

COMERCIO CUANTITATIVO

CONTEXTO

- Mercado: EUR/USD
- Marco temporal: 1H
- Descarga de datos (3 años)
- Presenta sin fuga ("solo velas cerradas")
- Target (próxima vela)
- Validación temporal + walk-forward opcional
- Modelo de línea base + aumento de gradiente
- Backtest más realista con costos (spread + comisión + slippage)
- Métricas de trading (acciones, CAGR aprox, reducción máxima, Sharpe aprox)

Nota: Esto es para aprender el tubo. No es consejo financiero.

0) Instalación

```
En [2]: ! pip instalar yfinance pandas numpy scikit-learn matplotlib
```

Requisito ya satisfecho: yfinance en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (1.1.0)

Requisito ya satisfecho: pandas en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (3.0.0)

Requisito ya satisfecho: numpy en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (2.4.1)

Requisito ya satisfecho: scikit-learn en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (1.8.0)

Requisito ya satisfecho: matplotlib en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (3.10.8)

Requisito ya satisfecho: solicitudes >=2.31 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (2.32.5)

Requisito ya satisfecho: multitarea >=0.0.7 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (0.0.12)

Requisito ya satisfecho: platformdirs >=2.0.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (4.5.1)

Requisito ya satisfecho: pytz >=2022.5 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (2025.2)

Requisito ya satisfecho: frozendict >=2.3.4 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (2.4.7)

Requisito ya satisfecho: peewee >=3.16.2 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (3.19.0)

Requisito ya satisfecho: beautifulsoup4 >=4.11.1 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (4.14.3)

Requisito ya satisfecho: curl_cffi <0.14, >=0.7 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (0.13.0)

Requisito ya satisfecho: protobuf >=3.19.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (6.33.4)

Requisito ya satisfecho: websockets >=13.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de yfinance) (16.0)

Requisito ya satisfecho: python-dateutil >=2.8.2 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de pandas) (2.9.0.post0)

Requisito ya satisfecho: tzdata en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de pandas) (2025.3)

Requisito ya satisfecho: scipy >=1.10.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de scikit-learn) (1.17.0)

Requisito ya satisfecho: joblib >=1.3.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de scikit-learn) (1.5.3)

Requisito ya satisfecho: threadpoolctl >=3.2.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de scikit-learn) (3.6.0)

Requisito ya satisfecho: contourpy >=1.0.1 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (1.3.3)

Requisito ya satisfecho: cyceler >=0.10 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (0.12.1)

Requisito ya satisfecho: fonttools >=4.22.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (4.61.1)

Requisito ya satisfecho: kiwisolver >=1.3.1 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (1.4.9)

Requisito ya satisfecho: empaquetado >=20.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (26.0)

Requisito ya satisfecho: pillow >=8 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (12.1.0)

Requisito ya satisfecho: pyparsing >=3 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de matplotlib) (3.3.2)

Requisito ya satisfecho: soupsieve >=1.6.1 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de beautifulsoup4 >=4.11.1->yfinance) (2.8.3)

Requisito ya satisfecho: typing-extensions >=4.0.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de beautifulsoup4 >=4.11.1->yfinance) (4.15.0)

Requisito ya satisfecho: cffi>=1.12.0 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de curl_cffi<0.14,>=0.7->yfinance) (2.0.0)
Requisito ya satisfecho: certifi>=2024.2.2 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de curl_cffi<0.14,>=0.7->yfinance) (2026.1.4)
Requisito ya satisfecho: seis>=1.5 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de python-dateutil>=2.8.2->pandas) (1.17.0)
Requisito ya satisfecho: charset_normalizer<4,>=2 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de requests>=2.31->yfinance) (3.4.4)
Requisito ya satisfecho: idna<4,>=2.5 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de requests>=2.31->yfinance) (3.11)
Requisito ya satisfecho: urllib3<3,>=1.21.1 en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de requests>=2.31->yfinance) (2.6.3)
Requisito ya satisfecho: pycparser en c:\users\trendingpc\appdata\local\programs\python\python311\lib\site-packages (de cffi>=1.12.0->curl_cffi<0.14,>=0.7->yfinance) (3.0)

[aviso] Una nueva versión de pip disponible: 22.3 -> 25.3

[aviso] Para actualizar, ejecute: C:\Users\TRENDINGPC\AppData\Local\Programs\Python\Python311\python.exe -m pip install --upgrade pip

En [4]: `import sys !{ sys.ejecutable } -m pip install yfinance`

Recolección de yfinance

Descargando yfinance-1.1.0-py2.py3-none-any.whl.metadata (6,1 kB)
Requisito ya satisfecho: pandas>=1.3.0 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de yfinance) (2.3.3)
Requisito ya satisfecho: numpy>=1.16.5 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de yfinance) (2.3.5)
Requisito ya satisfecho: solicitudes >=2.31 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de yfinance) (2.32.5)
Recopilación de multitarea >=0.0.7 (de yfinance)
Descargando multitasking-0.0.12.tar.gz (19 kB)
Instalación de dependencias de compilación: iniciada
Instalación de dependencias de compilación: finalizada con estado 'listo'
Obteniendo requisitos para construir la rueda: iniciado
Obtención de requisitos para construir la rueda: finalizado con estado 'listo'
Preparación de metadatos (pyproject.toml): iniciado
Preparando metadatos (pyproject.toml): finalizado con estado 'listo'
Requisito ya satisfecho: platformdirs>=2.0.0 en c:\users\trendingpc\appdata\roaming\python\python314\site-packages (de yfinance) (4.5.1)
Requisito ya satisfecho: pytz>=2022.5 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de yfinance) (2025.2)
Recopilación de frozendict>=2.3.4 (de yfinance)
Descargando frozendict-2.4.7-py3-none-any.whl.metadata (23 kB)
Recolección de peewee>=3.16.2 (de yfinance)
Descargando peewee-3.19.0-py3-none-any.whl.metadata (7,0 kB)
Requisito ya satisfecho: beautifulsoup4>=4.11.1 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de yfinance) (4.14.3)
Recopilación de curl_cffi<0.14,>=0.7 (de yfinance)
Descargando curl_cffi-0.13.0-cp39-abi3-win_amd64.whl.metadata (13 kB)
Recopilando protobuf>=3.19.0 (de yfinance)
Descargando protobuf-6.33.4-cp310-abi3-win_amd64.whl.metadata (593 bytes)
Recopilación de websockets >=13.0 (de yfinance)
Descargando websockets-16.0-cp314-cp314-win_amd64.whl.metadata (7,0 kB)
Requisito ya satisfecho: cffi>=1.12.0 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de curl_cffi<0.14,>=0.7->yfinance) (2.0.0)
Requisito ya satisfecho: certifi>=2024.2.2 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de curl_cffi<0.14,>=0.7->yfinance) (2025.11.12)
Requisito ya satisfecho: soupsieve>=1.6.1 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de beautifulsoup4>=4.11.1->yfinance) (2.8)
Requisito ya satisfecho: typing-extensions>=4.0.0 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de beautifulsoup4>=4.11.1->yfinance) (4.15.0)
Requisito ya satisfecho: pycparser en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de cffi>=1.12.0->curl_cffi<0.14,>=0.7->yfinance) (2.23)
Requisito ya satisfecho: python-dateutil>=2.8.2 en c:\users\trendingpc\appdata\roaming\python\python314\site-packages (de pandas>=1.3.0->yfinance) (2.9.0.post0)
Requisito ya satisfecho: tzdata>=2022.7 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de pandas>=1.3.0->yfinance) (2025.2)
Requisito ya satisfecho: seis>=1.5 en c:\users\trendingpc\appdata\roaming\python\python314\site-packages (de python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
Requisito ya satisfecho: charset_normalizer<4,>=2 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de requests>=2.31->yfinance) (3.4.4)
Requisito ya satisfecho: idna<4,>=2.5 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de requests>=2.31->yfinance) (3.11)
Requisito ya satisfecho: urllib3<3,>=1.21.1 en c:\users\trendingpc\appdata\local\programs\python\python314\lib\site-packages (de requests>=2.31->yfinance) (2.6.

```

2)
Descargando yfinance-1.1.0-py2.py3-none-any.whl (129 kB)
Descargando curl_cffi-0.13.0-cp39-abi3-win_amd64.whl (1,6 MB)
----- 0,0/1,6 MB ? eta---:--
----- 0,5/1,6 MB 3,3 MB/s eta 0:00:01
----- 1.3/1.6 MB 3.3 MB/s eta 0:00:01
----- 1,6/1,6 MB 3,3 MB/s 0:00:00
Descargando frozendict-2.4.7-py3-none-any.whl (16 kB)
Descargando peewee-3.19.0-py3-none-any.whl (411 kB)
Descargando protobuf-6.33.4-cp310-abi3-win_amd64.whl (436 kB)
Descargando websockets-16.0-cp314-cp314-win_amd64.whl (178 kB)
Construyendo ruedas para paquetes recolectados: multitarea
  Construcción de una rueda para multitarea (pyproject.toml): iniciado
  Rueda de construcción para multitarea (pyproject.toml): finalizada con estado
'listo'
  Rueda creada para multitarea: nombre de archivo=multitasking-0.0.12-py3-none-an
y.whl tamaño=15690 sha256=f73bbb48914212708c1e3c83f1fc86c93a088e37d7c1b69a20017b4
eff5d53b7
  Almacenado en el directorio: c:\users\trendingpc\appdata\local\pip\cache\wheels
\7e\62\f9\20d7dbb144b6f563edab8e3a7fda71d976870cd41972035cdd
Multitarea construida con éxito
Instalación de paquetes recopilados: peewee, multitasking, websockets, protobuf,
frozendict, curl_cffi, yfinance

```

```

----- 0/7 [peewee]
----- 0/7 [peewee]
----- 2/7 [sockets web]
----- 2/7 [sockets web]
----- 2/7 [sockets web]
----- 2/7 [sockets web]
----- 3/7 [protobuf]
----- 3/7 [protobuf]
----- 3/7 [protobuf]
----- 4/7 [frozendict]
----- 5/7 [curl_cffi]
----- 6/7 [yfinance]
----- 6/7 [yfinance]
----- 7/7 [yfinance]

```

Se instalaron correctamente curl_cffi-0.13.0, frozendict-2.4.7, multitasking-0.0.12, peewee-3.19.0, protobuf-6.33.4, websockets-16.0 yfinance-1.1.0

En [5]: `importar yfinance como yf`

1) Descargar datos EURUSD 1H (3 años)

```

En [7]: importar yfinance como yf importar pandas como pd

símbolo = "EURUSD=X"
df = yf . descarga ( símbolo , periodo = "730d" , intervalo = "1h" , auto_a

# yfinance a veces devuelve columnas MultiIndex cuando hay tickers
df . column = [ c [ 0 ] if isinstance ( c , tuple ) else c for c in d
df = df . rename ( column = str . lower ) . dropna ( ) . copy ( )

df . cabeza ( ) , df . cola ( ) , df . forma

```

```
Salida[... ( adj cerrar cerrar alto bajo abrir \
Fecha y hora
13/04/2023 23:00:00+00:00 1.105461 1.105461 1.105461 1.105094 1.105094
14/04/2023 00:00:00+00:00 1.105705 1.105705 1.106317 1.105217 1.105217
14/04/2023 01:00:00+00:00 1.106684 1.106684 1.106684 1.105705 1.105828
14/04/2023 02:00:00+00:00 1.107665 1.107665 1.107788 1.106684 1.106684
14/04/2023 03:00:00+00:00 1.107297 1.107297 1.107665 1.106929 1.107297

volumen

Fecha y hora
13/04/2023 23:00:00+00:00 0
14/04/2023 00:00:00+00:00 0
14/04/2023 01:00:00+00:00 0
14/04/2023 02:00:00+00:00 0
2023-04-14 03:00:00+00:00 0 ,
adj cerrar cerrar alto bajo abrir \
Fecha y hora
2026-01-29 14:00:00+00:00 1.197175 1.197175 1.197318 1.195600 1.196172
2026-01-29 15:00:00+00:00 1.193317 1.193317 1.197031 1.191185 1.197031
2026-01-29 16:00:00+00:00 1.194600 1.194600 1.195457 1.193175 1.193460
2026-01-29 17:00:00+00:00 1.195886 1.195886 1.196458 1.194743 1.194886
2026-01-29 18:00:00+00:00 1.195314 1.195314 1.196315 1.195314 1.195743

volumen

Fecha y hora
2026-01-29 14:00:00+00:00 0
2026-01-29 15:00:00+00:00 0
2026-01-29 16:00:00+00:00 0
2026-01-29 17:00:00+00:00 0
2026-01-29 18:00:00+00:00 0 ,
(17272, 6))
```

2) Limpieza básica + cheques

```
En [8]: # Quitar velas inválidas (muy raro, pero buena práctica)
df = df [( df [ "high" ] >= df [ "low" ]) & ( df [ "close" ] > 0 )] . Co

# Asegura índice datetime y orden
df = df . sort_index ()
df . index = pd . to_datetime ( df . index )

df . isna () . suma ()
```

```
Salida[... adj cerrar 0
cerrar 0
alto 0
bajo 0
abierto 0
volumen 0
tipo de dato: int64
```

```
En [9]: # Verificando que el DataFrame no esté vacío
if df . vacío :
    elevar ValueError ( "El DataFrame está vacío después de la limpieza de dato
df . cabeza (), df . cola (), df . forma
df . información ()
df . describir ()
df . índice . min (), df . índice . máximo ()
```

```
df . índice . freq # Ninguno porque hay huecos en los datos
df = df . asfreq ( 'H' ) # Reindexa con frecuencia horaria
df . isna () . suma ()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
DatetimeIndex: 17272 entradas, del 13/04/2023 a las 23:00:00 y 00:00 al 29/01/202
6 a las 18:00:00 y 00:00
```

```
Columnas de datos (6 columnas en total):
```

```
# Columna Conteo no nulo Dtype
```

```
-----
0 adj close 17272 float64 no nulo
1 cierre 17272 float64 no nulo
2 alto 17272 no nulo float64
3 bajo 17272 no nulo float64
4 abierto 17272 float64 no nulo
5 volumen 17272 int64 no nulo
```

```
tipos de datos: float64(5), int64(1)
```

```
uso de memoria: 944,6 KB
```

```
C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\2343294818.py:9: FutureWarn
ing: 'H' está obsoleto y se eliminará en una versión futura, utilice 'h' en su lu
gar.
```

```
df = df.asfreq('H') # Reindexa con frecuencia horaria
```

```
Salida[... adj cerrar 7252
cerrar 7252
alto 7252
bajo 7252
abierto 7252
volumen 7252
tipo de dato: int64
```

3) Funciones (solo con información pasada)

Estas características son “clásicas” y seguras para empezar.

```
En [10]: def add_features ( datos : pd.DataFrame ) -> pd.DataFrame : d = datos.copy ( )

# Retornos
d [ "ret_1" ] = d [ "cerrar" ] . pct_change ( 1 )
d [ "ret_3" ] = d [ "cerrar" ] . pct_change ( 3 )
d [ "ret_6" ] = d [ "cerrar" ] . pct_change ( 6 )
d [ "ret_12" ] = d [ "cerrar" ] . pct_change ( 12 )
d [ "ret_24" ] = d [ "cerrar" ] . pct_change ( 24 )

# Medias
d [ "sma_10" ] = d [ "close" ] . rolling ( 10 ) . mean ( )
d [ "sma_30" ] = d [ "close" ] . rolling ( 30 ) . mean ( )
d [ "sma_ratio" ] = d [ "sma_10" ] / d [ "sma_30" ] - 1

# Volatilidad rolling
d [ "vol_20" ] = d [ "ret_1" ] . rolling ( 20 ) . std ( )

# Rango relativo (proxy ATR simple)
d [ "range" ] = ( d [ "high" ] - d [ "low" ] ) / d [ "close" ]

# Velas (cuerpo y mechas) - normalizadas
d [ "body" ] = ( d [ "close" ] - d [ "open" ] ) / d [ "close" ]
d [ "upper_wick" ] = ( d [ "high" ] - d [ [ "close" , "open" ] ] . max ( a
```

```

d [ "lower_wick" ] = ( d [ [ "close" , "open" ] ] . min ( axis = 1 ) - d [
    devolver d

df_feat = add_features ( df )
df_feat = df_feat.dropna ( ) . copy ( ) df_feat.head ( )

```

C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\3869259806.py:5: FutureWarning: El método predeterminado fill_method='pad' en Series.pct_change está obsoleto y se eliminará en una versión futura. Complete los valores NA no iniciales antes de llamar a pct_change o especifique 'fill_method=None' para no completar los valores NA.

```
d["ret_1"] = d["cerrar"].pct_change(1)
```

C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\3869259806.py:6: FutureWarning: El método predeterminado fill_method='pad' en Series.pct_change está obsoleto y se eliminará en una versión futura. Complete los valores NA no iniciales antes de llamar a pct_change o especifique 'fill_method=None' para no completar los valores NA.

```
d["ret_3"] = d["cerrar"].pct_change(3)
```

C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\3869259806.py:7: FutureWarning: El método predeterminado fill_method='pad' en Series.pct_change está obsoleto y se eliminará en una versión futura. Complete los valores NA no iniciales antes de llamar a pct_change o especifique 'fill_method=None' para no completar los valores NA.

```
d["ret_6"] = d["cerrar"].pct_change(6)
```

C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\3869259806.py:8: FutureWarning: El método predeterminado fill_method='pad' en Series.pct_change está obsoleto y se eliminará en una versión futura. Complete los valores NA no iniciales antes de llamar a pct_change o especifique 'fill_method=None' para no completar los valores NA.

```
d["ret_12"] = d["cerrar"].pct_change(12)
```

C:\Users\TRENDINGPC\AppData\Local\Temp\ipykernel_6056\3869259806.py:9: FutureWarning: El método predeterminado fill_method='pad' en Series.pct_change está obsoleto y se eliminará en una versión futura. Complete los valores NA no iniciales antes de llamar a pct_change o especifique 'fill_method=None' para no completar los valores NA.

```
d["ret_24"] = d["cerrar"].pct_change(24)
```


Salida[...

	adj cerrar	cerca	alto	bajo	abierto	volumen	ret_1	
Fecha y hora								
18/04/2023 04:00:00+00:00	1.094212	1.094212	1.094331	1.093733	1.093853	0.0	0.000000	0
18/04/2023 05:00:00+00:00	1.094811	1.094811	1.094811	1.093733	1.094092	0.0	0,000547	0
18/04/2023 06:00:00+00:00	1.095290	1.095290	1.095410	1.093853	1.094451	0.0	0,000438	0
18/04/2023 07:00:00+00:00	1.097333	1.097333	1.097333	1.095050	1.095170	0.0	0,001865	0
18/04/2023 08:00:00+00:00	1.098418	1.098418	1.098539	1.096972	1.096972	0.0	0,000989	0

4) Objetivo: dirección de la próxima vela (t+1)

```
En [11]: # Target: 1 si la próxima vela arriba cierra; 0 si no
df_feat [ "objetivo" ] = ( df_feat [ "cerrar" ] . shift ( - 1 ) > df_feat [

# Quitamos la última fila (no tiene target)
df_feat = df_feat . dropna () . Copiar ()

feature_cols = [
    "ret_1" , "ret_3" , "ret_6" , "ret_12" , "ret_24" ,
    "sma_ratio" , "vol_20" , "rango" ,
    "cuerpo" , "mecha superior" , "mecha inferior"
]

X = df_feat [ columnas_de_características ]
y = df_feat [ "objetivo" ]

X . forma , y . valor_cuenta ( normalizar = Verdadero )
```

Salida[... ((12862, 11),
objetivo
0 0.540662
1 0,459338
Nombre: proporción, tipo de dato: float64)

5) División temporal (80/20) + línea base y Gradient Boosting

```
En [12]: dividir = int ( len ( df_feat ) * 0.8 )

X_tren , X_prueba = X . iloc [: dividir ] , X . iloc [ división : ] y_train , y
```

```

de sklearn.metrics importar classification_report , roc_auc_score de sklearn.pip

pipe_lr = Pipeline ([
    ( "scaler" , StandardScaler ()),
    ( "clf" , LogisticRegression ( max_iter = 300 ))
])

pipe_lr . fit ( X_train , y_train )
proba_lr = pipe_lr . predict_proba ( X_test )[:, 1 ]
pred_lr = ( proba_lr >= 0.5 ) . astype ( int )

imprimir ( "LogReg AUC:" , puntuación_auc_roc ( y_test , proba_lr ))
imprimir ( informe_clasificación ( y_test , pred_lr ))

```

Área bajo la curva de registro log: 0,5265316690120682

Soporte para puntuación f1 de recuperación de precisión

0 0,55 0,97 0,70 1403

1 0,54 0,04 0,07 1170

precisión 0,55 2573

promedio macro 0,55 0,51 0,39 2573

promedio ponderado 0,55 0,55 0,41 2573

Ahora Gradient Boosting (suele funcionar muy bien como primer "modelo serio" sin libs extra):

En [13]:

```

de sklearn.ensemble importar GradientBoostingClassifier

gb = GradientBoostingClassifier ( estado_aleatorio = 42 )
gb .fit ( tren_X , tren_Y )

proba_gb = gb .predic_proba ( X_test )[:, 1 ] pred_gb = ( proba_gb >= 0.5 )

imprimir ( "GB AUC:" , puntuación_auc_roc ( y_test , proba_gb ))
imprimir ( informe_clasificación ( y_test , pred_gb ))

```

GB AUC: 0,5310525065336185

Soporte para puntuación f1 de recuperación de precisión

0 0,55 0,83 0,66 1403

1 0,48 0,19 0,27 1170

precisión 0,54 2573

promedio macro 0,52 0,51 0,47 2573

promedio ponderado 0,52 0,54 0,48 2573

6) Backtest con costos (spread + comisión + slippage)

Supuestos de costos (ajustables)

- **spread_pips:** spread promedio en pips (EURUSD típico 0,6–1,5 pips en broker decente)
- **slippage_pips:** Deslizamiento promedio por entrada/salida
- **Commission_per_round_turn:** comisión por ida y vuelta (si aplica). En FX retail a veces viene “embebida” en spread (pon 0).

1 pip en EURUSD = 0,0001 (aprox).

Costos se aplican **cuando cambias de posición** (trade).

```
En [14]: import matplotlib.pyplot as plt

def backtest_directional(
    prices: pd.Series,
    proba: pd.Series,
    threshold: float = 0.55,
    spread_pips: float = 1.0,
    slippage_pips: float = 0.2,
    commission_round_turn: float = 0.0
):
    """
    Estrategia:
    - si P(up) >= threshold -> long (+1)
    - si P(up) <= 1-threshold -> short (-1)
    - si está en zona gris -> flat (0)
    Ejecuta en la siguiente vela (retorno t->t+1).
    Costos:
    - al cambiar posición se paga (spread+slippage) en pips (por lado aproximado)
    """
    df_bt = pd.DataFrame({"close": prices, "proba": proba}).copy()

    # Retorno forward (siguiente vela)
    df_bt["fwd_ret"] = df_bt["close"].pct_change().shift(-1)

    # Señal con zona gris
    df_bt["pos"] = 0
    df_bt.loc[df_bt["proba"] >= threshold, "pos"] = 1
    df_bt.loc[df_bt["proba"] <= (1 - threshold), "pos"] = -1

    # Cambio de posición (trade)
    df_bt["pos_prev"] = df_bt["pos"].shift(1).fillna(0)
    df_bt["trade"] = (df_bt["pos"] != df_bt["pos_prev"]).astype(int)

    # Costo por trade en retorno:
    # Convertimos pips a "return" aproximado dividiendo por precio.
    pip_value = 0.0001
    total_pips = spread_pips + slippage_pips # simplificación
    df_bt["cost_ret"] = df_bt["trade"] * (total_pips * pip_value) / df_bt["close"]

    # Comisión: si quieres modelarla como retorno fijo por trade (aprox), puedes
    # Para mantenerlo simple la dejamos como 0 o la conviertes a ret sobre capit
    df_bt["commission_ret"] = df_bt["trade"] * commission_round_turn

    # Retorno estrategia
    df_bt["strategy_ret"] = df_bt["pos"] * df_bt["fwd_ret"] - df_bt["cost_ret"]

    df_bt = df_bt.dropna().copy()
    df_bt["equity"] = (1 + df_bt["strategy_ret"]).cumprod()
```

```
df_bt["buy_hold"] = (1 + df_bt["fwd_ret"]).cumprod()

return df_bt

# Backtest en el set de test usando Gradient Boosting:
df_test = df_feat.iloc[split:].copy()
proba_series = pd.Series(proba_gb, index=df_test.index)

bt = backtest_directional(
    prices=df_test["close"],
    proba=proba_series,
    threshold=0.55,
    spread_pips=1.0,
    slippage_pips=0.2,
    commission_round_turn=0.0
)

bt[["equity", "buy_hold"]].tail()
```

Out[14]:

	equity	buy_hold
Datetime		
2026-01-29 13:00:00+00:00	0.895790	1.020951
2026-01-29 14:00:00+00:00	0.898587	1.017661
2026-01-29 15:00:00+00:00	0.898496	1.018755
2026-01-29 16:00:00+00:00	0.898496	1.019852
2026-01-29 17:00:00+00:00	0.898836	1.019364

Plot equity:

```
In [15]: plt.figure()
plt.plot(bt.index, bt["equity"], label="Strategy (costs)")
plt.plot(bt.index, bt["buy_hold"], label="Buy&Hold")
plt.legend()
plt.title("EURUSD 1H - Equity Curve (Test)")
plt.show()
```

