Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey

Explorando bases

TC3004B.104 Inteligencia Artificial Avanzada para la Ciencia de Datos I

Profesores:

Ivan Mauricio Amaya Contreras

Blanca Rosa Ruiz Hernandez

Antonio Carlos Bento

Frumencio Olivas Alvarez

Hugo Terashima Marín

Julian Lawrence Gil Soares - A00832272

17 de Agosto de 2023

Explorando bases TC3004B.104 Inteligencia Artificial Avanzada para la Ciencia de Datos I

Profesores: Ivan Mauricio Amaya Contreras Blanca Rosa Ruiz Hernandez Antonio Carlos Bento Frumencio Olivas Alvarez Hugo Terashima Marín

Julian Lawrence Gil Soares - A00832272

27 de Agosto de 2023

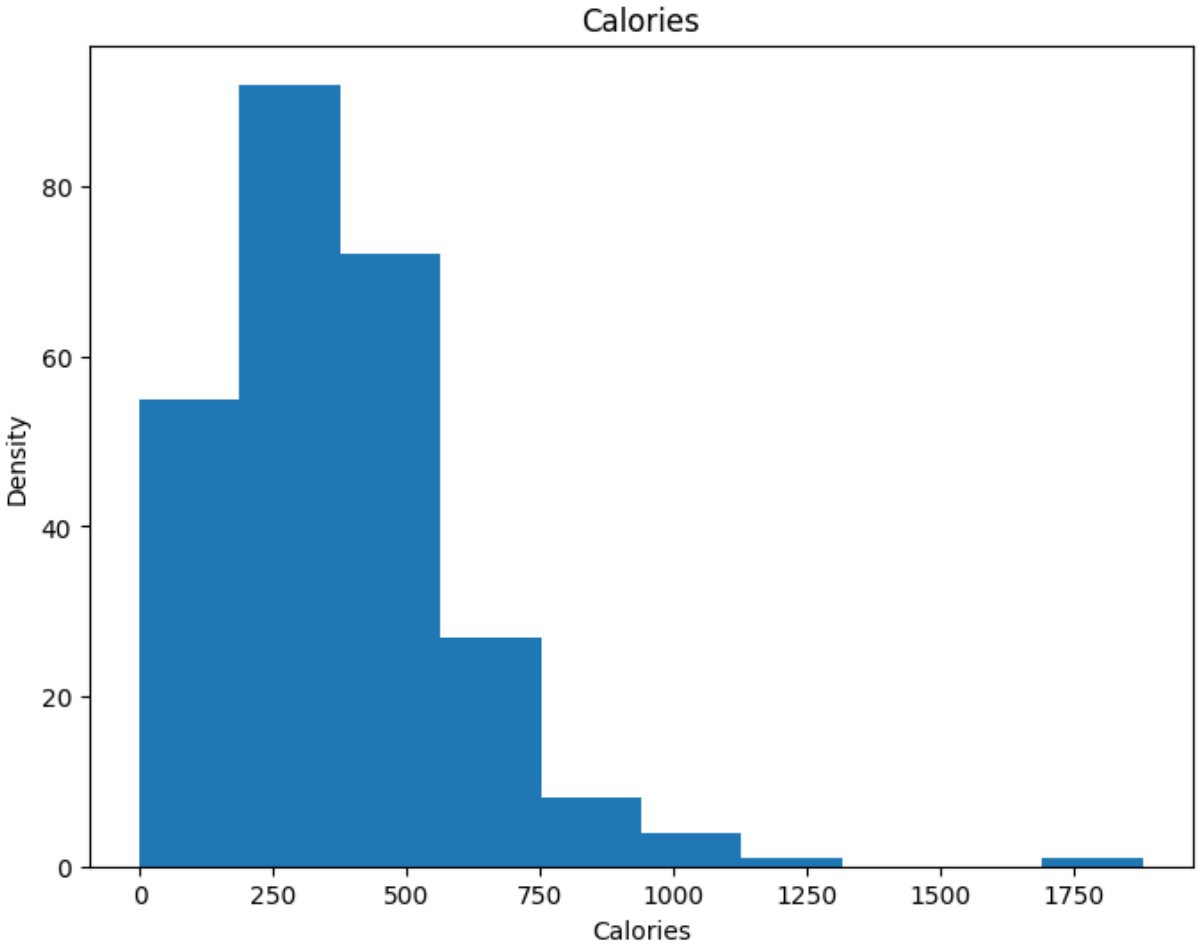
In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

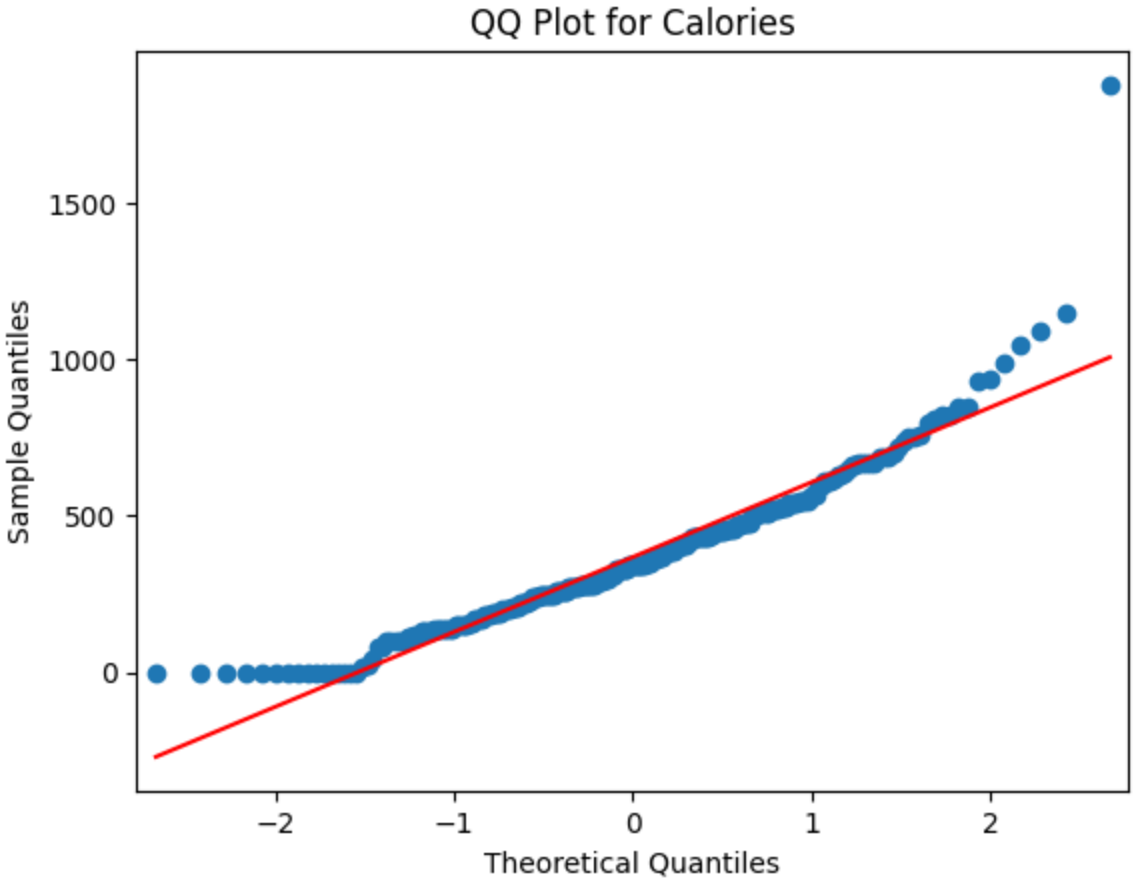
```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.
mount("/content/drive", force_remount=True).
```

In [ ]:
```
#Lectura de archivo y selecion de variables para analizar

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
from scipy import stats

data = pd.read_csv('/content/drive/MyDrive/Stats/mc-donalds-menu-1.csv')

variables_to_analyze = ['Calories', 'Protein']
```

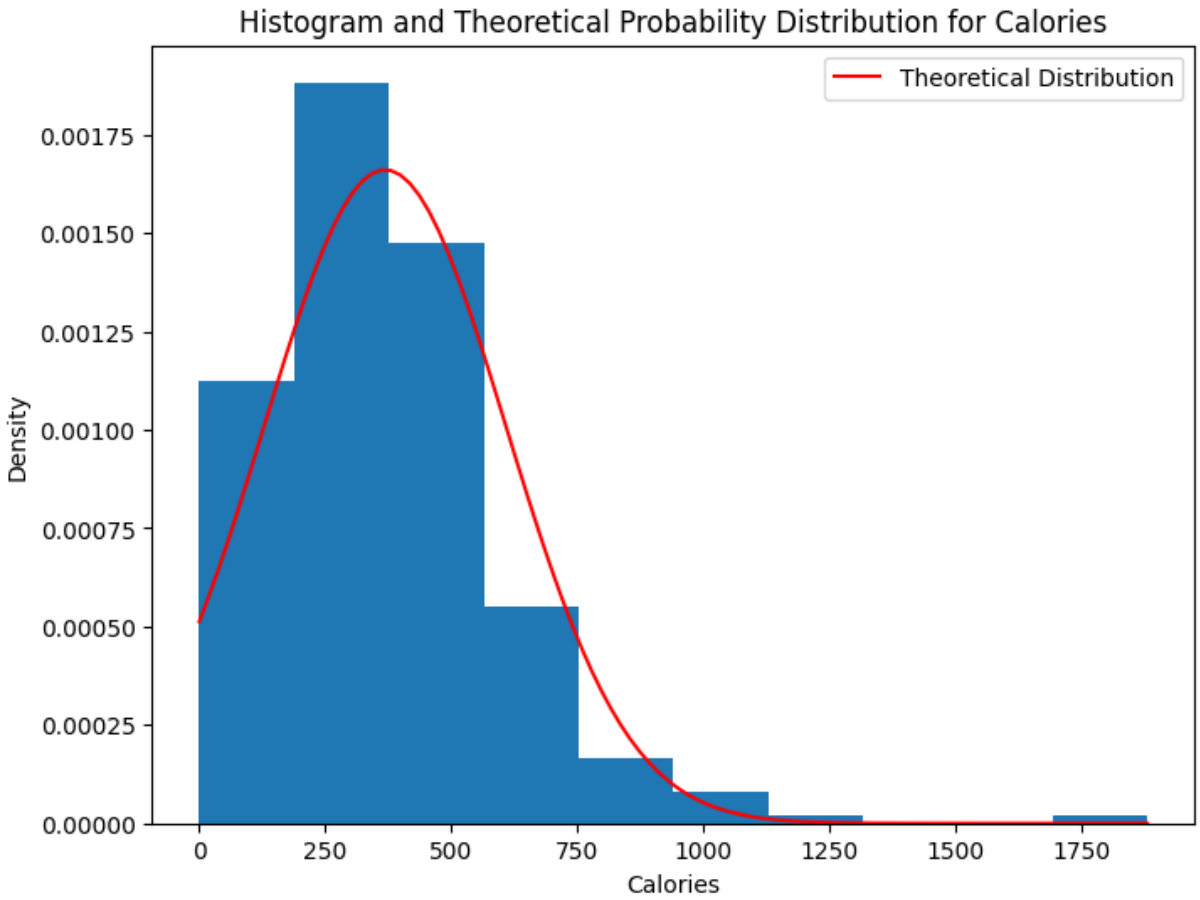
In [ ]:
```
for variable in variables_to_analyze:

    variable_data = data[variable]

    plt.figure(figsize=(8, 6))
    plt.hist(variable_data)
    plt.title(variable)
    plt.xlabel(variable)
    plt.ylabel('Density')
    plt.show()

    q25 = np.percentile(variable_data, 25)
    q75 = np.percentile(variable_data, 75)
    iqr = q75 - q25
    lower_bound = q25 - 1.5 * iqr
    upper_bound = q75 + 1.5 * iqr

    sorted_data = variable_data.sort_values()
    n = len(sorted_data)
    rank = np.arange(1, n + 1)
    expected_values = np.mean(sorted_data)
    z = (sorted_data - expected_values) / np.std(sorted_data, ddof=1)
    w = np.sum(rank * z) ** 2 / np.sum((rank - (n + 1) / 2) ** 2)

    sm.qqplot(variable_data, line='s')
    plt.title(f'QQ Plot for {variable}')
```

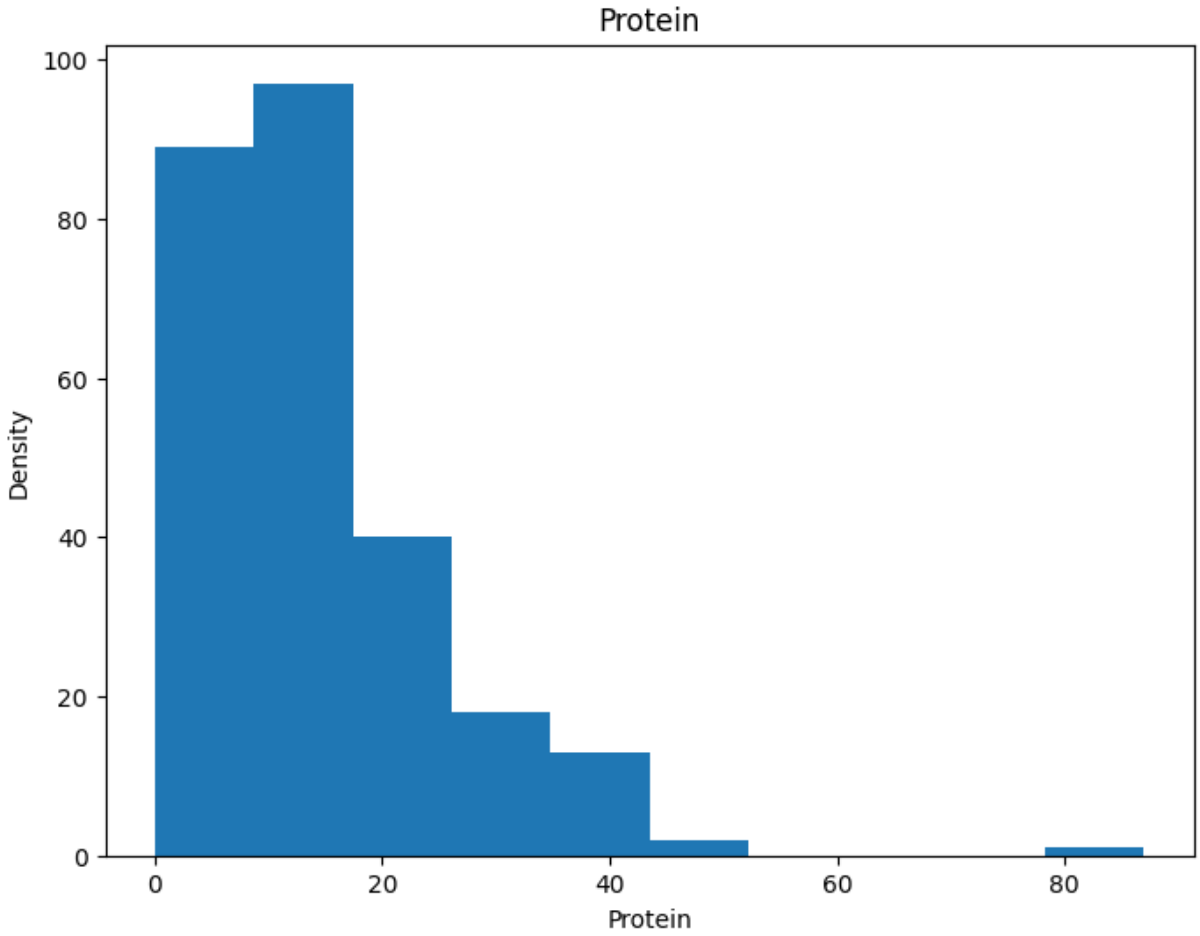
```
    plt.show()

    mean = variable_data.mean()
    median = variable_data.median()
    std = variable_data.std()
    sesgo = ((mean - median)*3) / std
    kurt = np.mean((variable_data - mean)**4) / std**4 - 3
    print('Sesgo:', sesgo)
    print('Kurtosis:', kurt)

    print('Antes:')
    print('Mean:', mean)
    print('Median:', median)

    normalized_data = (variable_data - mean) / std
    normalized_mean = normalized_data.mean()
    normalized_median = normalized_data.median()

    print('Despues:')
    print('Mean:', normalized_mean)
    print('Median:', normalized_median)
    print()


    plt.figure(figsize=(8, 6))
    plt.hist(variable_data, density=True)

    x = np.linspace(variable_data.min(), variable_data.max(), 100)
    mu = variable_data.mean()
    sigma = variable_data.std()
    pdf = (1 / (sigma * np.sqrt(2 * np.pi))) * np.exp(-0.5 * ((x - mu) / sigma)**2)
    plt.plot(x, pdf, color='red', label='Theoretical Distribution')

    plt.title(f'Histogram and Theoretical Probability Distribution for {variable}')
    plt.xlabel(variable)
    plt.ylabel('Density')
    plt.legend()
    plt.show()
```

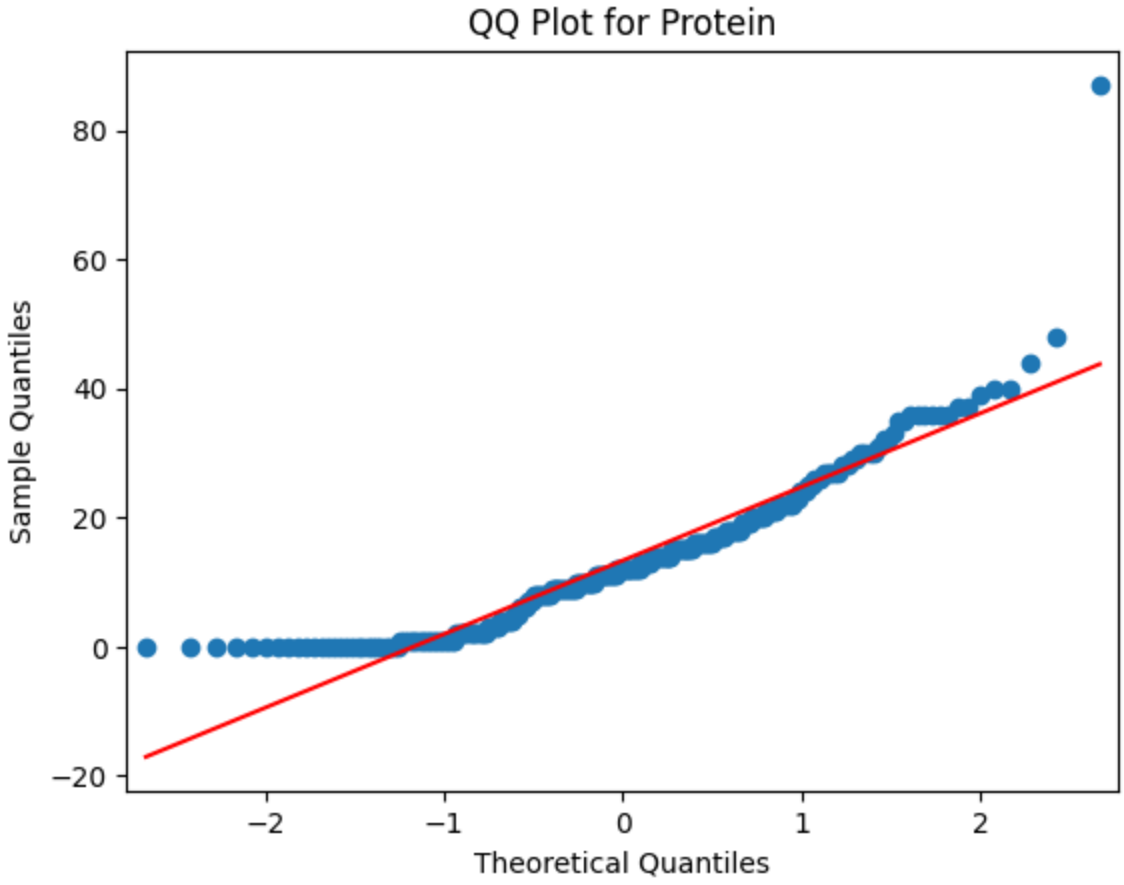

Calories
Density
0
20
40
60
80
0
250
500
750
1000
1250
1500
1750
Calories


QQ Plot for Calories
Sample Quantiles
0
500
1000
1500
−2
−1
0
1
2
Theoretical Quantiles

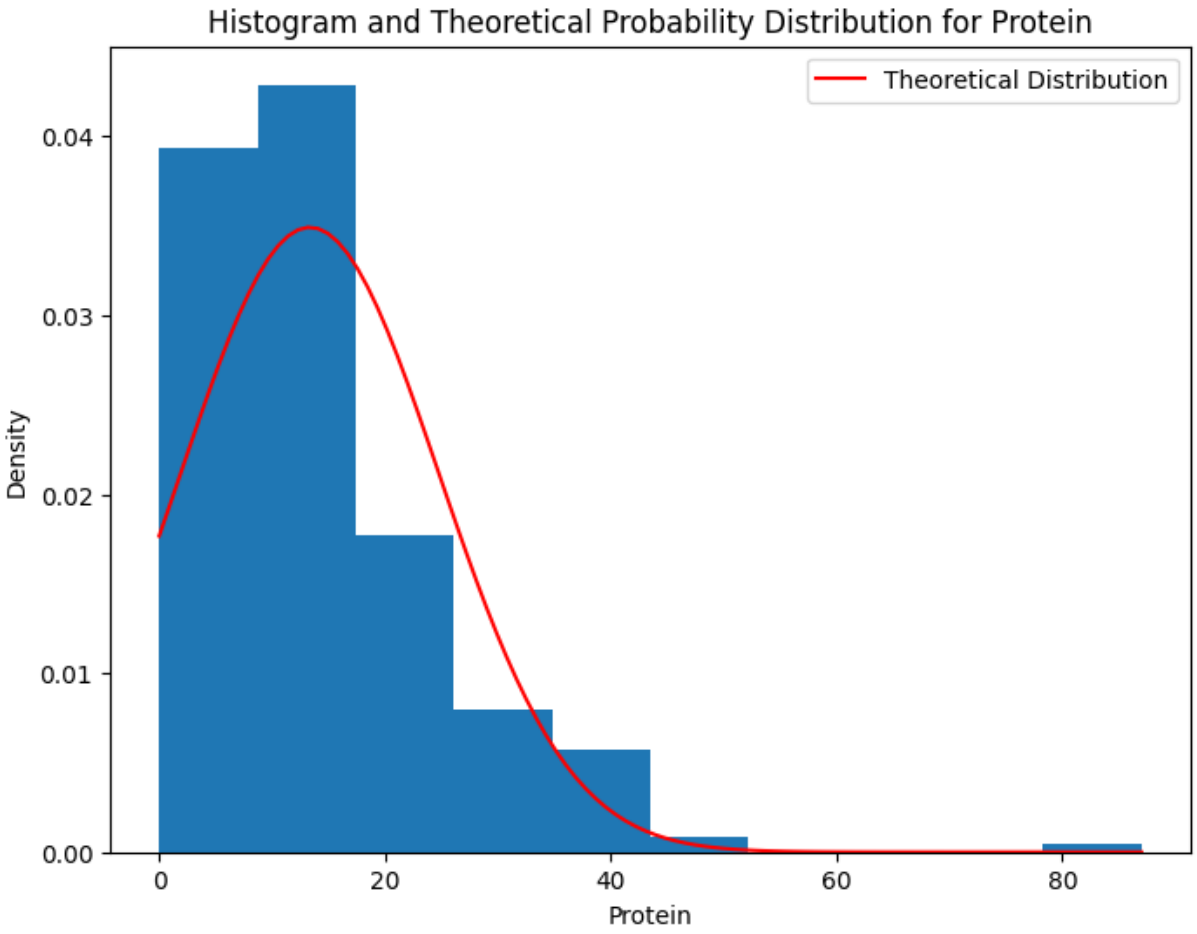
```
Sesgo: 0.3529684620328282
Kurtosis: 5.578899652449401
Antes:
Mean: 368.2692307692308
Median: 340.0
Despues:
Mean: 1.3664283380001927e-17
Median: -0.11765615401094273
```




Protein
Density
Protein
0
20
40
60
80
100


QQ Plot for Protein
Sample Quantiles
Theoretical Quantiles
−20
0
20
40
60
80
−2
−1
0
1
2

```
Sesgo: 0.3514207290732474
Kurtosis: 5.795499873186154
Antes:
Mean: 13.338461538461539
Median: 12.0
Despues:
Mean: 0.0
Median: -0.1171402430244158
```



In [ ]:
```
'''
Cuando hablamos de la normalidad de los datos nos referimos a la cercanía
que nuestros datos tienen a sus valores teóricos. Nuestra mejor herramienta
para interpretar la normalidad es usando un qq plot para observar como
nuestros datos reales se aproximan a los valores teóricos. También es muy
útil poder comparar el sesgo. El sesgo nos sirve para determinar la simetría
por ejemplo para ambas variables que analizamos encontramos un sesgo de
aproximadamente .35 así que sabemos que nuestros datos tienen buena
simetría con tendencia positiva así que nuestros datos están casi
normalizados.
'''
```

Out[ ]: '\nCuando hablamos de la normalidad de los datos nos referimos a la cercanía\nque nuestros datos tienen a sus valores teóricos. Nuestra mejor herramienta\npara interpretar la normalidad es usando un qq plot para observar como\nnuestros datos reales se aproximan a los valores teóricos. También es muy\nútil poder comparar el sesgo. El sesgo nos sirve para determinar la simetría\npor ejemplo para ambas variables que analizamos encontramos un sesgo de\naproximadamente .35 así que sabemos que nuestros datos tienen buena\nsimetría con tendencia positiva así que nuestros datos están casi\nnormalizados.\n'