

# Taller evaluable sobre la extracción, transformación y visualización de datos usando IPython

**Juan David Velásquez Henao**

jdvelasq@unal.edu.co

Universidad Nacional de Colombia, Sede Medellín

Facultad de Minas

Medellín, Colombia

**Desarrollado por**

**Diego Diaz**

diazdiego01@hotmail.com

CELEC EP Unidad de Negocio Coca Codo Sinclair Ecuador

## Instrucciones

En la carpeta 'Taller' del repositorio 'ETVL-IPython' se encuentran los archivos 'Precio\_BolsaNacional(\$kwh)\_\*\*.xls' en formato de Microsoft Excel, los cuales contienen los precios históricos horarios de la electricidad para el mercado eléctrico Colombiano entre los años 1995 y 2017 en COL-PESOS/kWh. A partir de la información suministrada resuelva los siguientes puntos usando el lenguaje de programación Python.

## Listado de Variables

**w** = Matriz procesada para presentacion.

**wmain** = Matriz de datos sin columnas adicionales

**wmedia** = Matriz de Precios Promedios Diarios

**wmax** = Matriz de Precios Maximo por mes

**wmin** = Matriz de Precios Minimo por mes

**wmedm** = Matriz de Precios Promedios por mes

**wrepI** = Matriz de horas de Precio maximo

**wrepI1** = Matriz de Frecuencia en las horas en las cuales se presentan los precios maximos para los dias laborables

**wrep1s** = Matriz de Frecuencia en las horas en las cuales se presentan los precios maximos para los dias sabados

**wrep1d** = Matriz de Frecuencia en las horas en las cuales se presentan los precios maximos para los dias domingos

**wminanio** = Matriz de precios minimos por año

**pmes** = Matriz de Matriz de Precios Promedios por mes con repitencia en los demas dias del mes para graficar en la misma dimension de los precios diarios

## Preguntas

Type *Markdown* and *LaTeX*:  $\alpha^2$

1.-- Lea los archivos y cree una tabla única concatenando la información para cada uno de los años. Imprima el encabezamiento de la tabla usando `head()`.

```
In [1]: import pandas as pd  
import numpy as np  
# Se importa pandas y numpy como Librerias base para procesar datos
```

```
In [2]: x=[]
w=[]
wmain=[]

for n in range(1995,2018):
    if n<2000:
        sk=3
    else:
        sk=2
    namefile='Precio_Bolsa_Nacional_($kwh)_' + str(n)
    if n>=2016:
        namefile += '.xls'
    else:
        namefile += '.xlsx'
    y=pd.read_excel(namefile,skiprows=sk)
    y=y[list(range(0,25))]
    x.append(y)
w=pd.concat(x)
w.head()
#Leer Los archivos en excel con un for para Leer todos y con append se ubica uno
#luego con concat se unifica los dataframes en una sola matriz 'w'
```

Out[2]:

	Fecha	0	1	2	3	4	5	6	7	8	...	14	15	16
0	1995-07-20	NaN	1.073	1.073	1.073	1.073	1.073	1.073	1.073	1.074	...	1.073	1.073	1.073
1	1995-07-21	1.073	1.000	1.000	1.000	1.000	1.000	5.000	6.000	6.000	...	5.000	1.000	1.000
2	1995-07-22	1.073	1.073	1.000	1.000	1.000	1.073	1.303	1.303	1.303	...	1.073	1.000	1.000
3	1995-07-23	1.073	1.000	1.000	1.000	1.000	1.000	0.100	1.000	1.000	...	1.000	0.100	0.100
4	1995-07-24	1.000	1.000	0.990	1.000	1.000	1.073	3.000	3.000	3.000	...	1.073	1.073	3.000

5 rows × 25 columns



2.-- Compute e imprima el número de registros con datos faltantes.

```
In [3]: len(w)-len(w.dropna())
# Se revisa de la longitud de la matriz principal cuantos elementos estan en blanco
```

Out[3]: 28

3.-- Compute e imprima el número de registros duplicados.

```
In [4]: z=w[w.duplicated()]
len(z)
# Se revisa cuantos elementos se encuentran duplicados.
```

Out[4]: 67

4.-- Elimine los registros con datos duplicados o datos faltantes, e imprima la cantidad de registros que quedan (registros completos).

```
In [5]: w=w.dropna()
w=w.drop_duplicates()
wmain=w.drop_duplicates()
len(w)
#Se elimina a La matriz principal los elementos en blanco y Los elementos duplicados
```

Out[5]: 7875

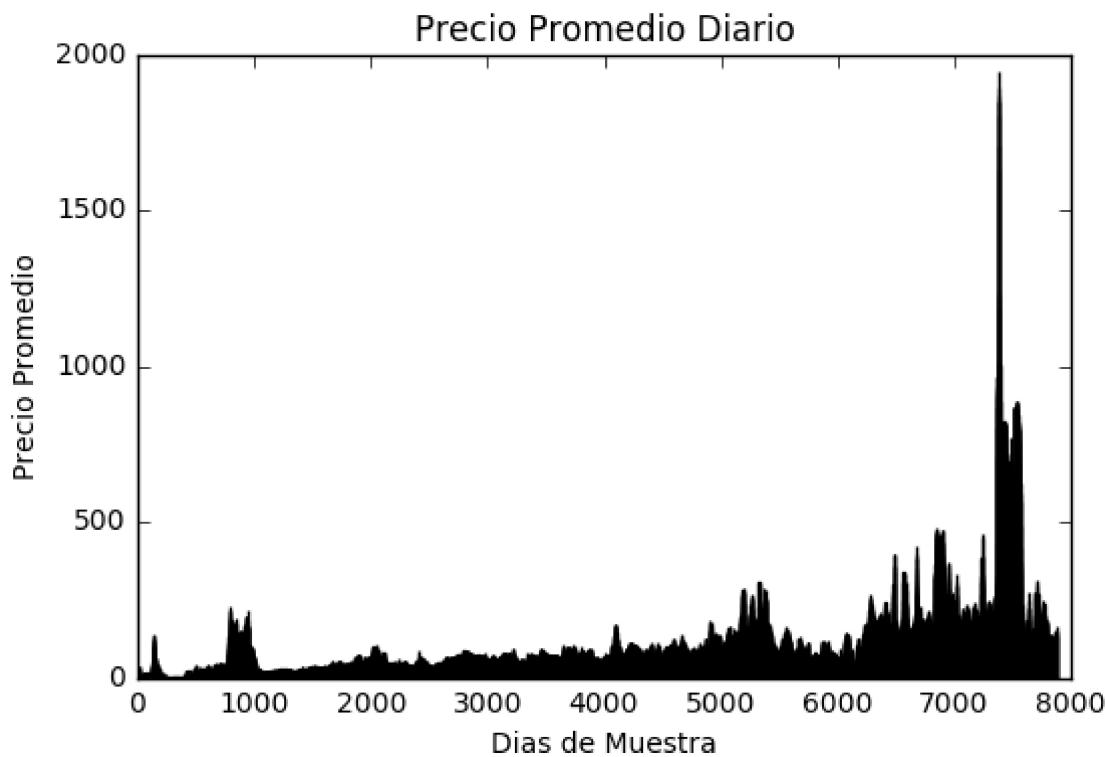
5.-- Compute y grafique el precio promedio diario.

```
In [6]: wmedia=wmain.mean(axis=1)
wmedia
w['Promedio']=wmedia
# Se encuentra la matriz de Los datos promedio por filas con axis = 1
# Se incluye una columna 'Promedio' en la matriz de visualizacion
```

```
In [7]: import matplotlib as mp
import matplotlib.pyplot as plt
%matplotlib inline
# Se importa Las Librerias de graficos
```

```
In [8]: plt.title("Precio Promedio Diario")
plt.xlabel("Dias de Muestra")
plt.ylabel("Precio Promedio")
plt.bar(range(len(wmedia)),wmedia)
#Se grafica la matriz de Precios Promedios Diarios
```

Out[8]: <Container object of 7875 artists>



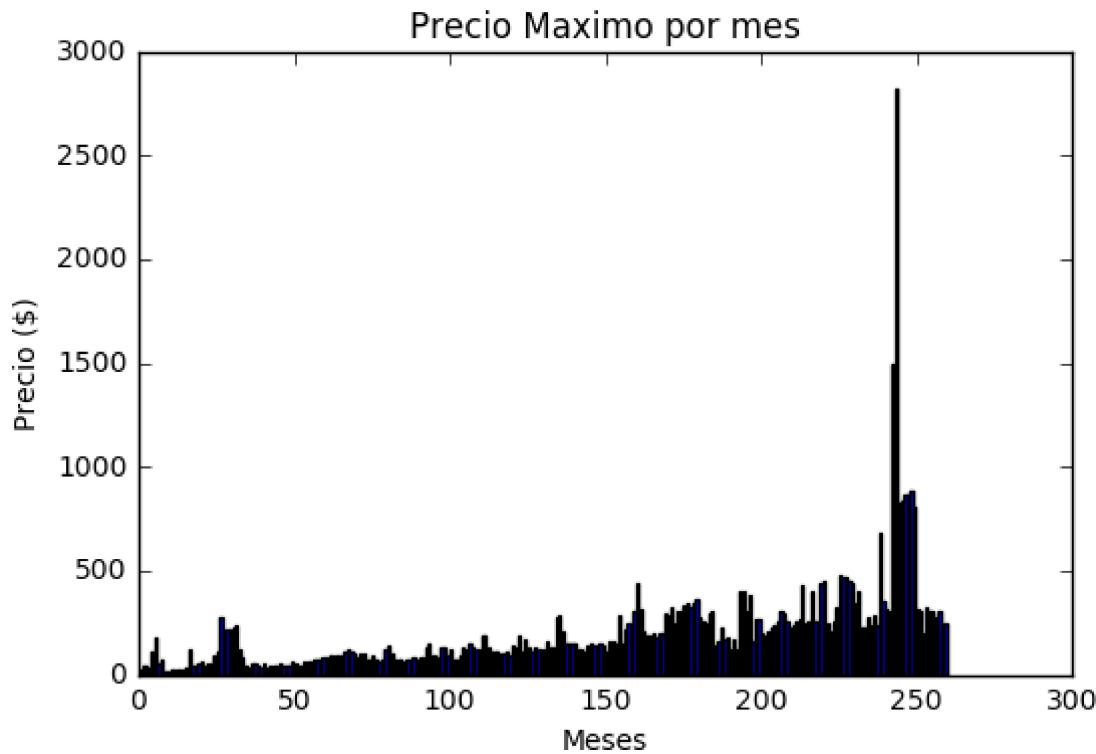
6.-- Compute y grafique el precio máximo por mes.

```
In [9]: w1=[]
for n in range(len(wmain['Fecha'])):
    w1.append(str(wmain.iloc[n,0])[0:7])
w['key']=w1
wmain['key']=w1
# Se crea una columna únicamente de año y mes para filtrarla luego
```

```
In [10]: wmax=wmain.groupby('key').max()
wmax=wmax.max(axis=1)
wmax.head()
# Se agrupa por año y mes y se calcula el valor max,
# luego se calcula el precio max de la fila para tener un solo valor por año y mes
```

```
Out[10]: key
1995-07      22.500
1995-08      40.000
1995-09      40.572
1995-10      33.700
1995-11     111.907
dtype: float64
```

```
In [11]: plt.title("Precio Maximo por mes")
plt.xlabel("Meses")
plt.ylabel("Precio ($)")
plt.bar(range(len(wmax)),wmax)
plt.show()
# Se grafica el precio maximo por mes
```

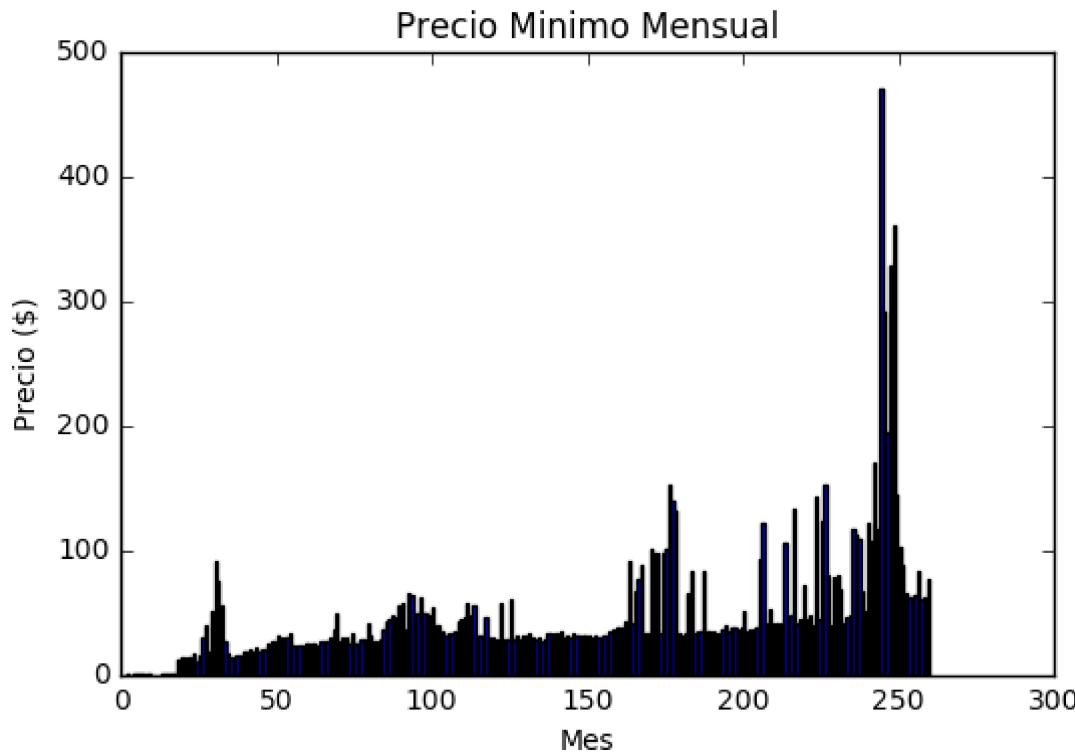


7.-- Compute y grafique el precio mínimo mensual.

```
In [12]: wmin=wmain.groupby('key').min()
wmin=wmin.min(axis=1)
wmin.head()
# Se agrupa por anio y mes y se calcula el valor min,
#luego se calcula el precio min de la fila para tener un solo valor por anio y me
```

```
Out[12]: key
1995-07    0.000
1995-08    0.000
1995-09    1.073
1995-10    0.000
1995-11    1.072
dtype: float64
```

```
In [13]: plt.title("Precio Minimo Mensual")
plt.xlabel("Mes")
plt.ylabel("Precio ($)")
plt.bar(range(len(wmin)),wmin)
plt.show()
# Se grafica el precio minimo mensual
```

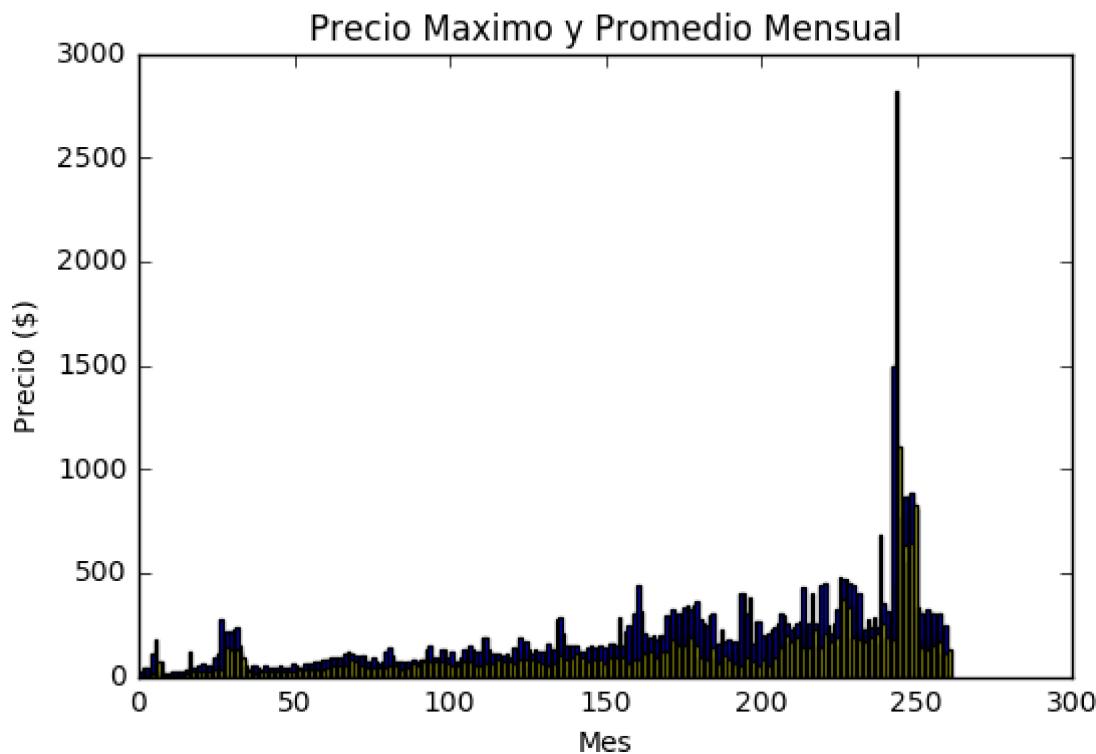


**8.--** Haga un gráfico para comparar el precio máximo del mes (para cada mes) y el precio promedio mensual.

```
In [14]: wmedm=wmain.groupby('key').mean()
wmedm=wmedm.mean(axis=1)
# Se agrupa por anio y mes y se calcula el valor promedio,
#luego se calcula el precio medio de la fila para tener un solo valor por anio y
```

```
In [16]: import plotly
import plotly.plotly as py
import plotly.tools as tls
import numpy as np
# Se importa las librerias de graficos
```

```
In [17]: plt.title("Precio Maximo y Promedio Mensual")
plt.xlabel("Mes")
plt.ylabel("Precio ($)")
X = np.arange(len(wmax))
plt.bar(X + 0.00, wmax, color = 'b', width = 1)
plt.bar(X + 1, wmedm, color = 'y', width = 1)
plt.show()
# Se grafica el Precio maximo y promedio mensual en La misma grafica con diferentes colores.
```



9.-- Haga un histograma que muestre a que horas se produce el máximo precio diario para los días laborales.

```
In [18]: import datetime
import calendar
import metakernel
# Se importa librerias de tiempo, aunque no se las uso.
```

```
In [19]: laborable=['L','S','D','L','L','L','L']*int(len(w)/7)
w['Laborable']=laborable
wmain['Laborable']=laborable
wmaxd=wmain.max(axis=1)
w['MaxDia']=wmaxd
w[w['Laborable']=='L'].head()
# Se crea una serie de 7 dias Laborables, sabados y domingos para incluir en La t
# Se crea La columna 'MaxDia'
```

Out[19]:

	<b>Fecha</b>	0	1	2	3	4	5	6	7	8	...	18	19	20
1	1995-07-21	1.073	1.00	1.000	1.00	1.00	1.000	5.000	6.000	6.000	...	12.000	16.67	11.929
4	1995-07-24	1.000	1.00	0.990	1.00	1.00	1.073	3.000	3.000	3.000	...	18.630	22.50	9.256
5	1995-07-25	0.990	0.99	0.989	0.99	0.99	1.073	1.263	1.263	1.263	...	1.263	1.50	1.263
6	1995-07-26	0.500	0.50	0.500	0.50	0.50	0.990	1.073	1.073	1.073	...	1.073	8.00	1.073
7	1995-07-27	0.500	0.50	0.500	0.50	0.50	0.500	0.500	0.500	0.500	...	1.073	1.50	0.990

5 rows × 29 columns

Out[20]:

```
windex = ['line_{}'.format(n) for n in range(len(w))]
w.index = windex
wlab=w[w['Laborable']=='L']
windex1 = ['line_{}'.format(n) for n in range(len(wlab))]
wlab.index = windex1
wlab.head()
```

Out[20]:

	<b>Fecha</b>	0	1	2	3	4	5	6	7	8	...	18	19	20
line_0	1995-07-21	1.073	1.00	1.000	1.00	1.00	1.000	5.000	6.000	6.000	...	12.000	16.67	11
line_1	1995-07-24	1.000	1.00	0.990	1.00	1.00	1.073	3.000	3.000	3.000	...	18.630	22.50	9.
line_2	1995-07-25	0.990	0.99	0.989	0.99	0.99	1.073	1.263	1.263	1.263	...	1.263	1.50	1.
line_3	1995-07-26	0.500	0.50	0.500	0.50	0.50	0.990	1.073	1.073	1.073	...	1.073	8.00	1.
line_4	1995-07-27	0.500	0.50	0.500	0.50	0.50	0.500	0.500	0.500	0.500	...	1.073	1.50	0.

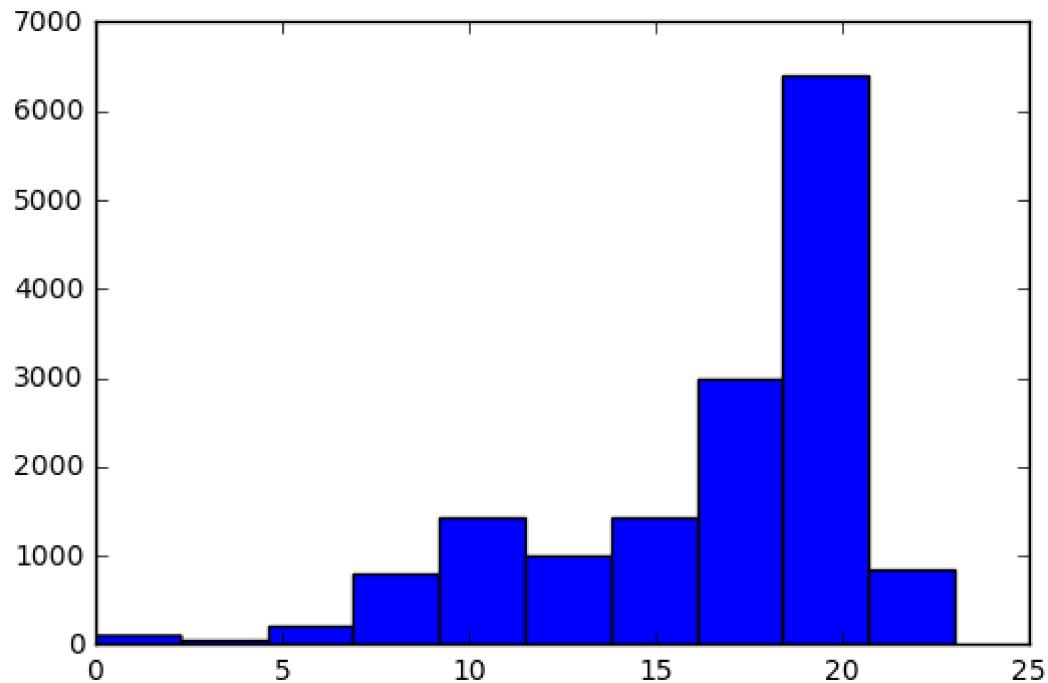
5 rows × 29 columns

```
In [21]: wmaxh=[]
wmaxh1=[]
for n in range(len(wlab)):
    wmaxh1=pd.Series(wlab.loc['line_'+str(n)]).values[1:25]
    wmaxh.append ([i for i,e in enumerate(wmaxh1) if e == max(wmaxh1)])
wmaxh
# Se filtra por dia las horas en las cuales se presentan las horas de precio maxi
```

```
Out[21]: [[19],
[19],
[19],
[19],
[19],
[19],
[10, 11, 18, 19, 20],
[19],
[9],
[6, 13, 16, 21],
[6, 7, 8, 9, 10, 11, 12, 18, 19, 20, 21],
[5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22],
[19],
[7, 19],
[10, 11],
[19],
[5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22],
[6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22],
[19],
[19],
```

```
In [22]: import statistics
import random
from matplotlib.pyplot import hist, show
# Se importa librerias de estadisticas y random para utilizarlo en los histograma
```

```
In [23]: wrepl=[]
for n in range (len(wmaxh)):
    for i in range(len(wmaxh[n])):
        wrepl.append(wmaxh[n][i])
hist(wrepl)
show()
# Se calcula una matriz que unifica en un solo arreglo Las horas que pertenecen e
```

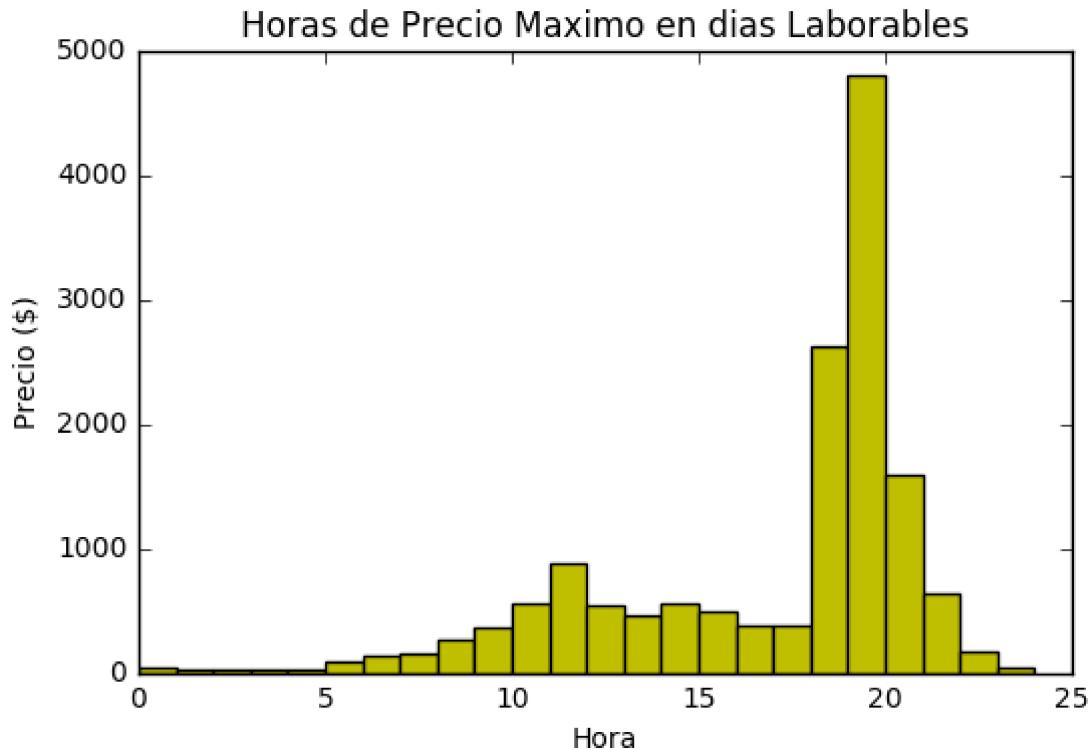


```
In [39]: wrepl
```

```
Out[39]: [19,
19,
19,
19,
19,
19,
10,
11,
18,
19,
20,
19,
9,
6,
13,
16,
21,
6,
7,
8,
0]
```

```
In [24]: wrepl1=[]
for n in range(24):
    wrepl1.append(wrepl.count(n))
horas=range(24)
# Se calcula una matriz de frecuencia en las horas en las que se produce el precio
```

```
In [25]: plt.title("Horas de Precio Maximo en dias Laborables")
plt.xlabel("Hora")
plt.ylabel("Precio ($)")
plt.bar(horas, wrepl1, color = 'y', width = 1)
plt.show()
# Se grafica en formato de barras la matriz de frecuencia de repeticion de horas
```



```
In [26]: wrep11  
# Se indica las veces que repite las horas del dia que presentan el precio maximo
```

```
Out[26]: [48,  
29,  
24,  
19,  
28,  
88,  
130,  
161,  
271,  
362,  
563,  
875,  
542,  
452,  
554,  
493,  
377,  
372,  
2616,  
4806,  
1591,  
638,  
163,  
48]
```

**10.--** Haga un histograma que muestre a que horas se produce el máximo precio diario para los días sábado.

```
In [27]: wsab=w[w['Laborable']=='S']
windex2 = ['line_{}'.format(n) for n in range(len(wsab))]
wsab.index = windex2
wsab.head()
# Se crea una matriz que filtra los días sábados de la matriz principal
```

Out[27]:

	Fecha	0	1	2	3	4	5	6	7	8	...	18
line_0	1995-07-22	1.073	1.073	1.000	1.000	1.000	1.073	1.303	1.303	1.303	...	1.303
line_1	1995-07-29	1.000	1.000	0.000	0.000	1.000	0.000	1.070	1.070	1.070	...	1.070
line_2	1995-08-05	2.000	2.000	2.000	1.073	2.000	2.500	2.500	2.558	2.558	...	2.558
line_3	1995-08-12	30.000	15.000	1.074	15.000	17.500	30.000	30.000	31.000	40.000	...	40.000
line_4	1995-08-19	1.000	1.000	1.000	1.000	1.074	1.074	1.000	1.000	1.000	...	1.000

5 rows × 29 columns

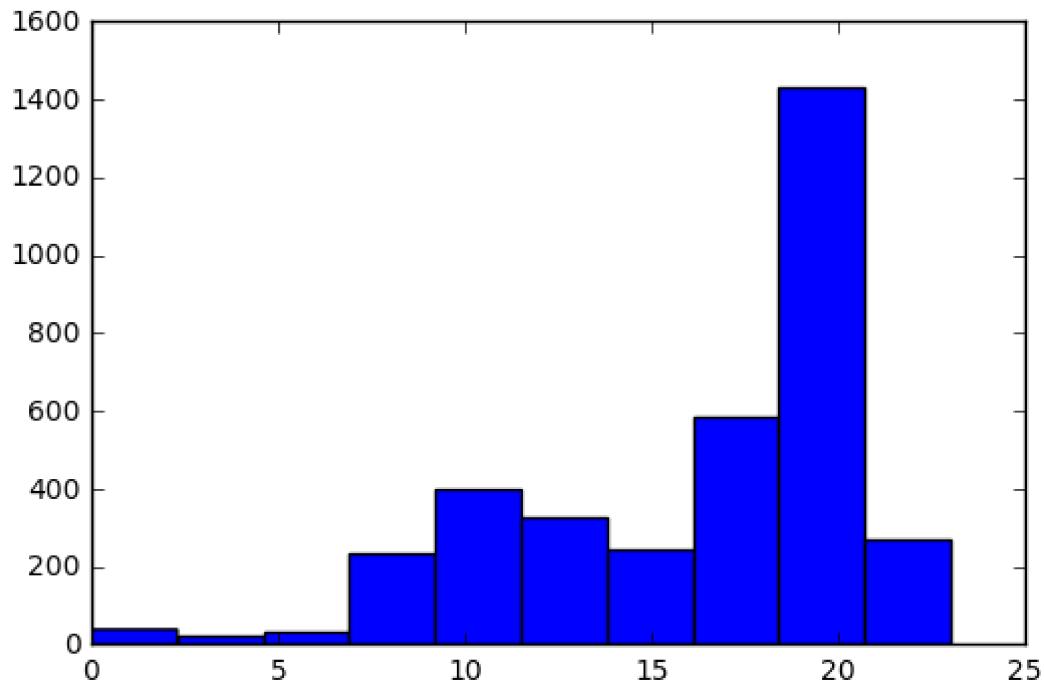
In [28]:

```
wmaxhs=[]
wmaxh1s=[]
for n in range(len(wsab)):
    wmaxh1s=pd.Series(wsab.loc['line_'+str(n)].values[1:25])
    wmaxhs.append ([i for i,e in enumerate(wmaxh1s) if e == max(wmaxh1s)])
wmaxhs
# Se calcula una matriz que unifica en un solo arreglo las horas que pertenecen a
```

Out[28]:

```
[[19, 20],
 [12, 20],
 [7, 8, 9, 10, 11, 12, 13, 18, 19, 20, 21],
 [8, 11, 12, 18, 19, 20],
 [22, 23],
 [19],
 [19],
 [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22],
 [7, 8, 9, 10, 11, 12, 18, 19, 20, 21],
 [7, 8, 9, 10, 11, 12, 13, 17, 18, 19, 20, 21, 22],
 [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 18, 19, 20, 21, 22, 23],
 [19],
 [18],
 [18],
 [6, 7, 8, 9, 10, 11, 18, 19, 20, 21, 22],
 [8, 9, 10, 11, 18, 19, 20, 21, 22],
 [8, 9, 10, 11, 12, 18, 19, 20],
 [18, 19],
 [18, 19, 20, 21, 22, 23],
```

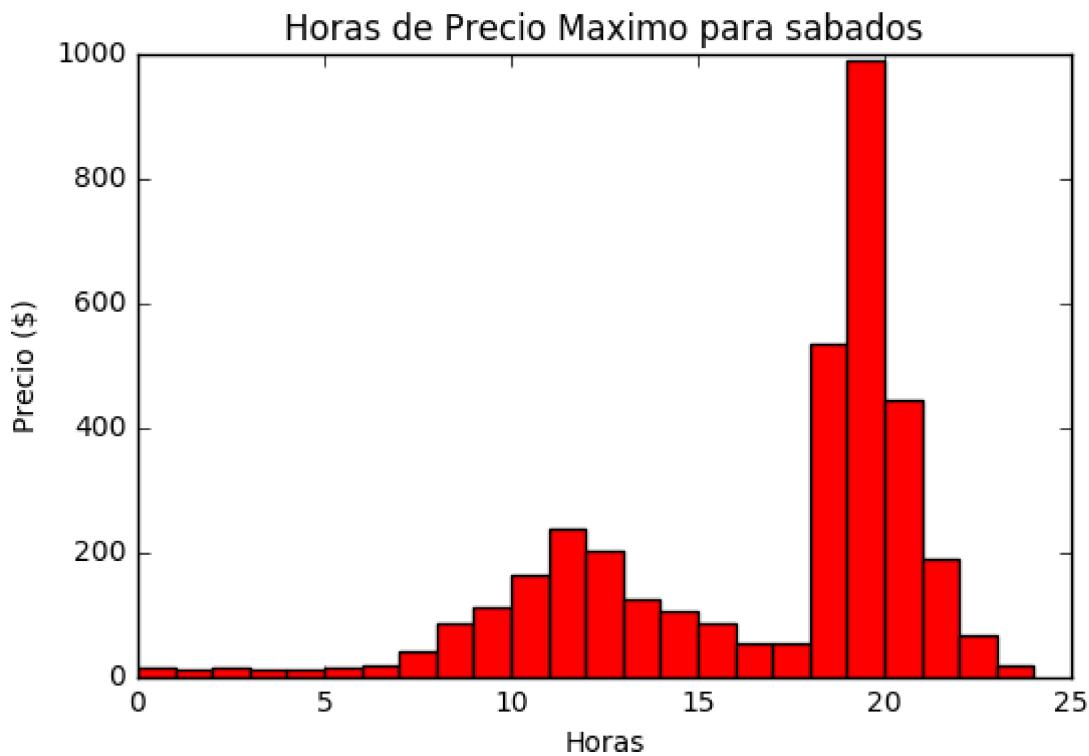
```
In [29]: wrepls=[]
for n in range (len(wmaxhs)):
    for i in range(len(wmaxhs[n])):
        wrepls.append(wmaxhs[n][i])
hist(wrepls)
show()
# Se calcula una matriz que unifica en un solo arreglo Las horas que pertenecen e
```



```
In [30]: wrepl1s=[]
for n in range(24):
    wrepl1s.append(wrepls.count(n))
wrepl1s
# Se calcula una matriz de frecuencia en las horas en las que se produce el precio
```

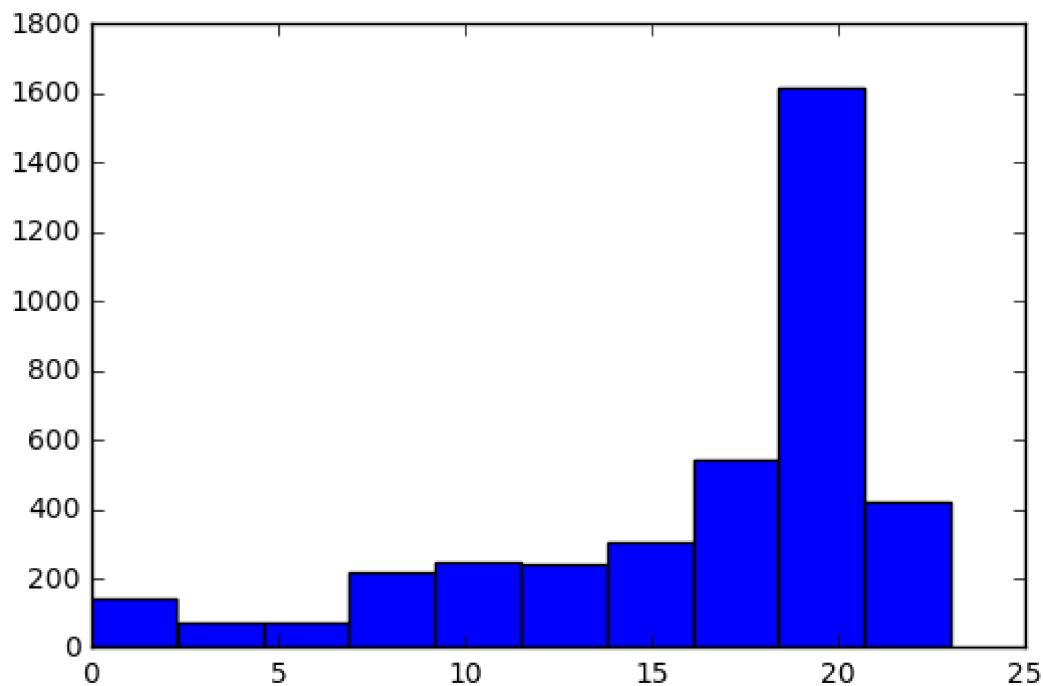
```
Out[30]: [15,
11,
13,
11,
12,
13,
19,
39,
86,
110,
163,
237,
202,
125,
106,
87,
54,
54,
533,
988,
443,
188,
67,
18]
```

```
In [31]: plt.title("Horas de Precio Maximo para sabados")
plt.xlabel("Horas")
plt.ylabel("Precio ($)")
plt.bar(horas, wrep1s, color = 'r', width = 1)
plt.show()
# Se grafica en formato de barras la matriz de frecuencia de repeticion de horas
```



**11.--** Haga un histograma que muestre a que horas se produce el máximo precio diario para los días domingo.

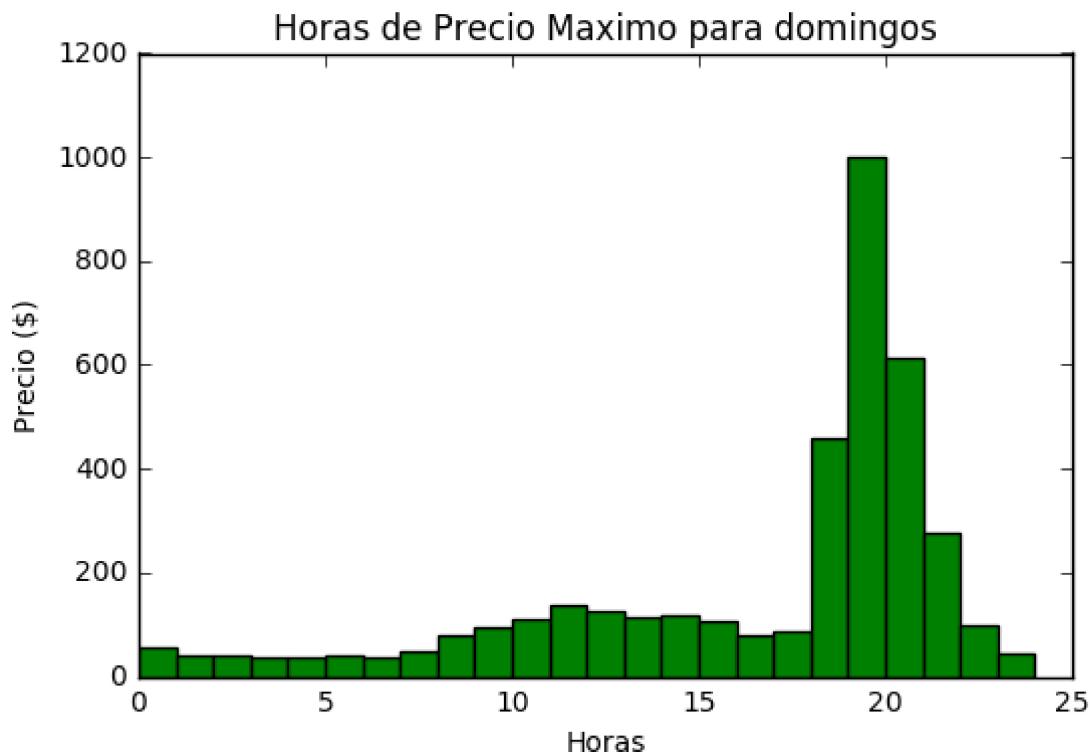
```
In [32]: wdom=w[w['Laborable']=='D']
windex3 = ['line_{}'.format(n) for n in range(len(wdom))]
wdom.index = windex3
wmaxhd=[]
wmaxh1d=[]
for n in range(len(wdom)):
    wmaxh1d=pd.Series(wdom.loc['line_'+str(n)].values[1:25])
    wmaxhd.append ([i for i,e in enumerate(wmaxh1d) if e == max(wmaxh1d)])
wrepld=[]
for n in range (len(wmaxhd)):
    for i in range(len(wmaxhd[n])):
        wrepld.append(wmaxhd[n][i])
hist(wrepld)
show()
# Se calcula una matriz que filtra por dias domingos
# Se cambia los indices
# Se filtra para sacar por dia las horas en Los cuales aparece el precio maximo e
# Se crea una matriz que presenta la frecuencia de repeticion de horas en las cuad
```



```
In [33]: wrep1d=[]
for n in range(24):
    wrep1d.append(wrep1d.count(n))
wrep1d
# Se presenta La matriz de frecuencia en La repeticion de veces en Las horas a La
```

```
Out[33]: [57,
42,
40,
37,
38,
39,
36,
47,
78,
95,
110,
137,
124,
115,
119,
108,
79,
86,
458,
1002,
614,
277,
97,
45]
```

```
In [34]: plt.title("Horas de Precio Maximo para domingos")
plt.xlabel("Horas")
plt.ylabel("Precio ($)")
plt.bar(horas, wrep1d, color = 'g', width = 1)
plt.show()
# Se grafica en formato de barras la frecuencia de repeticion de horas con precio
```



**12.--** Imprima una tabla con la fecha y el valor más bajo por año del precio de bolsa.

```
In [35]: anio=[]
for n in range(len(wmain['Fecha'])):
    anio.append(str(wmain.iloc[n,0])[0:4])
w['Anio']=anio
wmain['Anio']=anio
wminanio=wmain.groupby('Anio').min()
wminanio=wminanio.min(axis=1)
wminanio
# Se crea una columna adicional que se llama 'Anio' para filtrar por anio el precio
# Se agrupa y se calcula el precio promedio por anio
```

```
Out[35]: Anio
1995      0.000000
1996      0.000000
1997     10.882310
1998     13.847330
1999     18.359530
2000     21.531167
2001     24.822879
2002     26.777682
2003     37.013438
2004     32.252998
2005     27.581415
2006     26.714797
2007     30.173824
2008     29.199135
2009     32.892503
2010     32.024957
2011     33.291100
2012     34.988099
2013     40.115746
```

13.-- Haga una gráfica en que se muestre el precio promedio diario y el precio promedio mensual.

```
In [36]: windexmain = ['{}'.format(n) for n in range(len(wmain))]
wmain.index = windexmain
del wmain['Anio']
wmain.head()
#Se cambia Los indices de La matriz principal y se elimina La columna de anio
```

Out[36]:

	Fecha	0	1	2	3	4	5	6	7	8	...	16	17	18
0	1995-07-21	1.073	1.000	1.000	1.00	1.00	1.000	5.000	6.000	6.000	...	1.000	5.000	12.000
1	1995-07-22	1.073	1.073	1.000	1.00	1.00	1.073	1.303	1.303	1.303	...	1.000	1.000	1.303
2	1995-07-23	1.073	1.000	1.000	1.00	1.00	1.000	0.100	1.000	1.000	...	0.100	1.000	1.238
3	1995-07-24	1.000	1.000	0.990	1.00	1.00	1.073	3.000	3.000	3.000	...	3.000	2.000	18.630
4	1995-07-25	0.990	0.990	0.989	0.99	0.99	1.073	1.263	1.263	1.263	...	1.073	1.073	1.263

5 rows × 27 columns

In [37]:

```
pmes=[]
for n in range (len(wmain)):
    y=wmain.groupby('key')
    f=wmain.loc[y.groups[wmain['key'][n]]].mean()
    f=f.mean()
    pmes.append(f)

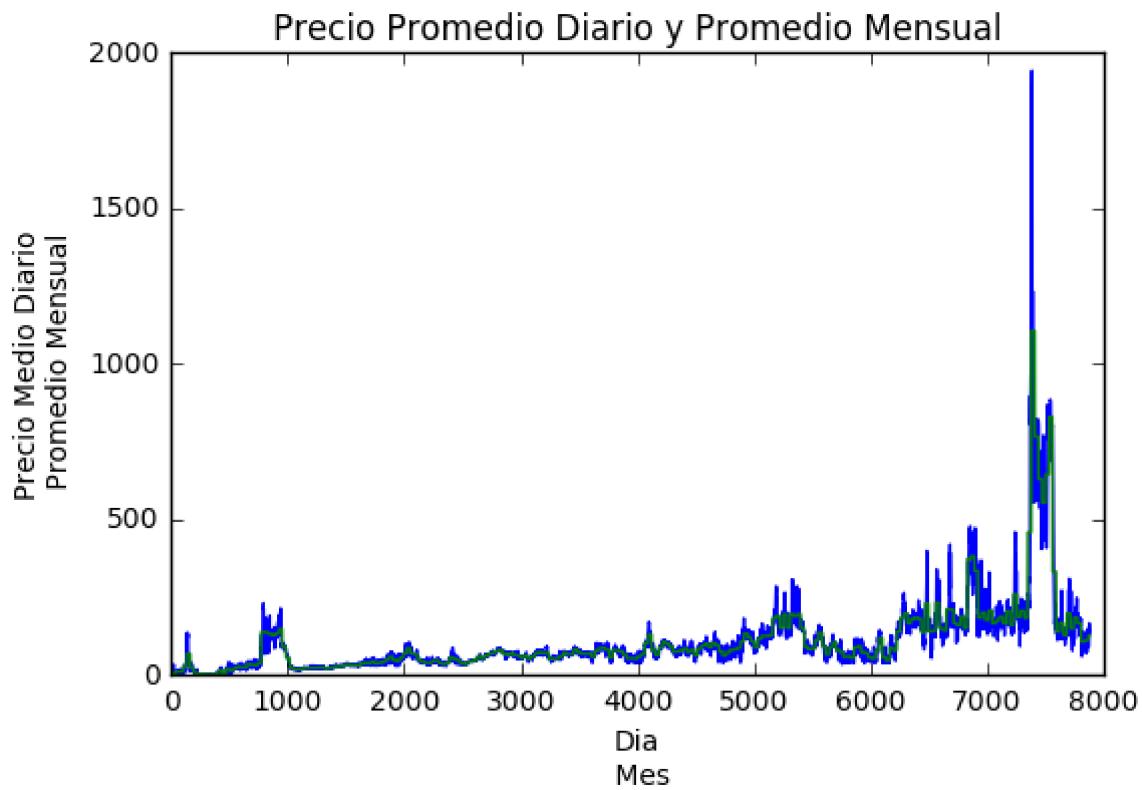
pmes
# Se agrupa por mes y anio y se repite el precio promedio mensual para todos Los
# dimensiones del precio promedio diario
```

Out[37]:

```
[1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214,
 1.55208712121214]
```

```
In [38]: plt.title("Precio Promedio Diario y Promedio Mensual")
plt.xlabel("Dia\n Mes")
plt.ylabel("Precio Medio Diario\n Promedio Mensual")
plt.plot(pd.Series(wmedia).values)
plt.plot(pd.Series(pmese).values)
# Se grafica el precio promedio diario y precio promedio mensual en las mismas di
```

Out[38]: [`<matplotlib.lines.Line2D at 0x2581ec8ed68>`]



In [ ]: