EJERCICIOS DE MATPLOTLIB

Cargue el fichero bmw.csv y prepare numpy y matplotlib con el siguiente código:

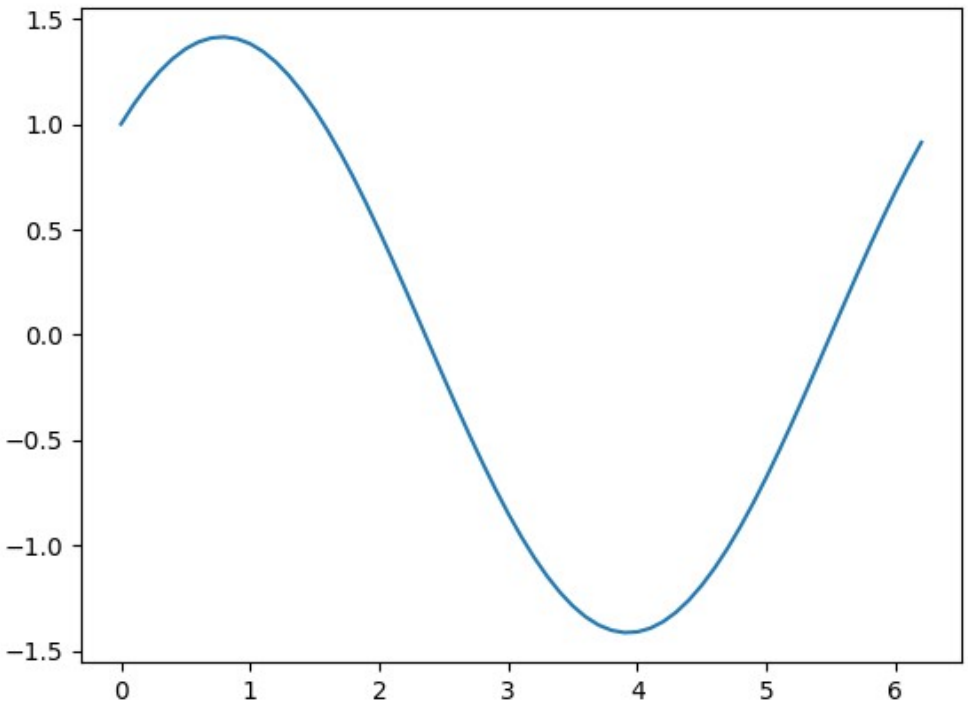
```
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

%matplotlib inline

# Conectar a Google Drive si está almacenado ahí
from google.colab import drive
drive.mount('/content/drive')

df = pd.read_csv('/content/drive/MyDrive/my_data/bmw.csv')
# en otro caso usar ubicación del archivo
df = pd.read_csv('C:/…/bmw.csv')
```

Ejercicio 1:

Representa la función f(x) = sin(x)+cos(x) en el intervalo [0,2*pi].

```
import matplotlib as mpl

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np


df = pd.read_csv('bmw.csv')


x = np.arange(start=0, stop=2*np.pi, step=0.1)


plt.plot(x, np.sin(x) + np.cos(x))

plt.savefig('ej1.png')
```

Resultado:

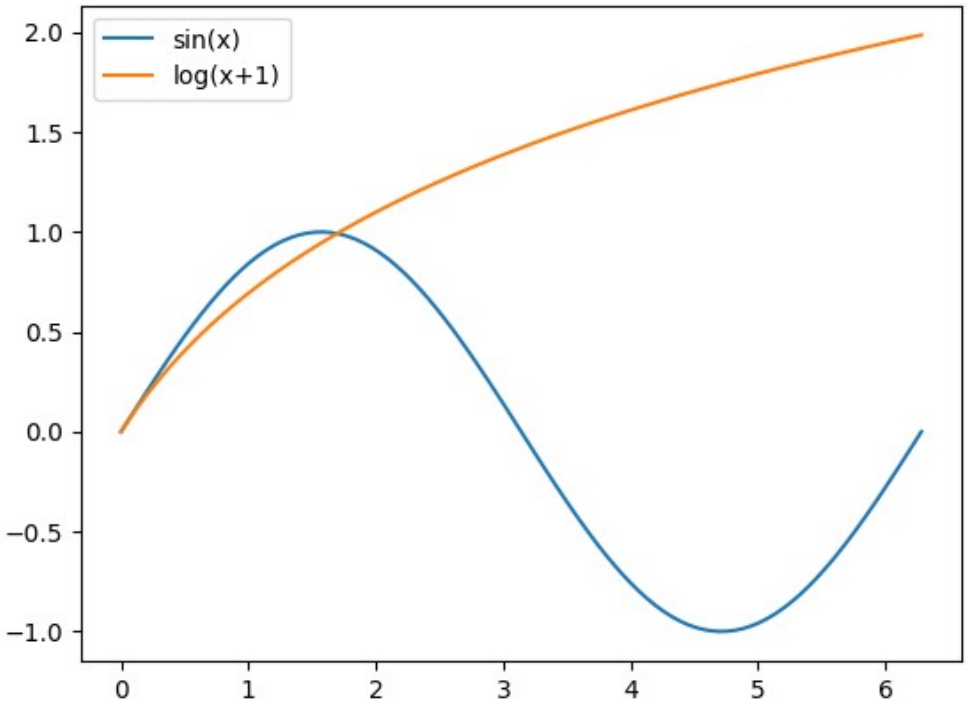


Ejercicio 2:

Representa las funciones f(x) = sin(x), g(x)=log(1+x) en el intervalo [0,2*pi], tomando 100 puntos igualmente espaciados en dicho intervalo.

```
import matplotlib as mpl

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np


df = pd.read_csv('bmw.csv')


x = np.linspace(0, 2*np.pi, 100)

plt.plot(x, np.sin(x), label= "f(x)")

plt.plot(x, np.log(1+x), label= "g(x)")

plt.legend()

plt.savefig('ej2.png')
```

Resultado:

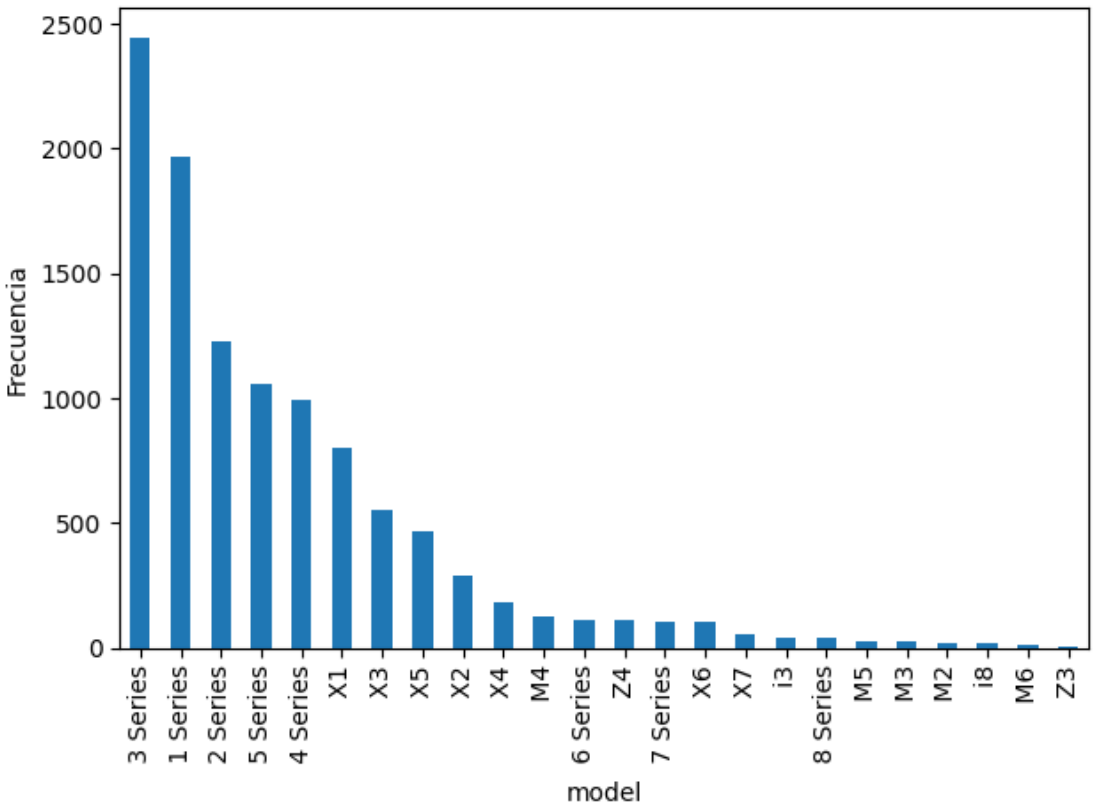


Ejercicio 3:

Crea un gráfico de barras que muestre las frecuencias de los distintos valores del atributo model de la base de datos.

```
import matplotlib as mpl

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

df = pd.read_csv('bmw.csv')

plt.xlabel("Modelos")

plt.ylabel("Frecuencia")

frecuencia = df['model'].value_counts()

frecuencia.plot(kind='bar')

plt.tight_layout()

plt.savefig('ej3.png')
```

Resultado:

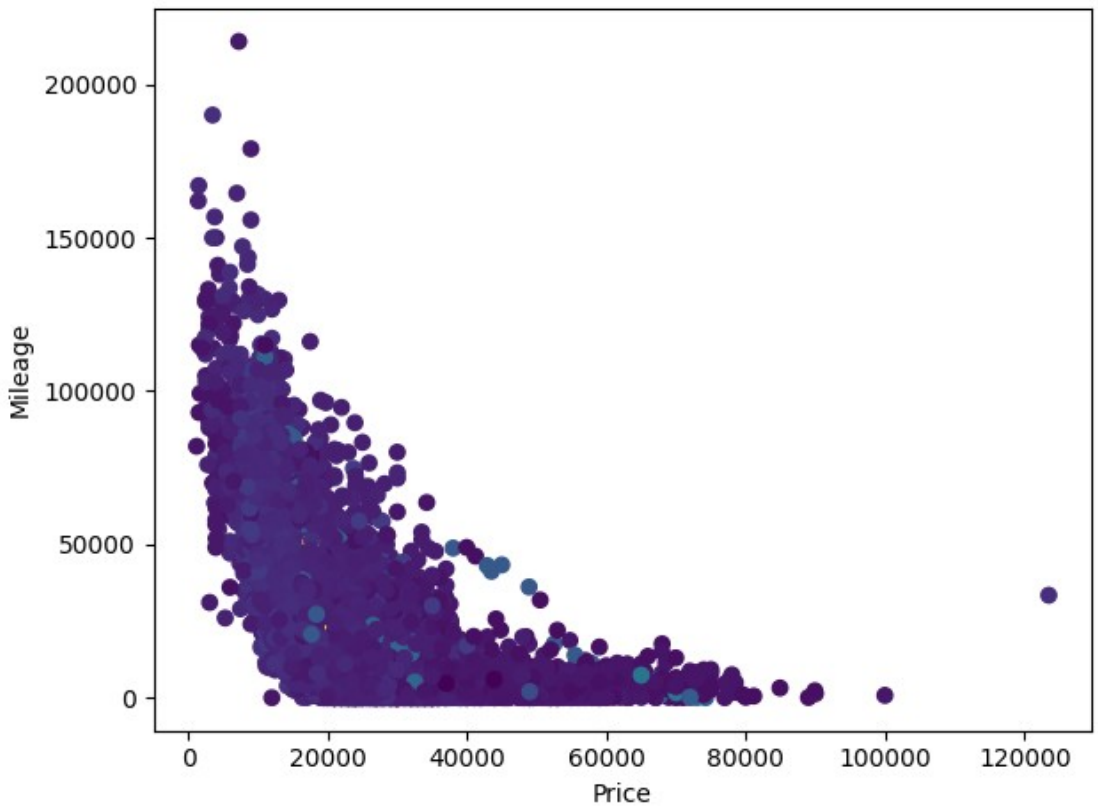


Ejercicio 4:

Crea un gráfico de dispersión que muestre los precios (atributo “price”) en el eje horizontal, los kilometrajes (atributo “mileage”) en el eje vertical, y las eficiencias (atributo “mpg”) como colores.

```
import matplotlib as mpl

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np


df = pd.read_csv('bmw.csv')

plt.xlabel("Price")

plt.ylabel("Mileage")

plt.scatter(df['price'], df['mileage'],c=df['mpg'])

plt.tight_layout()

plt.savefig('ej4.png')
```

Resultado:

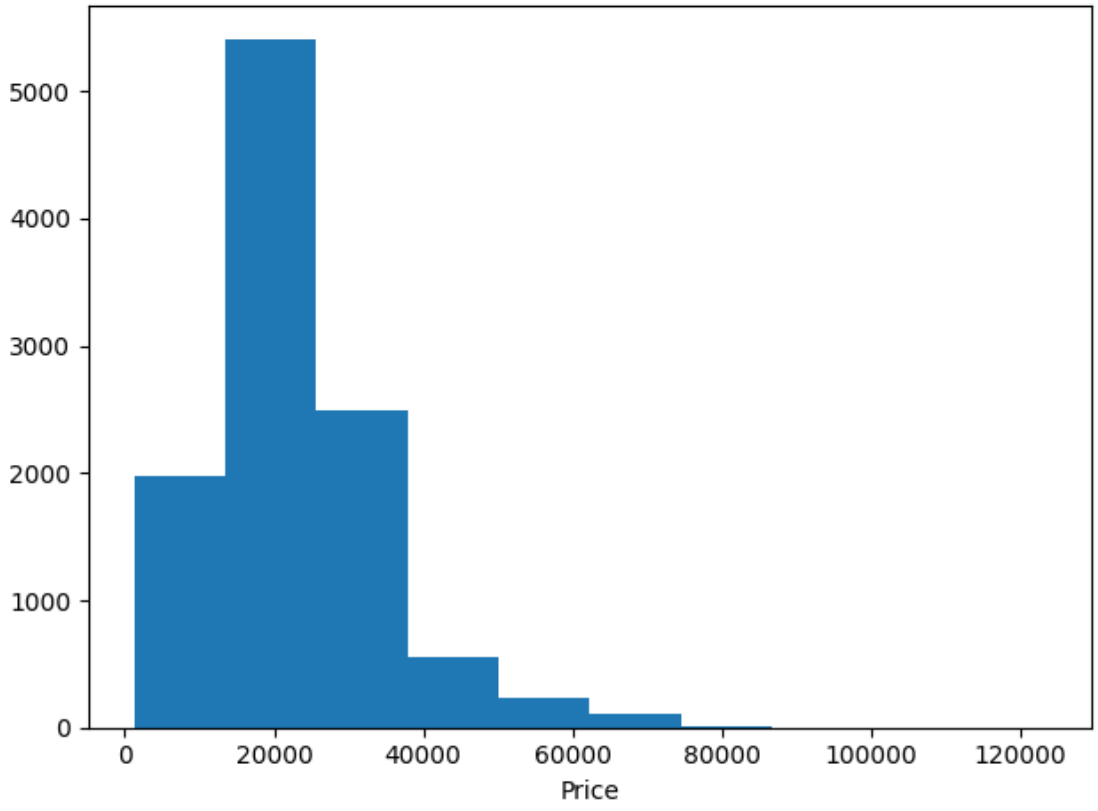


Ejercicio 5:

Crea un histograma que muestre los precios (atributo “price”) de la base de datos.

```
import matplotlib as mpl

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np


df = pd.read_csv('bmw.csv')


plt.xlabel("Price")

plt.hist(df["price"])

plt.tight_layout()

plt.savefig('ej5.png')
```

Resultado:



Ejercicio 6:

Considera la siguiente función, que calcula valores enteros entre 0 y 255 para dibujar el fractal de Mandelbrot, siendo a y b números en coma flotante que representan la parte real y la parte imaginaria de un número complejo, respectivamente:

```
MAXIMO_ITERACIONES = 80

def mandelbrot(a,b):
    c = complex(a,b)
    z = 0
    n = 0
    while abs(z) <= 2 and n < MAXIMO_ITERACIONES:
        z = z*z + c
        n += 1
    color_pixel = 255 - int(n * 255 / MAXIMO_ITERACIONES)
    return color_pixel
```

Utilizando el código anterior, dibuja un mapa de calor que represente una parte del fractal de Mandelbrot. El mapa debe tener 600 píxeles de ancho por 400 píxeles de alto, de manera que se consideren valores de la parte real entre -2 y 1, mientras que los valores de la parte imaginaria deben estar entre -1 y 1.

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

MAXIMO_ITERACIONES = 80
def mandelbrot(a,b):
    c = complex(a,b)
    z = 0
    n = 0
    while abs(z) <= 2 and n < MAXIMO_ITERACIONES:
        z = z*z + c
```

```
        n += 1
    color_pixel = 255 - int(n * 255 / MAXIMO_ITERACIONES)
    return color_pixel

entero = np.linspace(-2,1, 600)
imaginario = np.linspace(-1,1,400)
resultado = np.vectorize(mandelbrot)(entero, imaginario[:, np.newaxis])
plt.imshow(resultado)
plt.savefig('ej6.png')
```

Resultado: