas Categoría los aaaaa: 0
En las columnas R. Ciudad (km/l) los aaaaa: 0
En las columnas R. Carr. (km/l) los aaaaa: 0
En las columnas R. Comb. (km/l) los aaaaa: 0
En las columnas R. Ajust. (km/l) los aaaaa: 0
En las columnas CO2(g/km) los aaaaa: 0
En las columnas NOx (g/100km) los aaaaa: 0
```

13. Ya ahora ocupamos los distintos comandos para asegurarnos que ya no existen valores “raros” en nuestra base de datos. “duplicated ()” para saber la existencia de valores duplicados, “isnull (). sum ()” para verificar que ya no existen datos nulos,

```
df4.duplicated() #No existen valores duplicados ya que sino apareceria True
0      False
1      False
2      False
3      False
4      False
...
5771    False
5775    False
5777    False
5788    False
5794    False
Length: 4694, dtype: bool

df4.isnull().sum() #Verificamos que no existen ya datos nulos
Marca                0
Submarca             0
Versión              0
Modelo               0
Trans.               0
Comb.                0
Cilindros            0
Potencia (HP)        0
Tamaño (L)           0
```

14. Ya por último realicé una búsqueda de filas duplicadas por si las dudas.

```
df4=df4.drop_duplicates() #Eliminamos por si acaso existiera algo duplicado

df4.duplicated().sum() #Ya no hay nada duplicado
np.int64(0)
```

15. Ya que nuestra base quedo limpia, procedí a cambiar con ayuda del comando “ . astype(int)” o “ . astype(float)” de acuerdo con el caso para cambiar el tipo de dato ya que antes la mayoría estaba en object, pero como recordamos la mayoría de las columnas son de números.

```
df4['Modelo']=df4['Modelo'].astype(int)
df4['cilindros']=df4['cilindros'].astype(int)
df4['Tamaño (L)']=df4['Tamaño (L)'].astype(float)
df4['R. Ciudad (km/l)']=df4['R. Ciudad (km/l)'].astype(float)
df4['R. Carr. (km/l)']=df4['R. Carr. (km/l)'].astype(float)
df4['R. Comb. (km/l)']=df4['R. Comb. (km/l)'].astype(float)
df4['R. Ajust. (km/l)']=df4['R. Ajust. (km/l)'].astype(float)

df4['Potencia (HP)']=df4['Potencia (HP)'].astype(int)
df4['CO2(g/km)']=df4['CO2(g/km)'].astype(int)
df4['NOx (g/100km)']=df4['NOx (g/100km)'].astype(int)
df4['Calificación Gas Ef. Inv.']=df4['Calificación Gas Ef. Inv.'].astype(int)
df4['Calificación Contam. Aire']=df4['Calificación Contam. Aire'].astype(float)
```

16. Realizamos un resumen para mostrar que ya se han cambiado los tipos de datos. Esto con la función “.info ()”

```
df4.info() #Mostramos resumen de que ahora ya nuestros datos son de diferente tipo
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4692 entries, 0 to 5794
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   Marca                 4692 non-null  object
1   Submarca              4692 non-null  object
2   Versión               4692 non-null  object
3   Modelo                4692 non-null  int64
4   Trans.                4692 non-null  object
5   Comb.                 4692 non-null  object
6   Cilindros             4692 non-null  int64
7   Potencia (HP)         4692 non-null  int64
8   Tamaño (L)            4692 non-null  float64
9   Categoría             4692 non-null  object
10  R. Ciudad (km/l)       4692 non-null  float64
11  R. Carr. (km/l)        4692 non-null  float64
12  R. Comb. (km/l)        4692 non-null  float64
13  R. Ajust. (km/l)       4692 non-null  float64
14  CO2(g/km)              4692 non-null  int64
15  NOx (g/100km)          4692 non-null  int64
16  Calificación Gas Ef. Inv. 4692 non-null  int64
17  Calificación Contam. Aire 4692 non-null  float64
dtypes: float64(6), int64(6), object(6)
memory usage: 696.5+ KB
```

17. Ya que nuestra base ha quedado lista procedemos a reindexar el índice esto con el comando (`.reset_index(drop=True)`) esto para que no existan “saltos” entre números de fila.

```
df5=df4
df5=df4.reset_index(drop=True)
df5.head(5)
```

	Marca	Submarca	Versión	Modelo	Trans.	Comb.	Cilindros	Potencia (HP)	Tamaño (L)	Categoría	R. Ciudad (km/l)	R. Carr. (km/l)	R. Comb. (km/l)	R. Ajust. (km/l)	CO2(g/km)	NOx (g/1000km)	Calificaci Gas Ef. In
0	FORD	FUSION	HIBRIDO 4PTS 2.0L 4CIL 188HP AUT (CVT)	2015	CVT	Gasolina	4	188	2.0	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107	5	
1	FORD	FUSION	HIBRIDO 4PTS 2.0L 4CIL 188HP AUT (eCVT)	2016	CVT	Gasolina	4	188	2.0	AUTOS COMPACTOS	27.44	28.57	28.93	21.70	107	0	
2	FORD	FUSION	HIBRIDO 4X2 4PTAS 2.0L 4CIL 141(+47e)HP E-CVT	2017	CVT	Gasolina	4	188	2.0	AUTOS COMPACTOS	25.62	24.77	25.23	18.92	123	2	
3	FORD	FUSION	HIBRIDO 4PTAS 2.0L 4CIL 188HP AUT eCVT	2018	AUT	Gasolina	4	188	2.0	AUTOS COMPACTOS	25.62	24.77	25.23	18.92	123	2	
4	FORD	FUSION	HYBRID 4PTS 2.5L	2014	CVT	Gasolina	4	155	2.5	AUTOS	24.88	24.68	23.84	17.36	135	5	

18. Finalmente guardamos nuestra base en un CSV esto con la función (`.to_csv[name], index=True`) para así ya tener nuestra base de datos limpia

```
#Guardar los resultados en un csv
df5.to_csv("Base_limpia_proyecto_CACG",index=True)
```

✓ 0.1s Python