Curso - Trading Algorítmico con Python

Docente: Frank Ygnacio Rosas (MFM Candidate, School of Mathematics, University of Minnesota) | fsyr.quantmoon@gmail.com

Asistente de cátedra: Helí González (MSc Estadística, Universidad Nacional Agraria La Molina) | heligonzalespe@gmail.com

---------- Ficha del proyecto ----------

Fecha límite de entrega: *Domingo 17 de noviembre a las 23:59 hrs (GMT-5)*.

Audiencia: Edúcate Perú Consultores

----------Fin de Ficha----------

Octubre, 2024

Título de proyecto:

Desarrollo de Sistema Algorítmico ``algoTrading”

1) Contexto General

El presente proyecto tiene por fin el desarrollo y puesta en práctica del sistema algorítmico `algoTrading` trabajado para el curso. Este sistema algorítmico deberá incluir los procesos de:

- Lectura de archivos `.zarr`
- Construcción de barras y etiquetas
- Construcción de features
- Selección de features
- Combinatorial Backtest y almacenamiento de información
- Conexión a Cloud
- Live (Paper) Trading

Tal y como se trabajó en clase, estas etapas están basadas en el libro `Advances in Financial Machine Learning` (1st Ed., 2018) de Marcos López de Prado.

2) Criterio de Evaluación

El criterio de evaluación se basará en dos aspectos clave: **integridad** y **funcionalidad**. En ese sentido, se evaluará que el sistema algorítmico incluya todos los archivos proporcionados en clase, correctamente organizados y almacenados, así como nuevos archivos que el alumno desee incluir en sus sitema. Asimismo, se evaluará el cumplimiento del sistema algorítmico en el *live (paper) trading*. La puntuación máxima es de 20 puntos. Las especificaciones para cada criterio se detallarán en la sección siguiente.

3) Especificaciones (20 pts)

Cada uno de las especificaciones mencionadas formarán parte de la evaluación. En ese sentido, estas son:

3.1. Archivos (Puntaje: 8 pts)

El sistema algorítmico deberá presentarse en un folder llamado `algoTrading`. Este folder deberá contener, *como mínimo*, los siguientes archivos y folders:

1. Archivos `[nombre]Computation.py`: estos son archivos ejecutables por el usuario para una etapa en específica del sistema; por ejemplo, un archivo `barsComputation.py` permitiría ejecutar *únicamente* la computación de las barras y etiquetas.
2. Archivos `[nombre]Stage.py`: estos son archivos no-ejecutables, los cuales reúnen los procesos que serán llamados en los archivos `[nombre]Computation.py`; por ejemplo, el archivo `barsStage.py` contendrá todas los paquetes, funciones, y clases llamadas en el archivo `barsComputation.py` por el usuario para el cómputo de las barras y etiquetas.
3. Folders:
    A. `backtestResults`: folder con los `.csv` con los resultados del backtest. Asimismo, este folder deberá contener los archivos pickle (`.pkl`) del modelo endógeno (size) y exógeno (side).
    B. `bars`: folder con los `.csv` con los archivos conteniendo las barras y las etiquetas por acción por cada archivo `.zarr` utilizado en el sistema. Por ejemplo, si se utiliza `MCD US Equity.zarr`, debería almacenarse en esta carpeta un archivo `MCD US Equity.csv`.
    C. Folder `processedFeatures`. Este folder contendrá:
        a. Archivos `.csv` con los features generales por acción. Por ejemplo, `MCD US Equity_Features_Global.csv`.
        b. Archivos `.csv` con los features seleccionados por acción. Por ejemplo, `MCD US Equity_Features_Selected.csv`.
        c. Archivo `.pkl` del proceso de escalamiento.
        d. Archivos `.csv` con la información útil para el proceso de backtesting. Esto es:
            i. Archivo `aligned_tprices.csv` con los precios de la matriz stackeada (precios de `entry` y `exit`).

ii. Archivo `aligned_labels.csv` con las etiquetas de la matriz stackeada.
iii. Archivo `stacked_scaled_features.csv` con las columnas de features de la matriz stackeada.

Noten que en este caso pueden haber más de un archivo (es decir, diferentes insumos para el backtestings).

D. Folder `tickData` con los archivos `.zarr`.
E. Folder `tSystem` con los archivos `.py` adicionales para el funcionamiento del sistema.

Adicionalmente, para los fines del proyecto, el alumno deberá crear lo siguiente:

- **Archivo `mainTSystem.py` :** Este archivo será el único que el evaluador ejecutará para generar la interfaz de usuario. Deberá permitir al usuario llevar a cabo tres procesos principales, encapsulados en los archivos `[nombre]Computation.py` : la generación de barras y etiquetas, el cálculo y selección de *features*, y el ajuste de modelos junto con el *backtest*.

  Recomendación: Se sugiere al estudiante crear un archivo `baseTSystem.py` que contenga una clase llamada `TradingSystem`. Esta clase debería integrar los procesos mencionados anteriormente, utilizando diversos *inputs*. Luego, el archivo `mainTSystem.py` debería instanciar y utilizar esta clase `TradingSystem`.

  Interfaz de Usuario: El evaluador (usuario) deberá poder llamar al archivo `mainTSystem.py` desde la consola y ver una interfaz interactiva. Este le deberá permitir ingresar *inputs*, seleccionar opciones, y abrir o cerrar el sistema. *El alumno es libre de crear la interfaz bajo el estilo y funcionalidad que crea conveniente*. Un ejemplo visual de lo que es una interfaz desde la consola es el siguiente (en este caso, para un programa de simulaciones de Monte Carlo):

```
***>>> ------ FM5353 | Monte Carlo Simulator ------ <<<***
           >> Exotic Options Type Class <<
 Enter 'exit' at any time to close the program.
**-----------------------------------------------------**
> Note: If Van der Corput sequence is activated, Antithetic and Control Variate cannot be activated.

>>> User Parameters Input:::

Enter stock price: 100
Enter strike price: 120
Enter risk-free rate: 0.05
Enter volatility: 0.2
Enter time to maturity (in years): 1
Activate Van der Corput Sequence (1: yes, 0: no): 0
Activate Antithetic Variable (1: yes, 0: no): 1
Activate Control Variate (1: yes, 0: no): 1
Enter number of steps: 100
Enter number of simulations: 1000
>> COMPUTATION: Enable multithreaded execution (1: yes, 0: no): 1

Select Option Type:
(1) European Option
(2) Asian Option (Geometric)
(3) Digital Option
(4) Barrier Option
(5) Lookback Option
(6) Range Option
Enter option type (1-6): 2
Is it a Call option? (1: yes, 0: no): 1

>>> Processing the FM5353 MC Simulator....
```

- Finalmente, el alumno es libre de incluir cualquier otro archivo/folder adicional, especialmente los que se requieran para asegurar la conexión a cloud (de utilizarse).

3.2. Funcionalidad (Puntaje: 7 pts)

Esta parte de la evaluación se basa en la correcta funcionalidad del archivo `mainTSystem.py`.

En ese sentido, al ejecutar `mainTSystem.py` desde el terminal, el usuario (evaluador) no debe encontrar ningún problema (bug) que impida la ejecución del archivo. Asimismo, una vez terminada la ejecución, debe poder visualizar los archivos `.csv` y `.pkl` organizados adecuadamente en las carpetas especificadas en la sección **3.1. Archivos**. Además, el evaluador deberá poder abrir estos archivos manualmente y revisar que la estructura y los datos almacenados sean coherentes (por ejemplo, evitar ratios de Sharpe inusuales como 700 o valores de etiquetas poco razonables como 1.0015).

En caso de emplearse una conexión a la nube, el evaluador debe ser capaz de transferir la carpeta `algoTrading` a una máquina virtual y ejecutar la interfaz sin problemas. Por lo tanto, el estudiante tiene la libertad de modificar los archivos detallados en **3.1. Archivos** para asegurar una conexión óptima con la nube.

3.3. Live (Paper) Trading (Puntaje: 5 pts)

Finalmente, la evaluación de la conexión a *Live Trading* se realizará a través de un archivo en formato `.pdf` titulado `[nombre_alumno]liveTrading.pdf`. Este documento debe incluir una descripción detallada que integre visualizaciones tanto de la cuenta de *Live Trading* en Interactive Brokers como del terminal de operaciones. Noten que cada cuenta de paperTrading en Interactive Brokers tiene un ID asociado; este ID asociado deberá ser compartido en el archivo folder (por ejemplo, `U2513008`). Finalmente, el estudiante debe incluir gráficos (especialmente del *backtesting*) que evidencien la efectividad y precisión del algoritmo implementado. Este archivo `.pdf` debe tener, como mínimo 3 y como máximo 5 páginas de contenido (sin contar anexos y visualizaciones); las especificaciones a utilizarse son Times New Roman 12, interlineado 1pt.

En ese sentio, **el objetivo principal de este reporte es conocer el mejor modelo de *Machine Learning* obtenido por el estudiante**. El estudiante es completamente autónomo en utilizar el criterio de selección que desee, ya sea automatizado o manual. Asimismo, el reporte debe explicar **los resultados obtenidos durante el live trading**. Se recomienda a los estudiantes que, más que enfocarse en obtener retornos positivos, se centren en analizar y comprender los fenómenos subyacentes que ocurren en sus algoritmos.

Este archivo `.pdf` deberá incluirse en la carpeta `algoTrading`.

4) Indicaciones para la subida de archivos a GitHub para la evaluación del proyecto

Los estudiantes deberán crear una cuenta en GitHub y establecer un repositorio titulado `algoTrading`. Se recomienda que el repositorio sea privado. Además, deberán agregar como colaborador al docente, utilizando el correo `fsyr.quantmoon@gmail.com` (usuario: frankstack).

La fecha límite para la subida de archivos es el domingo 17 de noviembre a las 23:59 hrs. Recuerden que GitHub registra cada *commit* y *push* (subida de archivos) realizado, por lo que el colaborador podrá ver la hora de cada actualización de archivos.

Para saber más sobre github, pueden visitar:

- https://docs.github.com/es

5) Recomendación final

Finalmente, se recomienda el uso de una conexión a *Cloud* para optimizar tanto la velocidad de procesamiento como la profundidad de los cálculos. La ejecución en un entorno en la nube permitirá procesar el universo completo de más de 200 acciones proporcionadas, lo cual aporta una mayor riqueza de información. En entornos locales, es probable que solo se puedan procesar alrededor de 100 acciones o, en el mejor de los casos, todas, pero con un rendimiento mucho más lento.

En ese sentido, aunque *la conexión a la nube es opcional*, los resultados pueden ser inestables y poco confiables si el análisis se realiza solo sobre un universo reducido de menos de 50 acciones. Por lo tanto, en estos casos, **el uso de la nube es sumamente recomendado** para garantizar la estabilidad y la precisión de los resultados obtenidos.

Recuerden que pueden contactarse con el docente o el asistente de cátedra a sus correos respectivos!

Asimismo, **recuerden que el uso de AI's como ChatGPT o Copilot son bienvenidos!**

> ## Happy coding! :)