

## INTRODUCCIÓN

## OBJETIVO

## FUNCIONAMIENTO

- [1.1 Recogida de datos](#)
- [1.2 Tipos de indicadores](#)
- [2 Filtrado de indicadores](#)
- [3 Entrenar modelos TensorFlow, XGB y Sklearn](#)
- [4.1 Evaluar la CALIDAD de esos modelos](#)
- [4.2 Evaluar esos BENEFICIO real de modelos](#)
- [5.1 Hacer predicciones de la última semana](#)
- [5.2 Mandar alertas en tiempo real](#)

## Puesta en marcha rápida

## Puesta en marcha

- [1 Recogida de datos históricos](#)
  - [1.0 \(Recomendado\) alphavantage API](#)
  - [1.1 Hay que generar el histórico de OHLCV de la acción](#)
- [2 Filtrado de indicadores técnicos](#)
- [3 Generar entrenamiento de modelos TensorFlow, XGB y Sklearn](#)
- [4 Evaluar calidad de los modelos predictivos](#)
- [5 Predicciones](#)
  - [5.0 hacer predicciones de la última semana Test Opcional](#)
  - [5.1 Conseguir datos OHLCV en tiempo real](#)
  - [5.2 Configurar chatID y tokens en Telegram](#)
  - [5.3 Envío de alertas en tiempo real Telegram](#)

## Posibles mejoras

- [Mejoras en los modelos predictivos, usando multidimensionales](#)
- [Revisar la manera de obtención de ground true](#)
- [Añadir el indicador de sentimientos noticias](#)
- [Añadir balances económicos](#)
- [Lista de mejoras sugeridas:](#)

## Nombres de indicadores:

## INTRODUCCIÓN

El mercado bursátil se mueve por indicadores técnicos, existen varios tipos de volatilidad, de volumen de ciclo, de velas, soportes, resistencias, medias móviles...

Una excelente página para ver todos los indicadores técnicos bursátiles es webull

<https://app.webull.com/trade?source=seo-google-home>

Imagen: webull con indicadores Estocástico, MACD y RSI



Sobre las gráficas de bolsa se ha inventado TODO lo posible para predecir la bolsa , con resultados dispares, dejando claro la dificultad de predecir el comportamiento humano

Estos indicadores indican donde comprar y vender , existen muchas creencias sobre ellos (nos referimos en creencias , ya que si funcionaran siempre todos seriamos ricos)

Todo indicador técnico se puede obtener mediante operaciones matemáticas programables

Tres ejemplos:

**RSI** o índice de fuerza relativa es un oscilador que refleja la fuerza relativa

Mayor que 70 sobrecomprado , indica que va a bajar

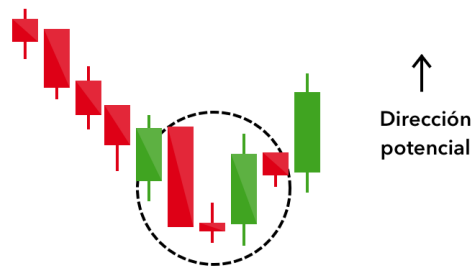
Menos que 70 sobrevendido, indica que va a subir

**MACD** es el acrónimo de Moving Average Convergence / Divergence. El MACD en bolsa se utiliza para medir la robustez del movimiento del precio. A través del cruce de la línea de este indicador y de la media

Se opera atendiendo a los cruces entre estas dos líneas

O bien se opera cuando ambas superan el cero

**Vela: Estrella de la mañana** El patrón de estrella de la mañana se considera una señal de esperanza en una tendencia bajista del mercado

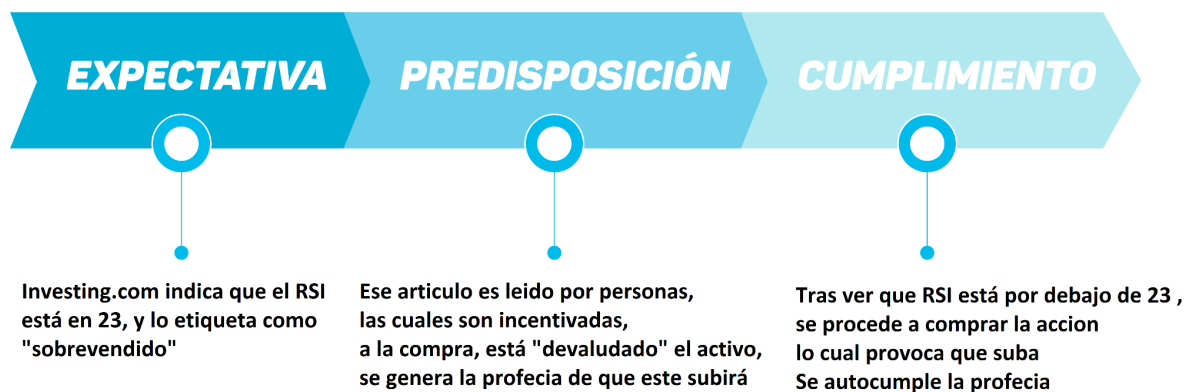


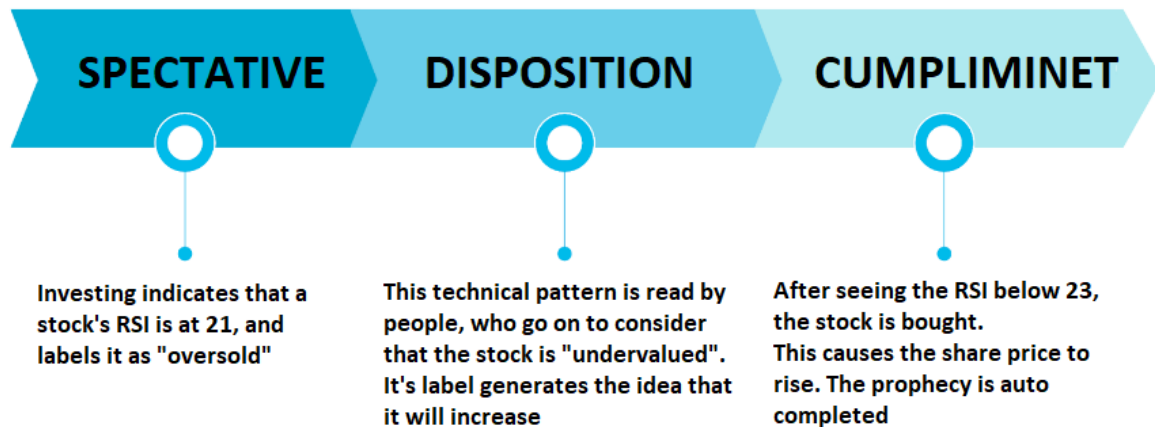
Estos indicadores están presentes, en refutadas y populares páginas web como investing.com para ser analizados por el mercado

<https://es.investing.com/equities/apple-computer-inc-technical>

La dificultad es mayúscula de predecir el precio de cualquier acción. Inflación, guerras, populismos, todo ello condiciona la economía, y se hace difícil, sino imposible predecir qué hará Vladimir Putin mañana.

Aquí entra el principio de profecía autocumplida de explicado es, al principio, una definición “falsa” de la situación, que despierta un nuevo comportamiento que hace que la falsa concepción original de la situación se vuelva “verdadera”. Ejemplo:





## OBJETIVO

Entendiendo el principio de profecía autocumplida, se puede obtener el patrón de las mismas, mediante la recogida masiva de patrones técnicos, su cálculo y el estudio de sus patrones

Para ello , se emplearán técnica como el big data mediante librerías Pandas Python, el machine learning mediante Sklearn, XGB y redes neuronales mediante la librería abierta de google Tensor Flow

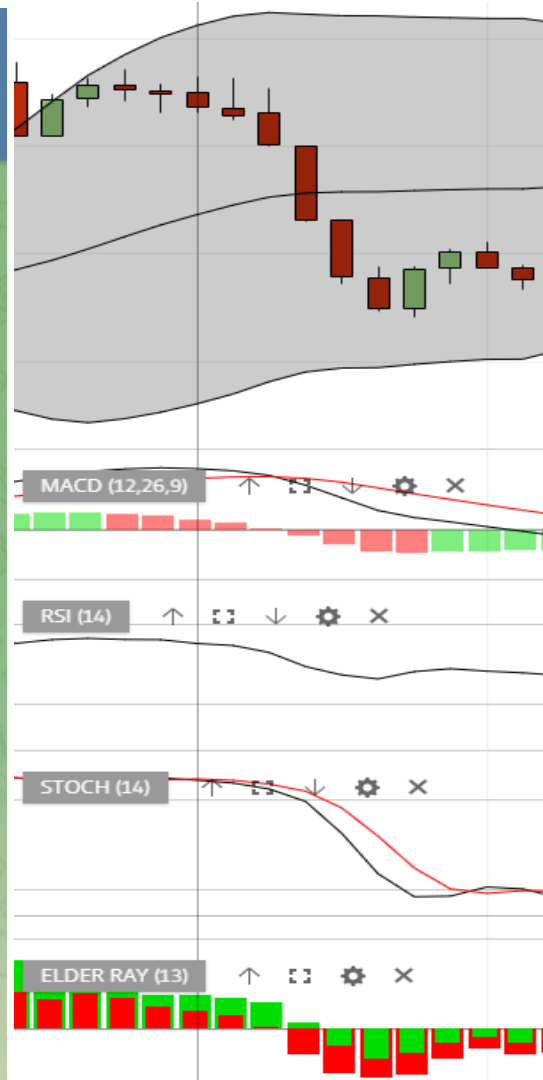
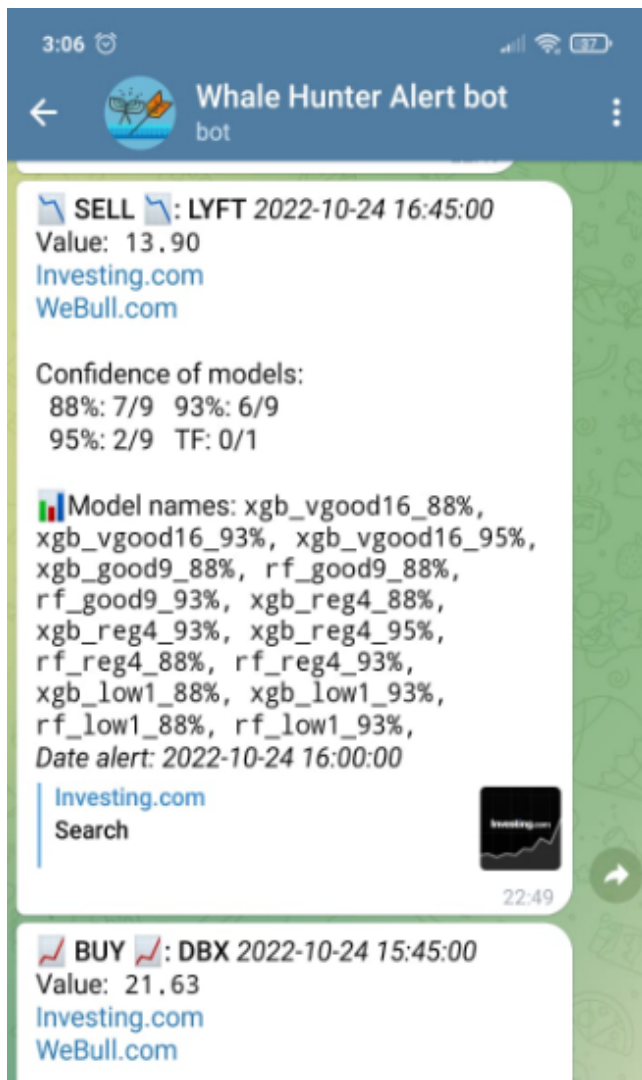
El resultado se mostrará de forma sencilla y amigable mediante alertas en móvil o ordenador.

Ejemplo de alerta en tiempo real a través del bot de telegram

[https://t.me/Whale\\_Hunter\\_Alertbot](https://t.me/Whale_Hunter_Alertbot)

Los modelos machine learnig Sklearn, XGB y Tensor Flow , mediante el aprendizaje de los últimos meses detectan el punto de venta. Para detectar este punto de venta se han tenido en cuenta una serie de indicadores: `olap_VMAP`, `ma_SMA_50`, `ichi_senkou_a`, `olap_BBAND_dif`, `mtum_MACD_ext`, `olap_BBAND_MIDDLE`, `mtum_MACD_ext_signal`, `fibo_s1`, `volu_PVI_1`, `ma_KAMA_5`, etcétera.

En la imagen se muestran: MACD, RSI , estocástico y Balance of power (Elder Ray)  
La alerta es mandada en la línea vertical, durante los 4 siguientes periodos la acción decrece un 2,4%. Cada periodo-vela de la imagen indica 15 minutos.



## FUNCIONAMIENTO

### 1.1 Recogida de datos

Recoger datos para entrenar el modelo

`yhoo_generate_big_all_csv.py`

Se obtienen los datos de cierre, mediante yahoo API finance, y se calculan centenares de patrones técnicos mediante las librerías pandas y talib

`yhoo_history_stock.get_SCALE_csv_stocks_history_Download_list()`

El modelo para ser capaz de entrenarse en detectar puntos de compra y de venta, crea la columna `buy_sell_point` tiene valor de 0, -100, 100. Estos son detectados en función de

los máximos cambios, (positivos 100, negativo -100) en el histórico de los últimos meses, este punto será con los que se entrene el entrenamiento, también llamado el *ground true*. Se asignará valor en `buy_sell_point` si el incremento o decremento de la acción es mayor del 2,5% en un periodo de 3 horas, mediante la función `get_buy_sell_points_Roll`.

Una vez obtenidos los datos históricos de la acción y calculados todos los indicadores técnicos, un total de 1068, se generan ficheros de tipo `AAPL_stock_history_MONTH_3_AD.csv`.

Ejemplo del fichero con los ocho primeros indicadores:

Date	buy_sell_point	Open	High	Low	Close	Volume	per_Close	per_Volume	has_preMarket	per_preMarket	olap_BBAND_UPPER	olap_BBAND_MIDDLE	olap_BBAND_LOWER
2022-08-05 13:45:00	0	48.040001	48.105	47.75	47.91	272406	-0.477772	-8.184974	False	0.0	50.433376	46.737485	43.041594
2022-08-05 15:00:00	0	47.810001	47.93	47.709999	47.91	238543	0.0	-12.431077	False	0.0	50.530725	46.874485	43.218245
2022-08-05 15:30:00	0	47.721001	47.990002	47.700001	47.950001	316013	0.083492	32.476325	False	0.0	50.602601	47.020735	43.438869
2022-08-05 15:45:00	0	47.959999	48.240002	47.880001	48.02	688875	0.145985	117.98945	False	0.0	50.603329	47.198735	43.794141
2022-08-08 10:45:00	0	51.27	51.490002	50.68	51.459999	360504	8.01847	-47.667719	True	6.768013	51.196945	47.544235	43.891525
2022-08-08 11:00:00	0	51.459999	51.710701	50.900002	50.900002	303730	-1.088219	-15.748508	False	0.0	51.516801	47.863735	44.21067
2022-08-08 11:30:00	0	50.334999	50.889999	49.880001	49.935001	419471	-1.895875	38.106542	False	0.0	51.620354	48.110485	44.600617
2022-08-08 11:45:00	0	49.919998	50.75	49.560001	50.66	291724	1.451884	-30.45431	False	0.0	51.75685	48.395985	45.035121
2022-08-08 12:00:00	0	50.68	50.740002	50.130001	50.330002	284326	-0.651398	-2.535959	False	0.0	51.794547	48.645985	45.497423
2022-08-08 12:15:00	0	50.380001	50.380001	49.77	49.799999	248416	-1.053055	-12.629869	False	0.0	51.330039	48.929985	46.529931
2022-08-08 12:45:00	0	50.139999	50.3298	50.02	50.27	151315	0.943778	-39.088062	False	0.0	51.437796	49.070985	46.704175
2022-08-08 13:00:00	0	50.25	50.549099	50.169998	50.509998	145834	0.477418	-3.622245	False	0.0	51.5979	49.151755	46.70561
2022-08-08 14:00:00	0	50.52	50.568901	50.080002	50.34	178358	-0.336563	22.302069	False	0.0	51.645137	49.169755	46.694373
2022-08-08 14:15:00	0	50.360001	50.459999	50.209999	50.389999	140100	0.099323	-21.450117	False	0.0	51.764291	49.282755	46.801219
2022-08-08 14:30:00	0	50.380001	50.419998	50.040001	50.200001	147313	-0.377056	5.148465	False	0.0	51.777464	49.422255	47.067047
2022-08-08 15:15:00	0	50.040001	50.049999	49.706402	49.959999	314087	-0.478091	113.210647	False	0.0	51.814527	49.503755	47.192983
2022-08-08 15:30:00	0	49.950001	49.965	49.700001	49.868301	307301	-0.183542	-2.160548	False	0.0	51.844971	49.56667	47.288369

Esta obtención de datos es personalizable, se puede obtener y entrenar modelos de cualquier acción del Nasdaq, para otros indicadores o cripto-activos, también es posible mediante pequeños cambios.

A través de la clase `Option_Historical` existe la posibilidad de crear ficheros de datos históricos: anuales, mensuales y diarios.

```
class Option_Historical(Enum): YEARS_3 = 1, MONTH_3 = 2,
MONTH_3_AD = 3, DAY_6 = 4, DAY_1 = 5
```

Se generan los ficheros `\d_price\min_max\AAPL_min_max_stock_MONTH_3.csv`, los cuales guardan el max y min valor de cada columna, para que sea leído en `Model_predictions_Nrows.py` para un rápido `fit_scaler()` (es el proceso "limpieza" que requieren los datos antes de entrar en los modelos de entrenamiento AI). Esta operación es de vital importancia para una correcta optimización en la lectura de datos en tiempo real.

## 1.2 Tipos de indicadores

Durante la generación de fichero de recogida de datos del punto 1

`AAPL_stock_history_MONTH_3_AD.csv` se calculan 1068 indicadores técnicos, los cuales se dividen en subtipos, en función de **prefijos** en el nombre. Lista de prefijos y ejemplo de nombre de alguno de ellos.

- Overlap o superposición: **olap\_**  
olap\_BBAND\_UPPER, olap\_BBAND\_MIDDLE, olap\_BBAND\_LOWER,
- Momentum: **mtum\_**  
mtum\_MACD, mtum\_MACD\_signal, mtum\_RSI, mtum\_STOCH\_k,
- Volatilidad: **vola\_**

- vola\_KCBe\_20\_2, vola\_KCUE\_20\_2, vola\_RVI\_14
- Patrones de ciclo: **cycl\_**
  - cycl\_DCPHASE, cycl\_PHASOR\_inph, cycl\_PHASOR\_quad
- Patrones de velas: **cdl\_**
  - cdl\_RICKSHAWMAN, cdl\_RISEFALL3METHODS, cdl\_SEPARATINGLINES
- Estadística: **sti\_**
  - sti\_STDDEV, sti\_TSF, sti\_VAR
- Medias móviles: **ma\_**
  - ma\_SMA\_100, ma\_WMA\_10, ma\_DEMA\_20, ma\_EMA\_100, ma\_KAMA\_10,
- Tendencia: **tend\_ y ti\_**
  - tend\_renko\_TR, tend\_renko\_brick, ti\_acc\_dist, ti\_chaikin\_10\_3
- Resistencias y soportes sufijos: **\_s3, \_s2, \_s1, \_pp, \_r1, \_r2, \_r3**
  - fibo\_s3, fibo\_s2, fibo\_s1, fibo\_pp, fibo\_r1, fibo\_r2, fibo\_r3
  - demark\_s1, demark\_pp, demark\_r1
- Punto de intersección con resistencia o soporte: **pcrh\_**
  - pcrh\_demark\_s1, pcrh\_demark\_pp, pcrh\_demark\_r1
- Punto de intersección con media móvil o de medias móviles entre ellas: **mcrh\_**
  - mcrh\_SMA\_20\_TRIMA\_50, mcrh\_SMA\_20\_WMA\_50, mcrh\_SMA\_20\_DEMA\_100
- Indicadores de cambios en el índice bursátil , nasdaq: **NQ\_**
  - NQ\_SMA\_20, NQ\_SMA\_100

Nota: Para ver los 1068 indicadores usados ir a las hojas adjuntas al final del documento.

## 2 Filtrado de indicadores

Ejecutar para saber qué columnas son relevantes para la obtención del modelo

`Feature_selection_create_json.py`

Hay que saber cuáles de las centenares columnas de datos técnicos , es válida para entrenar el modelo neuronal, y cuales son solo ruido. Esto se hará mediante correlaciones y modelos de Random Forest.

Responde a la pregunta:

¿Qué columnas son las más relevantes para puntos de compra o venta?

Genera los ficheros *best\_selection*, los cuales son un raking de mejores datos técnicos para entrenar el modelo , se pretende pasar de 1068 columnas a unas 120

Por ejemplo, para la acción Amazon, detección puntos de compra, en el periodo Junio a Octubre 2022, los indicadores más valiosos son:

- Senkuo de la nube de Ichimoku
- Volatilidad de Chaikin
- On-balance volume

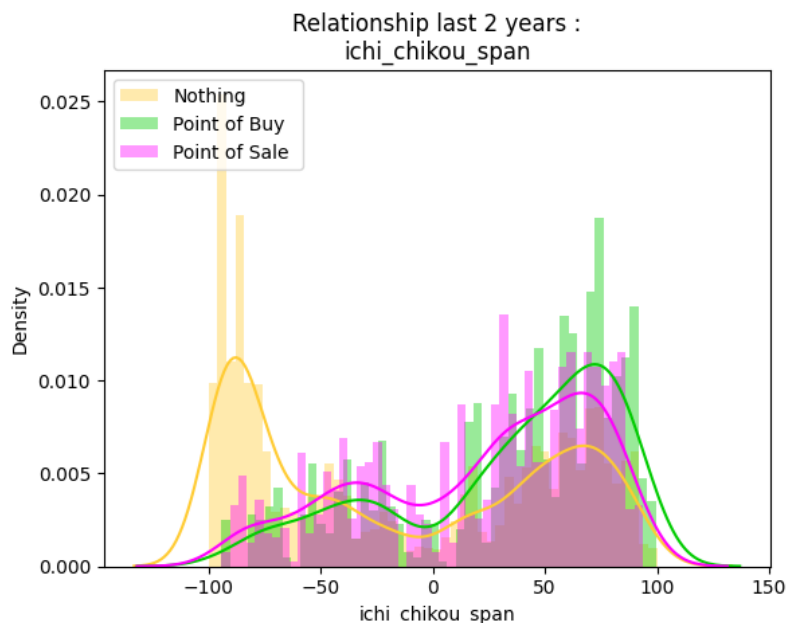
Ejemplo del fichero *plots\_relations/best\_selection\_AMNZ\_pos.json*

```

"index": {
  "12": [
    "ichi_chilou_span"
  ],
  "10": [
    "volu_ChaiKin_AD"
  ],
  "9": [
    "volu_OBV"
  ],

```

Se imprimen gráficos con los 3 mejores datos técnicos en la carpeta *plots\_relations/plot*  
Ejemplo de nombre: *TWLO\_neg\_buy\_sell\_point\_\_ichi\_chikou\_span.png*



### 3 Entrenar modelos TensorFlow, XGB y Sklearn

`Model_creation_models_for_a_stock.py`

para ello se requiere la selección de mejores columnas del punto #2

Hay cuatro tipos de algoritmos predictivos, modelos AI:

- **Gradient Boosting** está formado por un conjunto de árboles de decisión individuales, entrenados de forma secuencial, de forma que cada nuevo árbol trata de mejorar los errores de los árboles anteriores. Librería Sklearn
- **Random Forest** Los bosques aleatorios son un método de aprendizaje conjunto para clasificación, regresión y otras tareas que opera mediante la construcción de una multitud de árboles de decisión en el momento del entrenamiento. Librería Sklearn
- **XGBoost** es una biblioteca de aumento de gradiente distribuida optimizada diseñada para ser altamente eficiente, flexible y portátil. Implementa algoritmos de aprendizaje automático bajo el marco Gradient Boosting. Librería XGBoost
- **TensorFlow TF** es una biblioteca de código abierto para aprendizaje automático a través de un rango de tareas, y desarrollado por Google para satisfacer sus



necesidades de sistemas capaces de construir y entrenar redes neuronales para detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos. Librería TensorFlow

Existen modelos POS (compra) o modelo NEG (venta) y existe un modelo BOTH (BOTH se descarta, ya que los modelos de predicción son binario, sólo aceptan 2 posiciones, verdad o mentira)

Este punto genera modelos de predicción `.sav` para XGB y Sklearn. `.h5` para Tensor Flow

Ejemplos de nombrado: `XGboost_U_neg_vgood16_s28.sav` y `TF_AMZN_pos_low1_s128.h5`

Formato de los nombres:

- Tipo de AI con las que se entrena puede ser:
  - `XGboost`, `TF`, `TF64`, `GradientBoost` y `RandomForest`
- Ticker de la acción `AMZN` para amazon, `AAPL` para Apple ...
- Detecta puntos de compra o de venta `pos` o `neg`
- Cuantos indicadores han sido usados en el aprendizaje, pueden ser de 4 tipos en función de la relevancia dada por el punto #2 *Filtrado de indicadores*. Este ranking es organizado en la clase `MODEL_TYPE_COLM`,
  - `vgood16` los mejores 16 indicadores
  - `good9` los mejores 32 indicadores
  - `reg4` los mejores 64 indicadores
  - `low1` los mejores 128 indicadores
- Solo para modelos TF. En función de la densidad de las neuronas usadas, definidas en la clase `a_manage_stocks_dict`. `MODEL_TF_DENSE_TYPE_ONE_DIMENSI` puede tomar valor: `s28`, `s64` y `s128`

Estas combinaciones implican que por cada acción se crean 5 tipos de AI, cada una en `pos` y `neg`, además por cada combinación se añade las 4 configuraciones de indicadores técnicos. Esto genera 40 modelos de AI, los cuales serán seleccionados en el punto: #4 *evaluar la CALIDAD de esos modelos*

Cada vez que se genera un modelo IA, se genera un fichero de registro:

`TF_balance\TF_AAPL_pos_reg4.h5_accuracy_87.6%__loss_2.74__epochs_10[160].csv`

Este contiene los datos de precisión y de pérdida del modelo, así como los registros del entrenamiento del mismo.

#### 4.1 Evaluar la CALIDAD de esos modelos

`Model_creation_scoring.py`

Para hacer una predicción con los AI, se recogen nuevos datos y se calculan los indicadores técnicos con los que ha sido entrenado según los ficheros *best\_selection*.

Cuando los modelos `.h5` y `.sav` son preguntados:

¿Esto es un punto de compra-venta?

Estos responden un número que puede variar entre 0,1 y 4  
Cuanto más alto sea el número mayor probabilidad de que sea un punto de compra-venta correcto.

Cada modelo tiene una escala de puntuación en el cual se considera punto de compra venta. para unos modelos con una puntuación de más 0,4 será suficiente (normalmente los `XGboost`) , mientras que para otros requieren más de 1,5 (normalmente los `TF`).

¿Cómo se sabe cuál es la puntuación umbral para cada modelo ?

La clase `Model_creation_scoring.py` genera los ficheros umbral de puntuación *threshold*, los cuales dicen cual es el punto umbral en el cual se considera punto de compra-venta.

Cada modelo AI contará con su propio fichero de tipo:

*Models/Scoring/AAPL\_neg\_\_when\_model\_ok\_threshold.csv*

Por cada acción el punto #3 *entrenar los modelos TF, XGB y Sklearn* se generan 40 modelos de AI. Esta clase evalúa y selecciona los modelos más precisos de tal forma que solo se ejecutarán en tiempo real los más precisos (normalmente se seleccionan entre 4 y 8)

*Models/Scoring/AAPL\_neg\_\_groupby\_buy\_sell\_point\_000.json*

```
"list_good_params": [  
    "r_rf_AFRM_pos_low1_",  
    "r_TF64_AFRM_pos_vgood16_",  
    "r_TF64_AFRM_pos_good9_",  
    "r_TF_AFRM_pos_reg4_"  
],
```

## 4.2 Evaluar esos BENEFICIO real de modelos

`Model_predictions_N_eval_profits.py`

Responde a la pregunta:

¿Si se deja N días ejecutándose, cuánto dinero hipotético se gana ?

Nota: esto debe ejecutarse con datos que no hayan sido usados en el modelo de entrenamiento, preferentemente

*Models/eval\_Profits/\_AAPL\_neg\_ALL\_stock\_20221021\_\_20221014.csv*

## 5.1 Hacer predicciones de la última semana

`Model_predictions_Nrows.py`

Se puede hacer predicciones con los datos en tiempo real de la acción.

A través de la llamada a la función cada 10-12min, descarga los datos de la acción en tiempo real a través de la API financiera de yahoo.

```
df_compar, df_vender = get_RealTime_buy_sell_points()
```

Esta ejecución genera el fichero de registro *d\_result/prediction\_results\_N\_rows.csv*

Este fichero y las notificaciones(telegram y mail ) contienen información sobre cada predicción que se ha realizado. Contiene las siguientes columnas:

- Date: fecha de las predicción
- Stock: acción
- buy\_sell: puede ser NEG o POS , dependiendo de que sea operación de compra o venta
- Close: Es el valor escalado del valor de cierre (no el valor real)
- Volume: Es el valor escalado del Volumen (no el valor real)
- 88%: Formato fracción ( 5/6 ) ¿Cuántos modelos han predicho que es un punto de operación válido por encima del 88%? Cinco de los seis analizados
- 93%: Formato fracción ( 5/6 ), número de modelos por encima de 93%
- 95%: Formato fracción ( 5/6 ), número de modelos por encima de 95%
- TF: Formato fracción ( 5/6 ), número de modelos por encima de 93%, cuya predicción se ha hecho con modelos Tensor Flow
- Models\_names: nombre de los modelos que han dado positivo, con el % de acierto (88%, 93%, 95%) como sufijo

Ejemplo de registro

```
2022-11-07 16:00:00 MELI NEG -51.8 -85.80 5/6 0/6 0/6 1/2
TF_reg4_s128_88%, rf_good9_88%, rf_low1_88%, rf_reg4_88%,
rf_vgood16_88%,
```

Para ser considerada una predicción válida para operar, al menos, debe tener la mitad de la fracción de puntuación en las columnas 93% y TF .

Más de la mitad de los modelos han predicho con una puntuación superior al 93% que es un punto bueno para operar

## 5.2 Mandar alertas en tiempo real

`predict_POOL_enque_Thread.py` *multihilo encolado 2s por acción*

Es posible ejecutarlo sin configurar telegram punto 5.2, en ese caso no se enviarán alertas en telegram, pero si se registraron los resultados en tiempo real en:

`d_result/prediction_real_time.csv`

Existe la posibilidad de mandar las alertas de compra venta de la acción, al telegram o mail se evalúan los múltiples modelos entrenados AI , y solo los mayores de 96% de probabilidad (según lo entrenado previamente) son notificados

Cada 15 minutos , se calculan **todos** los indicadores necesarios en real time por cada acción y se evalúan en los modelos AI

En la alerta se indica qué modelos están detectando puntos de compra y venta, correctos en los que ejecutar la transacción

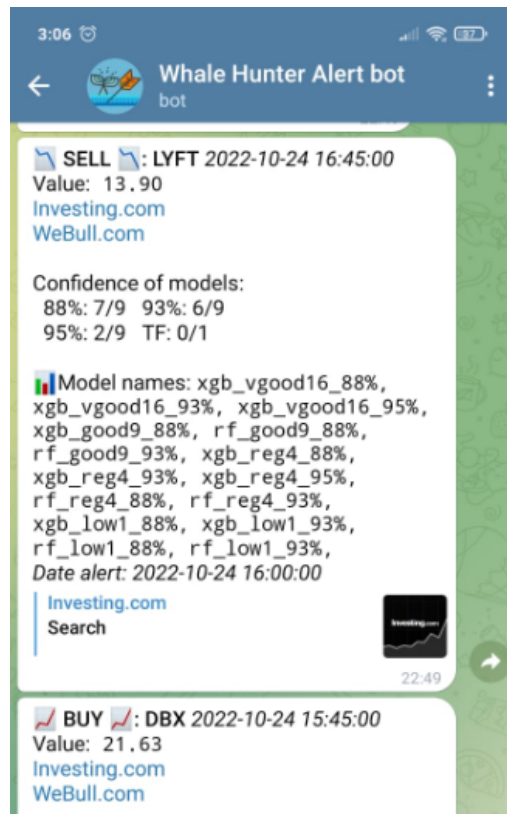
Estas alertas de compra-venta caducan en, más menos 7 minutos, dado la volatilidad del mercado

También se adjunta el precio al que se detectó, la hora, y los enlaces a las web de noticias.

Nota: las noticias financieras siempre deben prevalecer a los indicadores técnicos.

Lo que se muestra en alerta DEBUG, es la información de `d_result/prediction_results_N_rows.csv` del Punto: 5 hacer predicciones de la última semana Test

Para entender la información completa de la alerta ver Punto 5.1 Hacer predicciones de la última semana



## Puesta en marcha rápida

### Instalar requisitos

```
pip install -r requirements.txt
```

Ejecutar `Utils/API_alphavantage_get_old_history.py`

Ejecutar `yhoo_generate_big_all_csv.py`

Ejecutar `Model_creation_models_for_a_stock.py`

Ejecutar `Model_creation_scoring.py`

Ejecutar `Model_predictions_Nrows.py` Opcional, predicciones ultima semana

Predicciones tiempo real:

Ejecutar `Utils/Volume_WeBull_get_tickers.py` Ignorar en caso de usar configuración por defecto

Configurar token bot ver punto 5.2 Configurar chatID y tokens en Telegram

Ejecutar `predict_POOL_enque_Thread.py`

Es posible ejecutarlo sin configurar telegram punto 5.2, en ese caso no se enviarán alertas en telegram, pero si se registraron los resultados en tiempo real en:

`d_result/prediction_real_time.csv`

## Puesta en marcha

(Los tiempos de ejecución son estimados para un intel i3 y 8GB de RAM)

**0.0** El intérprete con el que se ha hecho el tutorial es python 3.8 , IDE Pycharm, precaución con la compatibilidad entre versiones de la librería pandas y python

Por ejemplo: a dia de hoy no usar python 3.10 , ya que es incompatible con pandas

<https://stackoverflow.com/questions/69586701/unable-to-install-pandas-for-python>

**0.1** Descargar e instalar requisitos, el proyecto es potente y exigente en cuanto a librerías

```
pip install -r requirements.txt
```

**0.2** Buscar en todos los ficheros la cadena `**DOCU**`

esto permite observar a todos los ficheros que son ejecutables del tutorial puesta en marcha fácilmente

**0.3** En el fichero `a_manage_stocks_dict.py` se guardan todas las configuraciones, observar el documento y saber donde se encuentra.

En él existe el diccionario `DICT_COMPANY`

El cual contiene los ID (GameStops cotiza con el ID: **GME**) de las empresas a analizar se puede personalizar y crear clase a partir de los tikers del **nasdaq** , por defecto se usará la clave **@FOLO3** la cual analizará estas 39 empresas.

**"@FOLO3"** :

```
[ "UPST", "MELI", "TWLO", "RIVN", "SNOW", "LYFT", "ADBE", "UBER",  
  "ZI", "QCOM", "PYPL", "SPOT", "RUN", "GTLB", "MDB", "NVDA", "AMD" ,  
  "ADSK", "AMZN", "CRWD", "NVST", "HUBS", "EPAM", "PINS", "TTD",  
  "SNAP", "APPS", "ASAN", "AFRM", "DOCN", "ETSY", "DDOG", "SHOP",  
  "NIO", "U", "GME", "RBLX", "CRSR" ],
```

Si se desea una ejecución más rápida se recomienda borrar elementos de la lista y dejar tres

## 1 Recogida de datos históricos

### 1.0 (Recomendado) alphavantage API

La API yfinance, si se desea intervalos entre precio y precio en intervalos de 15min está limitada a 2 meses, para obtener más datos de tiempo hasta de 2 años atrás (a más datos mejores modelos predictivos) se usa la versión gratuita de la API

<https://www.alphavantage.co/documentation/>

**Ejecutar** `Utils/API_alphavantage_get_old_history.py`

La clase es personalizable: intervalos de acción, meses a preguntar y acción ID

Nota: al ser la versión gratuita, existe retraso entre petición y petición, para obtener un solo historico de 2 años se toma 2-3 minutos por acción

Una vez ejecutado se rellenará la carpeta: `d_price/RAW_alpha` con .csv históricos OHLCV de precios de acción. Estos ficheros serán leídos en el paso siguiente. Ejemplo de nombre:

`alpha_GOOG_15min_20221031__20201112.csv`

Revisar que se hayan generado uno por cada acción en `d_price/RAW_alpha`.

### 1.1 Hay que generar el histórico de OHLCV de la acción

Así como el histórico de patrones técnicos. En calcular todos los patrones técnicos tarda +-1 minuto por acción

**Ejecutar** `yhoo_generate_big_all_csv.py`

Una vez ejecutado se rellenará la carpeta: `d_price` con .csv históricos OHLCV de precios de acción.

Tres tipos de ficheros son generados ( Ejemplo de tipo de nombre para acción: AMD):

- `AMD_SCALA_stock_history_MONTH_3_AD.csv` con todos los patrones técnicos calculados y aplicado un fit scaler(-100, 100), es decir los precios de acción están escalados (tamaño: 30-90mb)
- `d_price/min_max/AMD_min_max_stock_MONTH_3_AD.csv` con las claves del escalado (tamaño: 2-7kb)
- `AMD_stock_history_MONTH_3_AD.csv` el histórico puro de los OHLCV (tamaño: 2-7mb)

Nota: `MONTH_3_AD` significa 3 meses de API yfinance más el histórico recogido de alphavantage. Punto 1.0

Revisar que se hayan generado uno por cada acción.

## 2 Filtrado de indicadores técnicos

Hay que separar los indicadores técnicos que tienen relación con los puntos de compra o venta y cuales son ruido. 20 segundos por acción

**Ejecutar** `Model_creation_scoring.py`

Se generan tres ficheros por cada acción en la carpeta: `plots_relations`, relaciones para compra "pos", relaciones para venta "neg" y relaciones para ambos "both"

- `plots_relations/best_selection_AMD_both.json`

Estos ficheros contienen un ranking de qué indicador técnico es mejor para cada acción

Revisar que se hayan generado tres .json por cada acción en *plots\_relations* .

### 3 Generar entrenamiento de modelos TensorFlow, XGB y Sklearn

Entrenar los modelos , por cada acción se entrenan 36 modelos distintos.  
15 minutos por acción.

**Ejecutar** `Model_creation_models_for_a_stock.py`

Por cada acción se generan los siguientes ficheros:

Carpeta *Models/Sklearn\_smote*:

- XGboost\_AMD\_yyy\_xxx\_.sav
- RandomForest\_AMD\_yyy\_xxx\_.sav
- XGboost\_AMD\_yyy\_xxx\_.sav

Carpeta *Models/TF\_balance*:

- TF\_AMD\_yyy\_xxx\_zzz.h5
- TF\_AMD\_yyy\_xxx\_zzz.h5\_accuracy\_71.08%\_\_loss\_0.59\_\_epochs\_10[160].csv

xxx puede tomar valor `vgood16` `good9` `reg4` y `low1`

yyy puede tomar valor "pos" y "neg"

zzz puede tomar valor s28 s64 y s128

Revisar que se hayan generado todas las combinaciones de ficheros expuestos por cada acción en las subcarpetas de */Models*.

### 4 Evaluar calidad de los modelos predictivos

De los 36 modelos creados por cada historico OHLCV de cada acción, sólo se ejecutarán en tiempo real los mejores, para seleccionar y evaluar esos mejores

**Ejecutar** `Model_creation_scoring.py`

En la carpeta *Models/Scoring*

`AMD_yyy__groupby_buy_sell_point_000.json`

`AMD_yyy__when_model_ok_threshold.csv`

Revisar que se hayan generado dos por cada acción.

## 5 Predicciones

### 5.0 hacer predicciones de la última semana Test Opcional

**Ejecutar** `Model_predictions_Nrows.py`

Esta ejecución genera el fichero de registro `d_result/prediction_results_N_rows.csv`

Genera un fichero de ejemplo con predicciones de la última semana, datos obtenidos con yfinance

Revisar que existen los registros

## 5.1 Conseguir datos OHLCV en tiempo real

En caso de querer predecir acciones en la lista @FOLO3, ignorar este punto

Es difícil conseguir los OHLCV en tiempo real, en especial el volumen (yfinance da volumen en tiempo real, pero este no es un valor correcto y al cabo de 1-2 horas cambia, haciendo inviable usar yfinance para predicciones en tiempo real)

Para obtener volúmenes correctos en tiempo real, se realizan consultas a webull, por cada acción cada 2,5 minutos, se requiere un ID webull, están descargado en caché los por defecto @FOLO3 en `a_manage_stocks_dict.py.DICT_WEBULL_ID`

Pero si se desea usar acciones fuera de la lista @FOLO3

En `Utils/Volume_WeBull_get_tickers.py`

Cambiar la lista de ejemplo:

```
list_stocks = ["NEWS", "STOCKS", "WEBULL", "IDS"]
```

Por los ticker nasdaq, de los webull ID que se desea obtener.

Ejecutar `Utils/Volume_WeBull_get_tickers.py`

Una vez ejecutado se mostrará una lista en pantalla, esa debe ser añadida en `a_manage_stocks_dict.py.DICT_WEBULL_ID`

```
"MELI" : 913323000,  
"TWLO" : 913254000,
```

## 5.2 Configurar chatID y tokens en Telegram

Hay que obtener el token telegram y crear un canal

Se puede conseguir el token siguiendo el tutorial:

<https://www.siteguarding.com/en/how-to-get-telegram-bot-api-token>

Con el token actualizar la variable de `ztelegram_send_message_handle.py`

```
#Get from telegram  
TOKEN = "00000000xxxxxxx"
```

Una vez hecho obtenido el token se deben obtener los chatID de los usuarios y administrador

Los usuarios solo reciben alertas de venta compra y arranque, mientras que el administrador recibe la de los usuarios además de los posibles problemas.

Para obtener los chatID de cada usuario ejecutar `ztelegram_send_message_UpdateUser.py` y a continuación escribir cualquier mensaje al bot, aparece tanto en consola de ejecución como al usuario el chatID

```
[>>> BOT] Message Send on 2022-11-08 22:30:31  
Text: You "User nickname " send me:  
"Hello world"  
ChatId: "5058733760"  
From: Bot name  
Message ID: 915  
CHAT ID: 500000760  
-----
```

Recoger el `CHAT ID: 500000760`

Con los chatID de los usuarios deseados, añadirlos a la lista `LIST_PEOPLE_IDS_CHAT` en `ztelegram_send_message_handle.py`



### 5.3 Envío de alertas en tiempo real Telegram

Es posible ejecutarlo sin configurar telegram, en ese caso no se enviarán alertas en telegram, pero si se registraron los resultados en tiempo real en:

`d_result/prediction_real_time.csv`

Se informará en consola mediante:

`is_token_telegram_configurated()` - Results will be recorded in real time, but no alert will be sent on telegram. File:

`d_result/prediction_real_time.csv`

`is_token_telegram_configurated()` - There is no value for the telegram TOKEN, telegram is required to telegram one

El criterio de enviar alerta o no , está definido en el método

`ztelegram_send_message.will_send_alert()` .Si mas de la mitad de modelos tienen una puntuación mayor al 93% o los modelos TF tienen una puntuación mayor de 93%, se envía alerta a los usuarios consumidores

Ejecutar `predict_POOL_enque_Thread.py`

Esta clase hay 2 tipos de hilos

- Productor , pregunta constantemente por los datos OHLCV, una vez los obtienen los introduce en un cola
- Consumer (2 hilos corren simultáneamente) van sacando los datos OHLCV de la cola , calculan los parámetros técnicos, hacen la predicción de los modelos, las registran en `zTelegram_Registers.csv` , y si estas cumplen los requisitos se envían por telegram

### Posibles mejoras

Mejoras en los modelos predictivos, usando multidimensionales

Mejoras en los modelos predictivos TF usando tensores (múltiples matrices en el tiempo) y no matrices (mono temporales, diseño actual).

En la clase `Model_TF_definitions.ModelDefinition.py`

A través de ella se obtienen las configuraciones de los modelos, densidad, número de neuronas etc.

Existen dos métodos:

- `get_dicts_models_One_dimension()` se usa actualmente y genera configuraciones de modelos TF para matrices
- `get_dicts_models_multi_dimension()` no se encuentra en uso, está preparado para dar múltiples configuraciones de modelos que usen tensores

Existe el metodo `Utils.Utils_model_predict.df_to_df_multidimension_array(dataframe, BACHT_SIZE_LOOKBACK)` , el cual transforma df de 2 dimensiones [columnas , filas] a df de tres dimensiones [columnas , files, `BACHT_SIZE_LOOKBACK` ]

`BACHT_SIZE_LOOKBACK` significa cuantos registros en pasado se añaden al df, el número es configurable y valor por defecto es ocho.

Para comenzar el desarrollo debe ser llamar al método con `BACHT_SIZE_LOOKBACK` con un valor número entero, el método devolverá un df multidimensional [columnas , files, `BACHT_SIZE_LOOKBACK` ], con el que alimentar los modelos TF

```
Utils_model_predict.scaler_split_TF_onbalance(df,  
label_name=Y_TARGET, BACHT_SIZE_LOOKBACK=8)
```

**Mejora:** Una vez se devuelven esos arrays multidimensionales, se obtienen modelos con `get_dicts_models_multi_dimension()` no se logra hacer entrenar un modelo y hacer una predicción con array multidimensionales

Revisar la manera de obtención de ground true

Antes de entrenar los modelos los intervalos (de 15min) se clasifican en punto de compra 100 o 101, punto de venta -100 o -101 o punto de no operar 0, estos valores se introducen en la columna `Y_TARGET = 'buy_sell_point'` a través del método

```
Utils.Utils_buy_sell_points.get_buy_sell_points_Roll()
```

Se calcula la variación respecto a las siguientes 12 ventanas (15min \* 12 = 3 horas), y de ahí se obtiene los 8% puntos de mayor subida y mayor bajada , a esos puntos se les asigna valores distintos de 0

Para obtener el `Y_TARGET` existen 2 métodos que son los responsables de la estrategia a seguir una vez se compra-vende , encaso de perdida se optara por Stop Loss

```
rolling_get_sell_price_POS() y rolling_get_sell_price_NEG()
```

**Mejora opcional:** el sistema actual decide por porcentajes, es decir, los 16% mayores subidas y bajadas (8% cada uno) son el ground true. Es decir hay subidas o bajadas mayores del 3% que se pueden quedar fuera si la acción es muy volátil

Añadir el indicador de sentimientos noticias

Se obtienen las noticias de cada acción con

`news_get_data_NUTS.get_news_sentiment_data()` este método obtiene todas las noticias asociadas de: INVESTING.com YAHOO.com y FINVIZ.COM

( se usa la API investpy , la cual recientemente octubre 2022 ha empezado a fallar , probablemente debido al bloqueo de investing.com

<https://github.com/alvarobartt/investpy> )

Obtenidas esas noticias, el método

`news_sentiment_va_and_txtBlod.get_sentiment_predictorS()` procede a evaluar y puntuar de -100 negativa a 100 positiva, mediante 4 modelos. Resulta conveniente introducir más páginas de noticias

Los modelos están descargados de internet, bien vía modelos AI o librerías, se puede encontrar las referencias en:

```
news_sentiment_flair.get_sentiment_flair  
news_sentiment_t5.get_sentiment_t5  
news_sentiment_t5.get_sentiment_t5Be  
get_sentiment_textBlod
```

**Ejecutar** `news_get_data_NUTS.get_json_news_sentimet()`

Se genera un fichero .csv y.json , con fecha acción los cuatro modelos , la puntuación y la noticia recogida Ejemplo: `d_sentiment/stock_news_DATE_MELI.csv`

```
Date Ticker news_va news_fl news_t5 news_t5Be news_txtBlod Headline
2021-03-01 MELI 95.312 77.039 100.0 92.15 6.733 S&P 500 Earnings Increase As Yields Soar
The earnings per share (EPS) for all S&P 500 companies combined increased to $174.19 last week. The forward EPS
2019-09-05 MELI -82.026 -99.99 -100.0 -96.232 -12.56 MercadoLibre (MELI) Down 12% Since Last Earnings Report:
Can It Rebound? It has been about a month since the last earnings report for MercadoLibre (NASDAQ:MELI). Sha
```

**Mejora:** Obtenido el fichero de puntuación sentimientos-noticias introducirlo en los modelos predictivos junto con los indicadores técnicos, debe hacer en tiempo real.

Añadir balances económicos

Se puede añadir los balances económicos fácilmente usando la API de yahoo

<https://github.com/ranaroussi/yfinance>

```
# show financials
msft.financials
msft.quarterly_financials
```

Estos balances se actualizan cada trimestres.

Se pueden obtener las fechas de publicacion de resultados en API de yahoo

```
# show next event (earnings, etc)
msft.calendar
# show all earnings dates
msft.earnings_dates
```

Lista de mejoras sugeridas:

Permitir analizar acciones fuera del nasdaq, cambiar en :

```
yahoo_history_stock.__select_download_time_config()
Utils/API_alphavantage_get_old_history.py
```

Redirigir los restantes print() a `Logger.logr.debug()`

Traducir a través de <https://www.deepl.com/> los posibles mensajes restantes en español a ingles

Los esquemas generados en la carpeta `plots_relations/plot` por

Cambiar el funcionamiento del bot, que baste com mandar e comando `\start` , y retirar el caso de ejecucion de `ztelegram_send_message_UptateUser.py` descrito en el punto: 5.2

Enviar alerta en tiempo real mail

Revisar Stock prediction fail LSTM

LSTM time series + stock price prediction = FAI

<https://www.kaggle.com/code/carlmcbrideellis/lstm-time-series-stock-price-prediction-fail>

Encontrar la explicación de qué indicadores y valores toma el modelos AI, para predecir lo que predice y dar una pequeña explicación-esquema , por ejemplo los modelos random forest si se puede imprimir la secuencia que hace la predicción (verde compra , rojo no operar)

<https://stackoverflow.com/questions/40155128/plot-trees-for-a-random-forest-in-python-with-scikit-learn>



Nombres de indicadores:

Date	buy_sell_point	Open	High	Low	Close	Volume	per_Close	per_Volume	has_preMarket
	per_preMarket	olap_BBAND_UPPER	olap_BBAND_UPPER	olap_BBAND_MIDDLE	olap_BBAND_LOWER	olap_BBAND_UPPER_crash			
olap_BBAND_LOWER_crash		olap_BBAND_dif	olap_HT_TRENDLINE	olap_MIDPOINT		olap_MIDPRICE		olap_SAR	
olap_SAREXT	mtum_ADX	mtum_ADXR	mtum_APO	mtum_AROON_down	mtum_AROON_up	mtum_AROONOSC		mtum_BOP	
mtum_CCI	mtum_CMO	mtum_DX	mtum_MACD	mtum_MACD	signal	mtum_MACD_list	mtum_MACD_crash	mtum_MACD_ext	
mtum_MACD_ext_signal		mtum_MACD_ext_list	mtum_MACD_ext_crash	mtum_MACD_fix		mtum_MACD_fix_signal			
mtum_MACD_fix_list	mtum_MACD_fix_crash	mtum_MFI	mtum_MINUS_DI	mtum_MINUS_DM		mtum_MOM	mtum_PLUS_DI		
mtum_PLUS_DM	mtum_PPO	mtum_ROC	mtum_ROCP	mtum_ROCR	mtum_ROCR100	mtum_RSI	mtum_STOCH_k	mtum_STOCH_d	
	mtum_STOCH_kd	mtum_STOCH_crash	mtum_STOCH_Fa_k	mtum_STOCH_Fa_d		mtum_STOCH_Fa_kd			
mtum_STOCH_Fa_crash	mtum_STOCH_RSI_k	mtum_STOCH_RSI_d	mtum_STOCH_RSI_kd	mtum_STOCH_RSI_crash				mtum_TRIX	
mtum_ULTOSC	mtum_WILLIAMS_R	volu_ChaiKin_AD	volu_ChaiKin_ADOSC	volu_OBV		vola_ATR	vola_NATR	vola_TRANGE	
	cycl_DCPERIOD	cycl_DCPHASE	cycl_PHASOR_inph	cycl_PHASOR_quad		cycl_SINE_sine		cycl_SINE_lead	
	cycl_HT_TRENDMODE	cdl_2CROWS	cdl_3BLACKCROWS	cdl_3INSIDE		cdl_3LINESTRIKE		cdl_3OUTSIDE	
	cdl_3STARSINSOUTH	cdl_3WHITESOLDIERS	cdl_ABANDONEDBABY	cdl_ADVANCEBLOCK		cdl_BELTHOLD		cdl_BREAKAWAY	
	cdl_CLOSINGMARUBOZU	cdl_CONCEALBABYSWALL		cdl_COUNTERATTACK	cdl_DARKCLOUDCOVER	cdl_DOJI		cdl_DOJISTAR	
	cdl_DRAGONFLYDOJI	cdl_ENGULFING	cdl_EVENINGDOJISTAR	cdl_EVENINGSTAR		cdl_GAPSIDESIDEWHIWE			
cdl_GRAVESTONEDOJI	cdl_HAMMER		cdl_HANGINGMAN	cdl_HARAMI		cdl_HARAMICROSS		cdl_HIGHWAVE	
cdl_HIKKAKE	cdl_HIKKAKEMOD		cdl_HOMINGPIGEON	cdl_IDENTICAL3CROWS	cdl_INNECK		cdl_INVERTEDHAMMER		
cdl_KICKING	cdl_KICKINGBYLENGTH	cdl_LADDERBOTTOM	cdl_LONGLEGGEDDOJI	cdl_LONGLINE		cdl_MARUBOZU			
cdl_MATCHINGLOW	cdl_MATHOLD	cdl_MORNINGDOJISTAR	cdl_MORNINGSTAR	cdl_ONNECK		cdl_PIERCING			
cdl_RICKSHAWMAN	cdl_RISEFALL3METHODS		cdl_SEPARATINGLINES	cdl_SHOOTINGSTAR		cdl_SHORTLINE		cdl_SPINNINGTOP	
	cdl_STALLEDPATTERN	cdl_STICKSANDWICH	cdl_TAKURI	cdl_TASUKIGAP		cdl_THRUSTING		cdl_TRISTAR	
	cdl_UNIQUE3RIVER	cdl_UPSIDEGAP2CROWS	cdl_XSIDEGAP3METHODS		sti_BETA	sti_CORREL		sti_LINEARREG	
	sti_LINEARREG_ANGLE	sti_LINEARREG_INTERCEPT		sti_LINEARREG_SLOPE	sti_STDDEV		sti_TSF	sti_VAR	
ma_DEMA_5	ma_EMA_5	ma_KAMA_5	ma_SMA_5	ma_T3_5	ma_TEMA_5	ma_TRIMA_5	ma_WMA_5	ma_DEMA_10	ma_EMA_10
ma_KAMA_10		ma_SMA_10	ma_T3_10	ma_TEMA_10		ma_TRIMA_10	ma_WMA_10	ma_DEMA_20	ma_EMA_20
ma_KAMA_20		ma_SMA_20	ma_T3_20	ma_TEMA_20		ma_TRIMA_20	ma_WMA_20	ma_DEMA_50	ma_EMA_50
ma_KAMA_50		ma_SMA_50	ma_T3_50	ma_TEMA_50		ma_TRIMA_50	ma_WMA_50	ma_DEMA_100	ma_EMA_100
	ma_KAMA_100		ma_SMA_100	ma_T3_100	ma_TEMA_100		ma_TRIMA_100	ma_WMA_100	
trad_s3	trad_s2	trad_s1	trad_pp	trad_r1	trad_r2	trad_r3	clas_s3	clas_s2	clas_s1
clas_r3	clas_r2	fibo_s3	fibo_s2	fibo_s1	fibo_pp	fibo_r1	fibo_r2	fibo_r3	wood_s3
wood_pp	wood_r1	wood_r2	wood_r3	demark_s1	demark_pp	demark_r1	cama_s3	cama_s2	cama_s1
cama_r2	cama_r3	ti_acc_dist		ti_chaikin_10_3	ti_choppiness_14	ti_coppock_14	11_10	ti_donchian_lower_20	ti_donchian_center_20
	ti_donchian_center_20		ti_donchian_upper_20		ti_ease_of_movement_14	ti_force_index_13			
ti_hma_20	ti_kelt_20_lower	ti_kelt_20_upper		ti_mass_index_9_25	ti_supertrend_20		ti_vortex_pos_5	ti_vortex_neg_5	
	ti_vortex_pos_14	ti_vortex_neg_14		cycl_EBSW_40_10	mtum_AO_5_34		mtum_BIAS_SMA_26	mtum_AR_26	
	mtum_BR_26	mtum_CFO_9		mtum_CG_10	mtum_CTI_12		mtum_DMP_14	mtum_DMN_14	
	mtum_ER_10	mtum_BULLP_13		mtum_BEARP_13	mtum_FISHERT_9_1		mtum_FISHERTS_9_1		
mtum_INERTIA_20_14	mtum_K_9_3		mtum_D_9_3	mtum_J_9_3		mtum_PGO_14		mtum_PSL_12	
mtum_PVO_12_26_9	mtum_PVOH_12_26_9		mtum_PVOS_12_26_9	mtum_QQE_14_5_4236	RSIMA		mtum_QOEL_14_5_4236		
mtum_QQEs_14_5_4236	mtum_RSX_14		mtum_STC_10_12_26_05		mtum_STCmaed_10_12_26_05		mtum_STCstoch_10_12_26_05		
	mtum_SMI_5_20_5	mtum_SMIS_5_20_5	mtum_SMT_5_20_5	olap_ALMA_10_60_085		olap_HWMA_02_01_01		olap_YMA_7_0	
	olap_MCGD_10	olap_FWMA_10	olap_SINWMA_14	olap_SSF_10_2		olap_SWMA_10		olap_VMAP	
olap_VWMA_10	perf_CUMLOGRET_1	perf_CUMPCRET_1		perf_z_30_1		perf_ha	sti_ENTP_10	sti_KURT_30	
	sti_TOS_STDEVALL_LR	sti_TOS_STDEVALL_L_1		sti_TOS_STDEVALL_U_1		sti_TOS_STDEVALL_L_2			

tend_PSAR1_002_02	tend_PSARS_002_02	tend_PSARaf_002_02	tend_PSARr_002_02	tend_VHF_28	vola_HWM	vola_HWU
vola_HWL	vola_KCLe_20_2	vola_KCBe_20_2	vola_KCUe_20_2	vola_RVI_14	vola_THERMO_20_2_05	
vola_THERMoma_20_2_05	vola_THERMol_20_2_05	vola_THERMos_20_2_05	vola_THERMos_20_2_05	vola_TRUERANGE_1	vola_UI_14	
volu_EFI_13	volu_NVI_1	volu_PVI_1	volu_FVOL	volu_PVR	volu_PVT	mtum_murrey_math
mtum_td_seq	mtum_td_seq_sig	tend_hh	tend_hl	tend_l1	tend_lh	tend_hh_crash
tend_l1_crash	tend_lh_crash	ichi_tendkan_sen	ichi_kijun_sen	ichi_kijun_sen	ichi_senkou_a	ichi_senkou_b
ichi_isin_cloud	ichi_crash	ichi_chikou_span	tend_renko_TR	tend_renko_ATR	tend_renko_ATR	tend_renko_brick
tend_renko_change	pcrh_trad_s3	pcrh_trad_s2	pcrh_trad_s1	pcrh_trad_pp	pcrh_trad_pp	pcrh_trad_r1
pcrh_trad_r2	pcrh_trad_r3	pcrh_clas_s3	pcrh_clas_s2	pcrh_clas_s1	pcrh_clas_s1	pcrh_clas_pp
pcrh_clas_r1	pcrh_clas_r2	pcrh_clas_r3	pcrh_fibo_s3	pcrh_fibo_s2	pcrh_fibo_s1	pcrh_fibo_s1
pcrh_fibo_pp	pcrh_fibo_r1	pcrh_fibo_r2	pcrh_fibo_r3	pcrh_wood_s3	pcrh_wood_s2	pcrh_wood_s2
pcrh_wood_s1	pcrh_wood_pp	pcrh_wood_r1	pcrh_wood_r2	pcrh_wood_r3	pcrh_demark_s1	pcrh_demark_s1
pcrh_demark_pp	pcrh_demark_r1	pcrh_cama_s3	pcrh_cama_s2	pcrh_cama_s1	pcrh_cama_pp	pcrh_cama_pp
pcrh_cama_r1	pcrh_cama_r2	pcrh_cama_r3	pcrh_DEMA_5_DEMA_10	pcrh_DEMA_5_DEMA_10	pcrh_DEMA_5_DEMA_10	pcrh_DEMA_5_DEMA_10
pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_T3_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_TRIMA_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_WMA_10	pcrh_DEMA_5_WMA_10
pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_KAMA_20	pcrh_DEMA_5_SMA_20	pcrh_DEMA_5_T3_20	pcrh_DEMA_5_TEMA_20	pcrh_DEMA_5_TEMA_20
pcrh_DEMA_5_TRIMA_20	pcrh_DEMA_5_WMA_20	pcrh_DEMA_5_WMA_20	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50
pcrh_DEMA_5_SMA_50	pcrh_DEMA_5_T3_50	pcrh_DEMA_5_TEMA_50	pcrh_DEMA_5_TRIMA_50	pcrh_DEMA_5_WMA_50	pcrh_DEMA_5_WMA_50	pcrh_DEMA_5_WMA_50
pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_KAMA_100	pcrh_DEMA_5_SMA_100	pcrh_DEMA_5_T3_100	pcrh_DEMA_5_T3_100
pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_WMA_100	pcrh_DEMA_5_T3_h20	pcrh_DEMA_5_T3_h20	pcrh_DEMA_5_T3_h20
pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_KAMA_10	pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_T3_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_TRIMA_10	pcrh_DEMA_5_TRIMA_10
pcrh_DEMA_5_WMA_10	pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_KAMA_20	pcrh_DEMA_5_SMA_20	pcrh_DEMA_5_T3_20	pcrh_DEMA_5_T3_20
pcrh_DEMA_5_TEMA_20	pcrh_DEMA_5_TRIMA_20	pcrh_DEMA_5_WMA_20	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50
pcrh_DEMA_5_SMA_50	pcrh_DEMA_5_T3_50	pcrh_DEMA_5_TEMA_50	pcrh_DEMA_5_TRIMA_50	pcrh_DEMA_5_WMA_50	pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_DEMA_100
pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_KAMA_100	pcrh_DEMA_5_SMA_100	pcrh_DEMA_5_T3_100	pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_TRIMA_100	pcrh_DEMA_5_TRIMA_100
pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_WMA_100	pcrh_DEMA_5_T3_h20	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_KAMA_10	pcrh_DEMA_5_KAMA_10
pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_T3_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_TRIMA_10	pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_WMA_10	pcrh_DEMA_5_WMA_10
pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_KAMA_20	pcrh_DEMA_5_SMA_20	pcrh_DEMA_5_T3_20	pcrh_DEMA_5_TEMA_20	pcrh_DEMA_5_TEMA_20
pcrh_DEMA_5_TRIMA_20	pcrh_DEMA_5_WMA_20	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50
pcrh_DEMA_5_SMA_50	pcrh_DEMA_5_T3_50	pcrh_DEMA_5_TEMA_50	pcrh_DEMA_5_TRIMA_50	pcrh_DEMA_5_WMA_50	pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_DEMA_100
pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_KAMA_100	pcrh_DEMA_5_SMA_100	pcrh_DEMA_5_T3_100	pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_TRIMA_100	pcrh_DEMA_5_TRIMA_100
pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_WMA_100	pcrh_DEMA_5_T3_h20	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_KAMA_10	pcrh_DEMA_5_KAMA_10
pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_T3_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_TRIMA_10	pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_WMA_10	pcrh_DEMA_5_WMA_10
pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_DEMA_20	pcrh_DEMA_5_KAMA_20	pcrh_DEMA_5_SMA_20	pcrh_DEMA_5_T3_20	pcrh_DEMA_5_TEMA_20	pcrh_DEMA_5_TEMA_20
pcrh_DEMA_5_TRIMA_20	pcrh_DEMA_5_WMA_20	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_DEMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50	pcrh_DEMA_5_KAMA_50
pcrh_DEMA_5_SMA_50	pcrh_DEMA_5_T3_50	pcrh_DEMA_5_TEMA_50	pcrh_DEMA_5_TRIMA_50	pcrh_DEMA_5_WMA_50	pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_DEMA_100
pcrh_DEMA_5_DEMA_100	pcrh_DEMA_5_KAMA_100	pcrh_DEMA_5_SMA_100	pcrh_DEMA_5_T3_100	pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_TRIMA_100	pcrh_DEMA_5_TRIMA_100
pcrh_DEMA_5_TEMA_100	pcrh_DEMA_5_WMA_100	pcrh_DEMA_5_T3_h20	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_TEMA_KAMA_5	pcrh_DEMA_5_KAMA_10	pcrh_DEMA_5_KAMA_10
pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_T3_10	pcrh_DEMA_5_TEMA_10	pcrh_DEMA_5_TRIMA_10	pcrh_DEMA_5_SMA_10	pcrh_DEMA_5_WMA_10	pcrh_DEMA_5_WMA_10

