Academy

CURSO

Python aplicado a finanzas

MÓDULO 4

Indicador MACD

Diseñado para revelar cambios en la fuerza, la dirección, el impulso y la duración de una tendencia en el precio.

MÓDULO 4

Indicador MACD

- Fórmula: MACD=PME(12)-PME(26), donde PME es Promedio Móvil Exponencial.
- El segundo componente es la Señal o Signal, la cual corresponde al promedio móvil exponencial del MACD calculado anteriormente y se utiliza como señal para iniciar o cerrar una posición. El parámetro más común es 9 periodos. Su fórmula es: Señal=PME(9,MACD)
- El tercer componente es el Histograma, el cual corresponde a la diferencia entre el MACD y la Señal y sirve como indicador para iniciar o cerrar una posición. Su fórmula es: Histograma= MACD-Señal

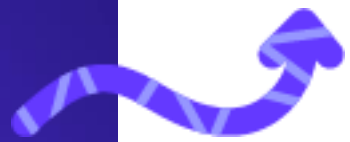
Indicador **MACD**

```
[ ]  import yfinance as yf

     df = yf.download("ggal", auto_adjust=True)

     '''
     El MACD toma la columna "Close" para hacer los calculos.
     Enviar datos ajustados
     '''
     df = df["2021-05":].copy()

     fast = 9
     slow = 26
     suavizado = 9

     df['ema_fast'] = df.Close.ewm(span=fast).mean()
     df['ema_slow'] = df["Close"].ewm(span=slow).mean()
     df['macd'] = df.ema_fast - df.ema_slow
     df['signal'] = df.macd.rolling(suavizado).mean()
     df['histograma'] = df.macd - df.signal
     df = df.dropna().round(2)
     df
```

```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Volume	ema_fast	ema_slow	macd	signal	histograma
2021-05-13	7.65	7.92	7.63	7.90	726700	7.68	7.61	0.07	0.04	0.04
2021-05-14	8.07	8.53	8.07	8.33	1340500	7.83	7.71	0.12	0.05	0.07
2021-05-17	8.36	8.58	8.04	8.56	1081100	7.99	7.82	0.17	0.07	0.10
2021-05-18	8.68	8.70	8.35	8.42	873200	8.08	7.89	0.19	0.09	0.10
2021-05-19	8.30	8.42	8.22	8.34	449500	8.14	7.95	0.19	0.11	0.08
...	...	...	...	...	...	...	...	...	...	...
2021-11-18	9.90	10.33	9.69	10.19	784400	10.93	11.04	-0.11	0.30	-0.40
2021-11-19	10.12	10.28	9.55	9.58	1000700	10.66	10.93	-0.27	0.23	-0.49
2021-11-22	9.72	9.72	9.12	9.15	827400	10.36	10.80	-0.44	0.13	-0.57
2021-11-23	9.25	9.54	9.11	9.26	1477100	10.14	10.69	-0.55	0.01	-0.56
2021-11-24	9.30	9.41	9.01	9.14	721534	9.94	10.57	-0.63	-0.11	-0.52

137 rows × 10 columns



Medias Móviles **Simples**

```
[ ]  df = yf.download("ggal", auto_adjust=True)
```

```
[**********100%**********]  1 of 1 completed
```

```
[ ]  for n in [3, 6, 9]:
       df[f"sma{n}"] = df.Close.rolling(n).mean()

     df
```

Date	Open	High	Low	Close	Volume	sma3	sma6	sma9
2000-07-25	16.019054	16.262417	15.346225	16.033369	126200	NaN	NaN	NaN
2000-07-26	15.804321	16.090631	15.747059	16.033369	28900	NaN	NaN	NaN
2000-07-27	16.033369	16.147893	15.918845	16.033369	61200	16.033369	NaN	NaN
2000-07-28	16.090632	16.090632	15.689798	15.918846	146100	15.995195	NaN	NaN
2000-07-31	16.033368	16.205154	15.460748	16.205154	178400	16.052457	NaN	NaN
...	...	...	...	...	...	...	...	...
2021-11-18	9.900000	10.330000	9.690000	10.190000	784400	10.250000	10.923333	11.294445
2021-11-19	10.120000	10.280000	9.550000	9.580000	1000700	9.943333	10.540000	11.037778
2021-11-22	9.720000	9.720000	9.120000	9.150000	827400	9.640000	10.145000	10.712222
2021-11-23	9.250000	9.540000	9.110000	9.260000	1477100	9.330000	9.790000	10.392222
2021-11-24	9.300000	9.415000	9.010000	9.140000	721534	9.183333	9.563333	10.087778

5370 rows × 8 columns

Bandas de **Bollinger**

- Introducido por John Bollinger en los años ‘80.
- Son dos curvas que envuelven al gráfico de precios.
- Se calcula a partir de una media móvil (simple o exponencial) sobre el precio de cierre a la que envuelven dos bandas que se obtienen de añadir y sustraer al valor de la media K desviaciones estándar (habitualmente, K = 2)
- La distancia entre las curvas superior e inferior, igual a cuatro desviaciones estándar, es por lo tanto una medida de la volatilidad del precio del activo.
- De acuerdo con el análisis técnico, el que los precios sobrepasen las bandas indica que el mercado está sobrecomprado (si lo hacen por arriba) o sobrevendido (si lo hacen por abajo)

Bandas de **Bollinger**

```
[ ]  df  = yf.download("AAPL", auto_adjust=True)

     df['sma_20'] = df["Close"].rolling(20).mean()
     volatilidad = df["Close"].rolling(20).std()
     df['boll_inf'] = df.sma_20 - 2 * volatilidad
     df['boll_sup'] = df.sma_20 + 2 * volatilidad
     df.dropna(inplace=True)
     df
```

```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Volume	sma_20	boll_inf	boll_sup
1981-01-12	0.111372	0.111372	0.110499	0.110499	23699200	0.108468	0.085547	0.131389
1981-01-13	0.107005	0.107005	0.106568	0.106568	23049600	0.108774	0.086141	0.131406
1981-01-14	0.107005	0.107442	0.107005	0.107005	14291200	0.109363	0.087622	0.131105
1981-01-15	0.109189	0.110062	0.109189	0.109189	14067200	0.110412	0.091072	0.129751
1981-01-16	0.108752	0.108752	0.108315	0.108315	13395200	0.111307	0.094357	0.128257
...	...	...	...	...	...	...	...	...
2021-11-18	153.710007	158.669998	153.050003	157.869995	137827700	150.389254	145.791921	154.986587
2021-11-19	157.649994	161.020004	156.529999	160.550003	117147500	150.993089	144.624231	157.361946
2021-11-22	161.679993	165.699997	161.000000	161.020004	117467900	151.622919	143.963415	159.282424
2021-11-23	161.119995	161.800003	159.059998	161.410004	96041900	152.238300	143.526140	160.950460
2021-11-24	160.750000	162.139999	159.639999	161.940002	66048978	152.903645	143.358131	162.449160

10308 rows × 8 columns

RSI (Relative Strength Index)

```
[ ]  import yfinance as yf
     import numpy as np

     data = yf.download('YPF', end='2020-04-03', auto_adjust=True)

     ruedas = 14

     data['dif'] = data['Adj Close'].diff()
     data['win'] = np.where(data['dif'] > 0, data['dif'], 0)
     data['loss'] = np.where(data['dif'] < 0, abs(data['dif']), 0)


     data['ema_win'] = data.win.ewm(alpha=1/ruedas).mean()
     data['ema_loss'] = data.loss.ewm(alpha=1/ruedas).mean()
     data['rs'] = data.ema_win / data.ema_loss
     data['rsi'] = 100 - (100 / (1+data.rs))
     data = data.reset_index().dropna().round(2)
     data
```

Gráfico con plot de Pandas

```
[ ]  df.Close.plot(figsize=(16, 10), legend=True )
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd1b88d3c50>
```

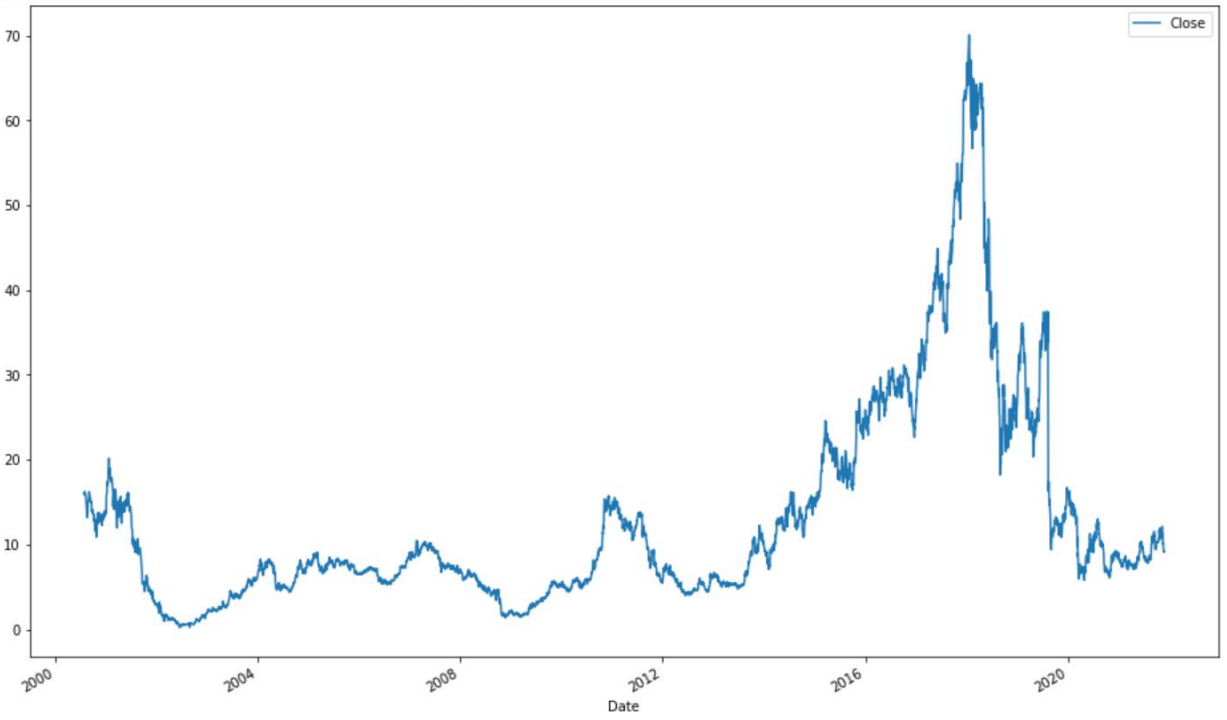


Gráfico de cruces de Medias

```
[ ]  df = yf.download("MMM", auto_adjust=True)

     for x in [3,9,18]:
       df[f"sma{x}"] = df.Close.rolling(x). mean()


     df = df["2021-05":].copy()
     df["Price"] = df.Close
     df.Price.plot(color="b", ls="--", legend=True)
     df.sma3.plot(color="green",legend=True)
     df.sma9.plot(color="r",legend=True)
     df.sma18.plot(color="y", title="Cruce 3,9 y 18", figsize= (16,10), grid= False, legend=True)
```

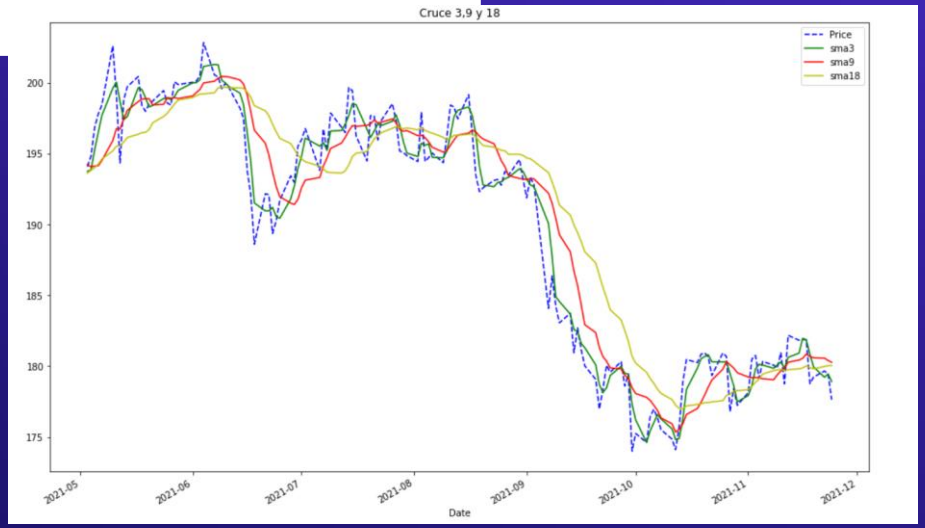
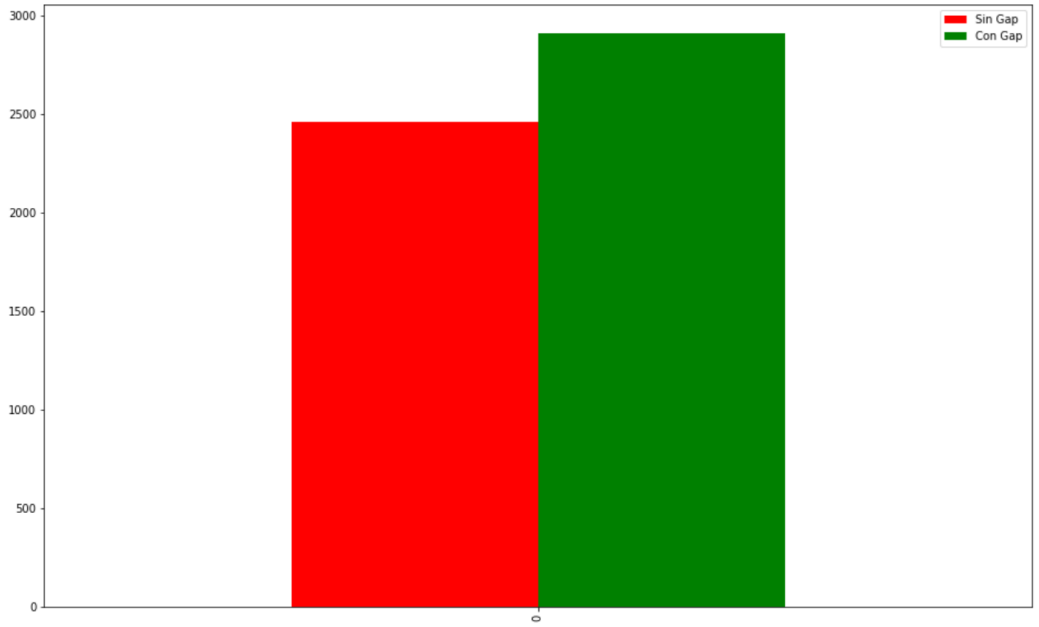


Gráfico de Barras

```
[ ]  ##Grafico de barras

     import yfinance as yf

     df = yf.download("ggal", auto_adjust=True)

     df["gap"] = (df.Open > df.Close.shift())
     frame = df.groupby("gap").size()
     print(frame)
     frame = frame.to_frame().transpose()
     print(frame)
     frame.columns = ["Sin Gap","Con Gap"]
```

```
frame.plot(kind="bar", figsize=(16,10), color=["r","g"])
```



Histograma

```
[ ]  ##Histograma
     ## https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.hist.html
     import yfinance as yf

     df = yf.download("ggal", start="2019-01-01", auto_adjust=True)
     df.sort_values("Close")
```

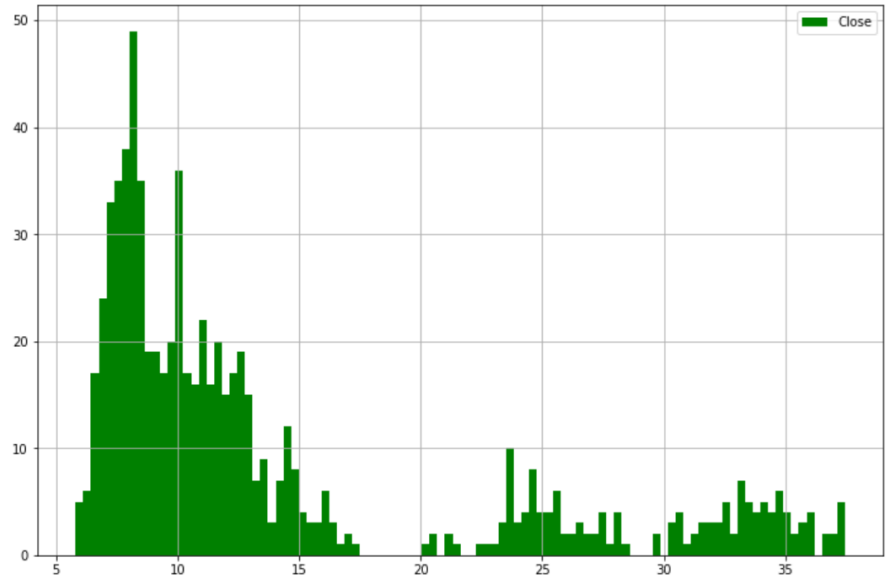
```
[**********100%**********]  1 of 1 completed
```

Date	Open	High	Low	Close	Volume
2020-04-27	6.090398	6.266931	5.550990	5.815790	2267500
2020-03-18	6.776916	6.963257	5.776561	5.962902	2319500
2020-03-19	5.786368	6.767109	5.737331	5.982516	1430500
2020-04-24	6.178665	6.345390	5.884442	5.982516	1468400
2020-10-28	6.140231	6.447739	5.812885	6.070794	1204000
...	...	...	...	...	...
2019-07-11	37.503517	38.042923	36.866034	37.199486	844300
2019-07-10	37.366208	38.464640	36.591426	37.268135	1033100
2019-07-09	36.336432	37.738891	36.208935	37.415249	801200
2019-08-09	34.227839	37.944842	33.845347	37.434856	1954900
2019-07-29	36.689499	38.003691	35.944138	37.454475	1311700

732 rows × 5 columns

```
[ ]  df.Close.hist( color="g", legend=True, bins = 100, figsize=(12,8))
```

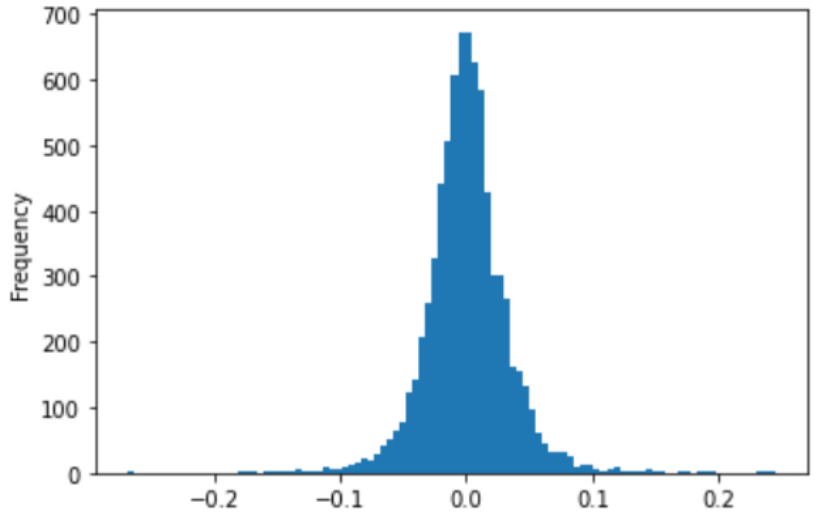
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd1b2dd0ed0>
```

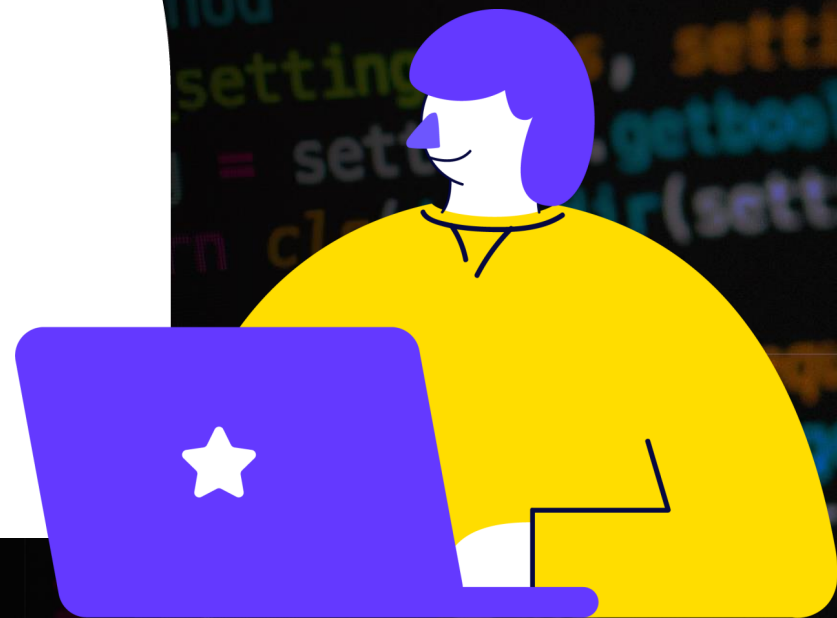


Histograma desde **Plot**

```
[ ]  df = yf.download('X')
     df['variacion'] = df['Adj Close'].pct_change()
     df.variacion.plot(kind='hist', bins=100)
```

```
[**********100%**********]  1 of 1 completed
<matplotlib.axes._subplots.AxesSubplot at 0x7fd1b294a050>
```






Academy