

Nombre: *Macías Pico Josselyn Stefany* ¶

Curso: *Sexto "B"*

Materia: *Metodos Numericos*

Docente: *Ing. Jorge Anibal Moya Delgado*

SISTEMA ESTADISTICO DE CRIPTOMONEDAS "BITCOIN"

La presente investigación se refiere al tema de las criptomonedas ya que hoy en día se vive en una sociedad basada en los negocios virtuales de manera que algunas de las personas han entrado a los negocios de criptomonedas ya que es un medio digital de intercambio, lo han utilizado para poder obtener mayores ingresos y darles mayor usabilidad a variedades de monedas, entre ellas están el bitcoin que es de la que se hablara en este informe.

Ejercicio

Este ejercicio se realiza despues de haber mencionado algunos obstaculos importantes sobre las criptomonedas "BITCOIN" dado que no hay un estado detrás de ellas lo cual pueda gestionar su precio siendo en este caso lo que hacen los bancos centrales al gestionar una inflación o garantizar el cumplimiento de los pagos. Es por esto que se ha decidido utilizar un software en el cual se pueda mostrar la variación con cada uno de los datos cuando sube o baja la moneda, para así poder llevar un control entre la informacion de los máximos y mínimos de la moneda incluyendo el valor de apertura y el valor que define al terminar el día.

Obtencion de datos

En esta parte se mostraran los datos dentro de la pagina como respuesta

```
In [1]: 1 import pandas as pd
        2 import matplotlib.pyplot as plt
        3 import numpy as np
        4
        5 datos = pd.read_excel('C:/Users/Villamar/Desktop/JOSS/SEXTOSEMESTRE/MODELAMIA
        6 datos
```

Out[1]:

	fecha	total-bitcoins	volumen-comercio	promedio-bloque	billetera-usuarios	pagos-moneda	precio-mercado
0	2020-07-03	18421362.50	1.072075e+08	3453.789809	50807793	542245	9087.98
1	2020-07-04	18421587.50	5.780329e+07	2975.134969	50807793	484947	9072.42
2	2020-07-05	18421806.25	3.883966e+07	4027.668831	50819572	620261	9131.31
3	2020-07-06	18422025.00	5.072085e+07	4034.937888	50834080	649625	9089.09
4	2020-07-07	18422243.75	1.137434e+08	4257.635762	50844380	642903	9348.91
...
361	2021-06-29	18500556.25	3.559005e+08	6129.449438	54290217	545521	34456.67
362	2021-06-30	18500781.25	3.851351e+08	6221.294737	54309711	591023	35847.70
363	2021-07-01	18501006.25	4.069678e+08	5856.044444	54309711	527044	35047.36
364	2021-07-02	18501225.00	3.241165e+08	4955.709091	54323892	671920	33536.88
365	2021-07-03	18501443.75	2.170665e+08	5407.932584	54328109	622738	33856.86

366 rows × 7 columns

Histogramas

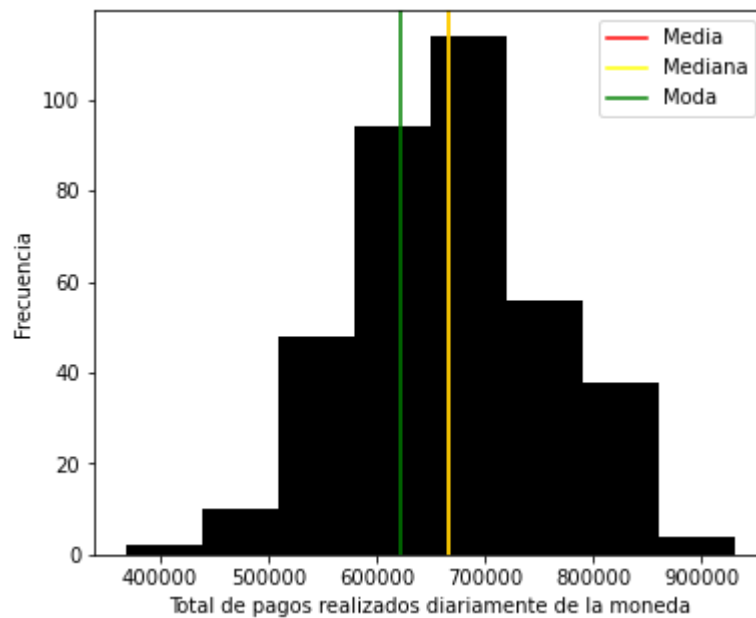
Histograma que mostrara el pago de la moneda diariamente

In [51]:

```

1 #Gráficos
2 x=datos["pagos-moneda"]
3 #modificacion del grafico anchura,altura
4 plt.figure(figsize=(6,5))
5 plt.hist(x,bins=8,color='black')
6 plt.axvline(x.mean(),color='red',label='Media')
7 plt.axvline(x.median(),color='yellow',label='Mediana')
8 plt.axvline(x.mode()[0],color='green',label='Moda')
9 #titulo de las ejes de la X
10 plt.xlabel('Total de pagos realizados diariamente de la moneda')
11 #titulo de las ejes de la y
12 plt.ylabel('Frecuencia')
13 plt.legend()
14 plt.show()

```

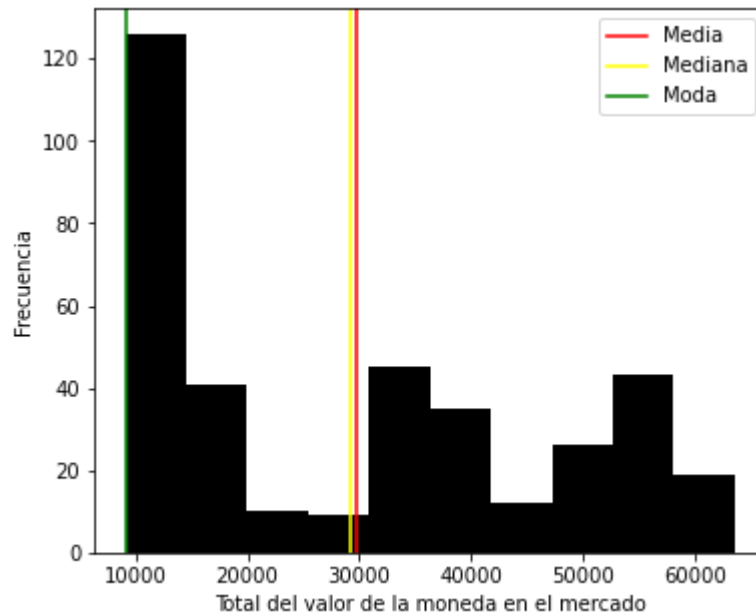


Histograma que mostrara el precio de la moneda en el mercado

```

In [52]: 1 x=datos["precio-mercado"]
2 #modificacion del grafico anchura,altura
3 plt.figure(figsize=(6,5))
4 plt.hist(x,bins=None,color='black')
5 plt.axvline(x.mean(),color='red',label='Media')
6 plt.axvline(x.median(),color='yellow',label='Mediana')
7 plt.axvline(x.mode()[0],color='green',label='Moda')
8 #titulo de las ejes de la X
9 plt.xlabel('Total del valor de la moneda en el mercado')
10 #titulo de las ejes de la y
11 plt.ylabel('Frecuencia')
12 plt.legend()
13 plt.show()

```

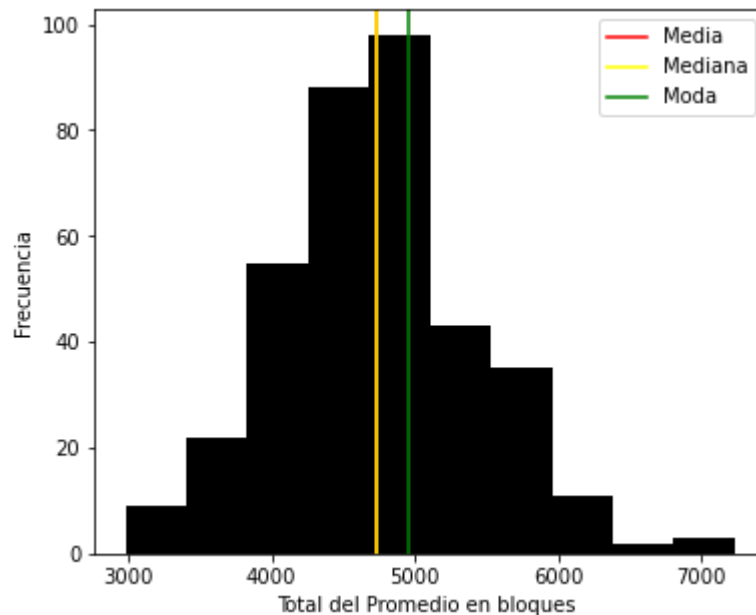


Histograma que mostrara el promedio de la moneda por bloques

```

In [53]: 1 x=datos["promedio-bloque"]
2 #modificacion del grafico anchura,altura
3 plt.figure(figsize=(6,5))
4 plt.hist(x,bins=None,color='black')
5 plt.axvline(x.mean(),color='red',label='Media')
6 plt.axvline(x.median(),color='yellow',label='Mediana')
7 plt.axvline(x.mode()[0],color='green',label='Moda')
8 #titulo de las ejes de la X
9 plt.xlabel('Total del Promedio en bloques ')
10 #titulo de las ejes de la y
11 plt.ylabel('Frecuencia')
12 plt.legend()
13 plt.show()

```



Media

Es el valor que tendrían los datos, si todos ellos fueran iguales.

En esta parte se calcula la media de las columnas:

- pago-moneda
- preco-mercado
- promedio-bloque

Cada una cuenta con una variable la cual sera mostrada seguido de un mensaje

```
In [54]: 1 #Calculo de la media
2 print("Media:")
3 #Entrada de los datos por medios del array
4
5 t4 =datos["pagos-monedas"].mean()
6 t7 =datos["precio-mercado"].mean()
7 t8 =datos["promedio-bloque"].mean()
8
9 #Mostreo de los datos
10 print( "\nla Media de los pagos en moneda realizados diariamente: ", t4)
11 print( "\nla Media del precio de la moneda en el mercado: ", t7)
12 print( "\nla Media del promedio de la moneda en bloques: ", t8)
13
```

Media:

la Media de los pagos en moneda realizados diariamente: 667037.9153005464

la Media del precio de la moneda en el mercado: 29757.23409836064

la Media del promedio de la moneda en bloques: 4737.817554775539

Media Aritmetica

Es el valor que ocupa la posición central. Si el número de datos es par, la mediana es la media aritmética de los dos centrales.

En esta parte se calcula la media aritmetica de las columnas:

- pago-monedas
- preco-mercado
- promedio-bloque

Cada una cuenta con una variable la cual sera mostrada seguido de un mensaje

```
In [58]: 1 #Calculo de la Mediana
2 print("Mediana:")
3 #Entrada de los datos por medios del array
4 m1 =datos["pagos-moneda"].median()
5 m2 =datos["precio-mercado"].median()
6 m3 =datos["promedio-bloque"].median()
7 #Mostreo de los datos
8 print( "\nla Mediana de los pagos en moneda realizados diariamente: ", m1)
9 print( "\nla Mediana del precio de la moneda en el mercado: ", m2)
10 print( "\nla Mediana del promedio de la moneda en bloques: ", m3)
```

Mediana:

la Mediana de los pagos en moneda realizados diariamente:: 666523.0

la Mediana del precio de la moneda en el mercado: 29188.155

la Mediana del promedio de la moneda en bloques: 4724.71015495296

Moda

Es el valor que más se repite o, lo que es lo mismo, el que tiene la mayor frecuencia.

En esta parte se calcula la moda de las columnas:

- pago-moneda
- preco-mercado
- promedio-bloque

Cada una cuenta con una variable la cual sera mostrada seguido de un mensaje, y los datos seran guardados a la libreria de panda

```
In [62]: 1 #Calculo de la Moda
2 print("Moda:")
3 #Entrada de los datos por medios del array
4 mo1 = datos["billetera-usuarios"].mode()
5 mo2 = datos["pagos-moneda"].mode()
6 mo3 = datos["pagos-moneda"].mode()
7 #Mostreo de los datos
8 print( "\nla Moda de los pagos en moneda realizados diariamente: \n\n", mo
9 print( "\nla Moda del precio de la moneda en el mercado: \n\n", mo2)
10 print( "\nla Moda del promedio de la moneda en bloques: \n\n", mo3)
11 #agregandolas a la libreria panda
12 pd.DataFrame(mo1)
13 pd.DataFrame(mo2)
14 pd.DataFrame(mo3)
15
16
17
```

Moda:

la Moda de los pagos en moneda realizados diariamente:

```
0      50807793
1      50874784
2      50933404
3      50999025
4      51072983
5      51143211
6      51213490
7      51289691
8      51363967
9      51438971
10     51504055
11     51585531
12     51667919
13     51762365
14     51838178
15     51933930
16     52016198
17     52083351
18     52148832
19     52179995
20     52217981
21     52281159
22     52349422
23     52429897
24     52507724
25     52580626
26     52602380
27     52649995
28     52716021
29     52782166
30     52845759
31     52905086
32     52969897
33     53029649
34     53087838
```



```

35    53134626
36    53193624
37    53256853
38    53320829
39    53375387
40    53425635
41    53486559
42    53539575
43    53602438
44    53662339
45    53730063
46    53830258
47    53897442
48    53969300
49    54030335
50    54089822
51    54156314
52    54224524
53    54309711
dtype: int64

```

la Moda del precio de la moneda en el mercado:

```

0    622738
1    671920
2    727861
dtype: int64

```

la Moda del promedio de la moneda en bloques:

```

0    622738
1    671920
2    727861
dtype: int64

```

```

Out[62]:
0
-----
0  622738
1  671920
2  727861

```

Medidas de tendencias

A continuacion se mostrara las medidas de tendencia central entre esos, valores iniciales, valor de salvamiento y periodos de recuperacion al utilizar la palabra describe se mostrara incluso los cuantiles.

```
In [80]: 1 #Medidas de tendencia central
2
3 datos[['promedio-bloque', 'billetera-usuarios', 'precio-mercado']]. describe()
```

```
Out[80]:
```

	promedio-bloque	billetera-usuarios	precio-mercado
count	366.000000	3.660000e+02	366.000000
mean	4737.817555	5.261965e+07	29757.234098
std	708.186550	1.015115e+06	17802.870974
min	2975.134969	5.080779e+07	9072.420000
25%	4276.447251	5.176236e+07	11683.595000
50%	4724.710155	5.269910e+07	29188.155000
75%	5143.439882	5.347240e+07	46552.595000
max	7236.203883	5.432811e+07	63554.440000

Despues, se ingresaran los datos estadisticos dentro de un array con el valor inicial, valor de salvamento y periodo de recuperacion

```
In [81]: 1 #Ingreso de Los datos al Array con su expacion
2
3 df_1 = datos[:7]
4 print ("Estadisticos de acuerdo a las fechas")
5 print ("-----")
6 df_1[['promedio-bloque', 'billetera-usuarios', 'pagos-monedas']]. describe()
```

Estadisticos de acuerdo a las fechas

```
Out[81]:
```

	promedio-bloque	billetera-usuarios	pagos-monedas
count	7.000000	7.000000e+00	7.000000
mean	3823.440563	5.083219e+07	606070.000000
std	447.318300	2.126225e+04	66468.344398
min	2975.134969	5.080779e+07	484947.000000
25%	3716.450805	5.081368e+07	581253.000000
50%	4027.668831	5.083408e+07	640637.000000
75%	4035.371383	5.084675e+07	646264.000000
max	4257.635762	5.086258e+07	661872.000000

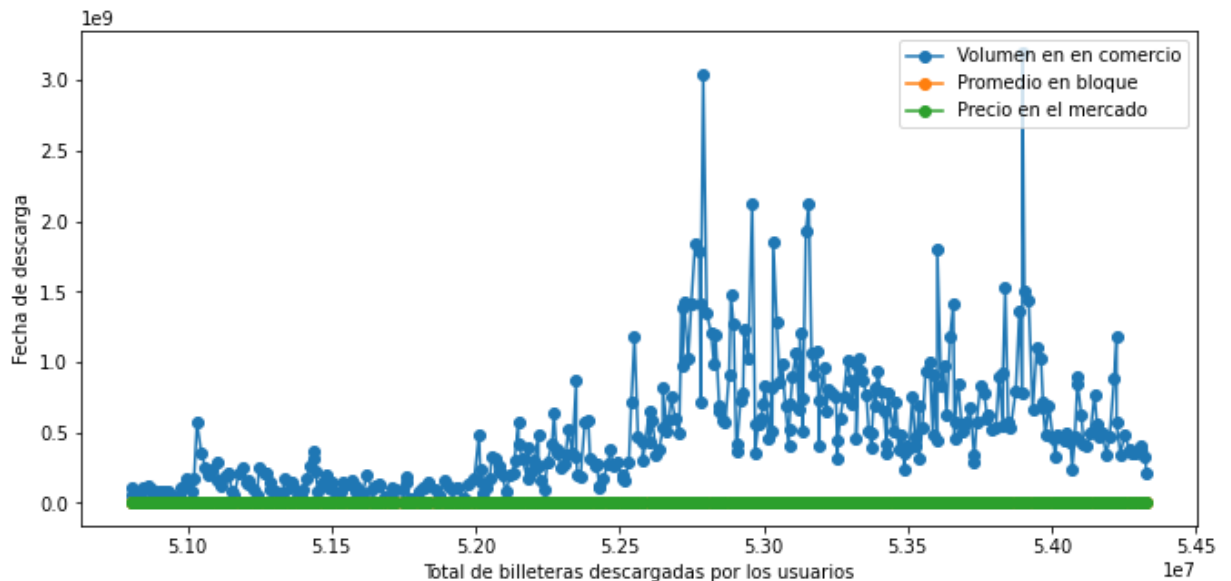
Graficos adicionales

Se obtendran datos de la fuente principal y se guardaran en una variable:

- Se ingresa principalmente los datos en las ejes de la x el cual en este caso es la billetera que descargan los usuarios
- Se agregan variables las cuales guardan los datos de las columnas respectivas
- Se realizan las modificaciones concorde a lo que se desee mostrar en el grafico
- y por ultimo se impren los datos para que se muestre el grafico

```
In [70]: 1 #fecha total-bitcoins volumen-comercio promedio-bloque billetera-usuari
2
3 #Otros Graficos Adicionales
4 #Ingreso de las categoria en las ejes de la X
5 x = datos["billetera-usuarios"]
6 #Ingreso de los datos al Array
7 t1 = datos["volumen-comercio"]
8 t2 = datos["promedio-bloque"]
9 t3 = datos["precio-mercado"]
10 #Modificacion del grafico
11 plt.figure(figsize=(11,5))
12 plt.plot(x,t1,x,t2,x,t3,marker='o')
13 #Imprecion de los datos y modificacion del grafico
14 plt.xlabel('Total de billeteras descargadas por los usuarios')
15 plt.ylabel('Fecha de descarga')
16 plt.legend(('Volumen en en comercio', 'Promedio en bloque', 'Precio en el mer
```

Out[70]: <matplotlib.legend.Legend at 0x237e2fb6280>



Grafica para cada columna utilizando solo una variable para las y, esta sera utilizada para cada una de las otras columnas:

- En esta parte se ubica el rango de los datos con los cuales se esta trabajando
- Luego se configuran las dimensiones en el grafico
- Se ingresa una variable la cual guarde los datos de la columna a utilizar
- Luego se detalla el primer grafico donde se ubica el titulo del grafico y el varlos de la y
- Despues se detalla el segundo grafico donde se agrega la columna con la cual se deba trabajar
- Por ultimo la otra columna y luego se ejecuta para poder mostrar los datos respectivos.

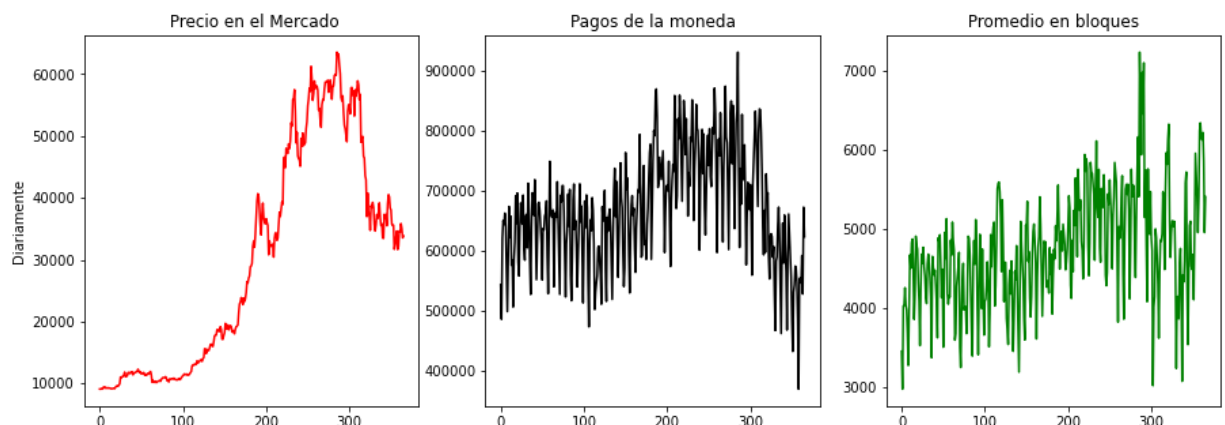
In [71]:

```

1
2
3 #Entra del rango al Array con Los datos que estan en la fuente de datos
4 x = range(366)
5 #Configuracion de Las dimensiones que el grafico optendra
6 plt.figure(figsize=(15,5))
7 plt.subplot(131)
8 #Entrada de Los datos a Las array
9 t1 = datos["precio-mercado"]
10 #'r'=> (rojo) es el color que se le puede agregar es una simbolo
11 p1 = plt.plot(x,t1, 'red')
12 plt.ylabel('Diariamente')
13 plt.title(' Precio en el Mercado')
14 plt.subplot(132)
15 t2 = datos["pagos-moneda"]
16 #'k'=> (negro)
17 p1 = plt.plot(x,t2,'black')
18 plt.title('Pagos de la moneda')
19 plt.subplot(133)
20 t3 = datos["promedio-bloque"]
21 #'g'=> (verde)
22 p1 = plt.plot(x,t3,'green')
23 plt.title(' Promedio en bloques ')
24

```

Out[71]: Text(0.5, 1.0, ' Promedio en bloques ')



Datos unicos de la tabla

A continuacion, se mostraran los datos unicos del la cantidad de usuarios que tienen descargadas sus billeteras lo cual se guardara en una variable llamada dfclases

In [72]:

```

1
2 #fecha total-bitcoins volumen-comercio promedio-bloque billetera-usuari
3
4 # OBTENER LOS DATOS UNICOS DE LA TABLA
5 #Agregar la columna de valor inicial a una variable
6 lis = datos["billetera-usuarios"].unique()
7 #Para poder sacar los valores unicos de la tabla
8 dfclases=pd.DataFrame(lis,columns=["billetera-usuarios"])
9 dfclases

```

Out[72]:

	billetera-usuarios
0	50807793
1	50819572
2	50834080
3	50844380
4	50849128
...	...
307	54274719
308	54290217
309	54309711
310	54323892
311	54328109

312 rows × 1 columns

Frecuencias absolutas

La frecuencia absoluta es una medida estadística que nos da información acerca de la cantidad de veces que se repite un suceso al realizar un número determinado de experimentos aleatorios.

Para obtener estos datos se debe crear una lista con los valores de las frecuencias y se agrega a la columna dataframe a la cual se le ha puesto Fi, y se muestran los datos unicos guardados anteriormente

```
In [82]: 1 #TABLA DE FRECUENCIAS ABSOLUTAS
2 # OBTENER FRECUENCIAS ABSOLUTAS DE CADA CLASE
3 datafi=pd.crosstab(index=datos["billetera-usuarios"], columns = "fi")
4 # Creamos una lista con los valores de las frecuencias
5 li = datafi.values
6 # agregamos una columna al dataframe
7 dfclases["fi"] = li
8 #observamos dfclase
9 dfclases
```

```
Out[82]:
```

	billetera-usuarios	fi	hi	FA	HI
0	50807793	2	0.005464	2	0.005464
1	50819572	1	0.002732	3	0.008197
2	50834080	1	0.002732	4	0.010929
3	50844380	1	0.002732	5	0.013661
4	50849128	1	0.002732	6	0.016393
...
307	54274719	1	0.002732	361	0.986339
308	54290217	1	0.002732	362	0.989071
309	54309711	2	0.005464	364	0.994536
310	54323892	1	0.002732	365	0.997268
311	54328109	1	0.002732	366	1.000000

312 rows × 5 columns

Total de los datos anteriores

```
In [74]: 1 total = dfclases.sum(axis=0)
2 total
3
```

```
Out[74]: billetera-usuarios    16418741387
fi                               366
dtype: int64
```

Frecuencia Relativa

La frecuencia relativa es una medida estadística que se calcula como el cociente de la frecuencia absoluta de algún valor de la población/muestra (fi) entre el total de valores que componen la población/muestra (N).

Para obtener los datos de la frecuencia relativa se calcula y se agrega la columna adicional con el resultado de los datos utilizando algunos datos de la Frecuencia Absoluta

```
In [75]: 1 # Columna de Frecuencia relativa
2 total = dfclases.sum(axis=0)
3 datahi = dfclases["fi"]/total["fi"] # aqui calculamos la frecuencia
4 datahi.values
5 # agregamos nueva columna de frecuencia relativa
6 dfclases["hi"] = datahi
7 dfclases
```

```
Out[75]:
```

	billetera-usuarios	fi	hi
0	50807793	2	0.005464
1	50819572	1	0.002732
2	50834080	1	0.002732
3	50844380	1	0.002732
4	50849128	1	0.002732
...
307	54274719	1	0.002732
308	54290217	1	0.002732
309	54309711	2	0.005464
310	54323892	1	0.002732
311	54328109	1	0.002732

312 rows × 3 columns

Mostramos el total de los datos utilizando las siguientes lineas de codigo

```
In [76]: 1 total1 = dfclases.sum(axis=0)
2 total1
```

```
Out[76]: billetera-usuarios    1.641874e+10
fi                               3.660000e+02
hi                               1.000000e+00
dtype: float64
```

Suma de frecuencias relativas

Realizamos una suma de frecuencias relativas utilizando algunos de los datos anteriores los guardamos en la variable utilizada y se mostraran cada uno de los datos mediante un for para que se puedan sumar los datos y asi mostrarlos de una mejor manera

```

In [77]: 1 # La suma de las frecuencias Relativas nos da 1
          2 # aquí vamos a calcular la frecuencia absoluta
          3 FA = dfclases["fi"].values
          4 # obtenemos FA
          5 a=[]
          6 b=0
          7 for c in FA:
          8     b = c + b
          9     a.append(b)
         10 dfclases["FA"] = a
         11 HI = dfclases["hi"].values
         12 # obtenemos HI
         13 a=[]
         14 b=0
         15 for c in HI:
         16     b = c + b
         17     a.append(b)
         18 dfclases["HI"] = a
         19 dfclases

```

```

Out[77]:

```

	billettera-usuarios	fi	hi	FA	HI
0	50807793	2	0.005464	2	0.005464
1	50819572	1	0.002732	3	0.008197
2	50834080	1	0.002732	4	0.010929
3	50844380	1	0.002732	5	0.013661
4	50849128	1	0.002732	6	0.016393
...
307	54274719	1	0.002732	361	0.986339
308	54290217	1	0.002732	362	0.989071
309	54309711	2	0.005464	364	0.994536
310	54323892	1	0.002732	365	0.997268
311	54328109	1	0.002732	366	1.000000

312 rows × 5 columns

Deteminamos el valor total 2 y lo mostramos


```
In [78]: 1 total2 = dfclases.sum(axis=0)
        2 total2
```

```
Out[78]: billetera-usuarios    1.641874e+10
         fi                    3.660000e+02
         hi                    1.000000e+00
         FA                    5.744000e+04
         HI                    1.569399e+02
         dtype: float64
```

Mostramos toda la informacion con la siguiente linea de codigo

```
In [79]: 1 datos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 366 entries, 0 to 365
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   fecha                 366 non-null   datetime64[ns]
1   total-bitcoins        366 non-null   float64
2   volumen-comercio      366 non-null   float64
3   promedio-bloque       366 non-null   float64
4   billetera-usuarios    366 non-null   int64
5   pagos-moneda          366 non-null   int64
6   precio-mercado        366 non-null   float64
dtypes: datetime64[ns](1), float64(4), int64(2)
memory usage: 20.1 KB
```