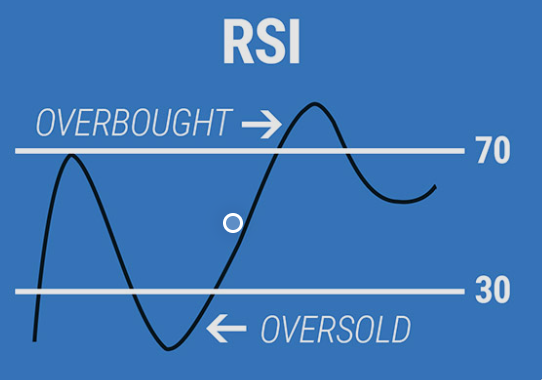
Como te he comentado anteriormente, me interesaba aprender Python para aplicarlo a las finanzas que es a lo que me dedico profesionalmente. Con lo que como proyecto me gustaría programar una estrategia que me diga cuando comprar y cuando vender una acción determinada.

Empecé con la estrategia mas sencilla que hay, que es el cruce de medias móviles pero los resultados eran bastante “poco rentables” con lo que probé con esta estrategia algo mas complicada y mi idea es poder probar con todas las que vaya considerando interesantes.

El objetivo de mi proyecto es conseguir automatizar una estrategia para que nos abra señales de compra o venta de unas acciones concretas con las que suelo trabajar según las “normas” del indicador técnico RSI .

La estrategia RSI, que es un indicador estocástico tipo oscilador puede darnos múltiples entradas tanto de compra como de venta seguidas.

Si el RSI es <30 o >70 entonces lo debemos programar para realizar una compra o una venta, es decir si cruza el valor de 30 abrirá una orden de compra de la acción y si cruza el 70 venderá la acción.



La idea es realizar entradas múltiples con un lote determinado y que únicamente se realicen mientras tenga capital disponible.

También deberemos tener en cuenta la comisión del bróker.

En el grafico tendremos:

1 – Las señales de compra venta.

2 – El Oscilador.

3 – El valor de la cartera o capital.

4 - La frecuencia de los retornos para ver como ha ido la estrategia.

El capital y la acción serán elegidos por el usuario en ventanas emergentes.

Las librerías para utilizar serán las siguientes:

Matplotlib para los gráficos.

Pandas por que vamos a trabajar con dataframes.

Numpy porque nos ayuda con algunas operaciones aritméticas.

Seaborn para los gráficos de retornos.

Yfinances para los datos económicos.

Utilizaré flask para hacerlo en una web local.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Lo primero que he querido hacer es la estrategia RSI, como he dicho antes podrá tener múltiples compras o ventas, y tendrá que tener una gestión del capital, para no quedarnos sin dinero a la hora de abrir operaciones.

También quiero que compruebe si hay capital antes de abrir la operación y ver si puede comprar mas que el lote mínimo y si no lo tuviéramos no compraría.

Creamos el formulario para pedir el capital y la acción, que sea mayor que 0.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Pondremos la fecha que queremos empezar el análisis:



Nos descargamos los datos:



Y obtenemos los datos de columna de cierre:



Ahora voy a calcular los retornos, en la variable “returns” la diferencia de porcentual en tanto por 1, pero me salto el primer dato.

Hago el calculo del RSI de 14 periodos (es decir los 14 precios anteriores).

En las variables up and down me quedo con los que son positivos y los que son negativos, le aplico el -1 por que se queden en valor absoluto.

Aplico la media exponencial para el numero de periodos que hemos puesto antes y ya calculo el RSI con su fórmula:

Texto

Descripción generada automáticamente

Creamos el Dataframe “signal” con el índice de las fechas del anterior, le paso en una columna la variable RSI y construyo las señales de compra y de venta. Lo hago con el modulo “where” de Numpy especifico que si el RSI esta por debajo de 30 esta en sobreventa le ponga un 1 y en caso contrario un 0.

En la columna positionbuy con la función diff es que si el RSI cruza el nivel de 30 delimita con un -1, que utilizaremos para señalizar esa operación de compra.

Hacemos lo mismo con signasell y position sell pero con 70.

Limpiamos el dataframe con iloc especificando que que excluya los periodos que indico.

Texto

Descripción generada automáticamente

Hago un print para ver que sea correcto:

Imagen de la pantalla de un celular

Descripción generada automáticamente con confianza media

Voy a construir manualmente un dataframe con ciertas columnas que me interesan

Imagen que contiene electrónica, circuito

Descripción generada automáticamente

Ahora especifico algunas variables, orden, seria el lote, para no poder comprar 0.5 acciones.

Un lote mínimo para no comprar menos de esas acciones, una comisión fija y otra comisión variable que tienen varios brokers, de 1 por cada mil. Si compramos 100000 serán 100 de variable.

En la variable parte, en tanto por uno el porcentaje de venta fijo. (50%).

Con la función dataloc vamos configurando el dataframe.

Texto

Descripción generada automáticamente

Creamos un bucle con la variable iterativa recorremos todas las filas y columnas.

Para cash, que me ponga el efectivo sustituyendo el 0, para el stock multiplico stock por precio y para PyG (perdidas y ganacias) quiero que el efectivo + portafolio – capital. Con X= 1 evito registros negativos.

Texto

Descripción generada automáticamente con confianza media

**Condición para las compras:**

Para la compra con if, le digo que si la señal de compra es =-1 y tenemos efectivo, teniendo en cuenta las comisiones la variable lote será = efectivo – compra con comisiones dividido para que sea entero y si el lote que nos queda es inferior a un lote mínimo lo ponemos en 0 para que no compre.

Texto

Descripción generada automáticamente

Dentro del if de las compras, si el efectivo es superior al lote \* precio de compra, actualizamos las columnas:

Imagen de la pantalla de un computador

Descripción generada automáticamente con confianza baja

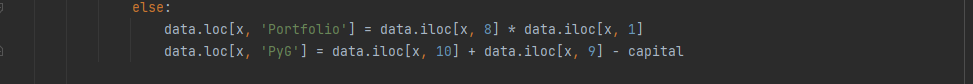
Con elif, si la señal de venta es -1 y hay stock, especificamos la variable “paquete” que en números enteros va a ser el stock por la ponderación en tanto por 1 del importe que se de cada vez que vendemos cuando se de cada una de estas señales. Si es inferior al lote mínimo no hará la venta.

Después configuramos el dataframe:

Texto

Descripción generada automáticamente

Finalizamos el bucle configurando cuando no haya ni compra ni venta.



Para finalizar el calculo la columna de fecha la pasamos a índice, generamos 2 nuevas columnas, total y neto y para los retornos calculamos con pct\_change, elimino el 0 ya que si esta parada la estrategia genera 0 y distorsiona la frecuencia.

Texto

Descripción generada automáticamente

Terminaríamos haciendo los gráficos:

Creo la tabla de 16x8 y pongo el titulo con el activo.

Guardo el grafico como imagen y lo mostramos en results:

Texto

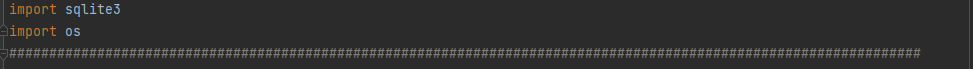
Descripción generada automáticamente

He implantado archivos CSS y HTML para hacer mas atractiva la aplicación. He buscado todo por internet (la verdad) y no ha sido complicada esta parte.

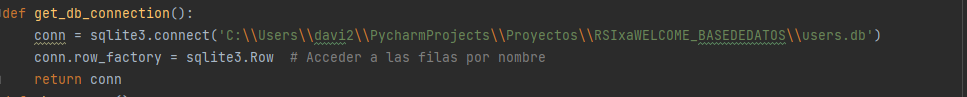
Lo que mas me ha costado ha sido hacer las funciones para calcular el RSI porque todo lo demás es bastante teórico y lo hemos visto en el curso.

No había puesto la base de datos (se me ha olvidado) así que he pensado pedir el nombre de la persona que se meta en la aplicación y guardarlo en una tabla para eso hago lo siguiente:

Importo el modulo sqlite3 y os para la ruta de la base de datos.



Defino la función get\_db\_connection para conectar la base de datos en la ruta indicada:



Creamos la función welcome:

Verificamos si es una solicitud POST y si es así obtiene el valor name y lo almacenamos en la variable “name”.

Nos conectamos a la base de datos.

Ejecutamos una consulta SQL para insertar el nombre que haya puyesto el usuario.

Guardamos los cambios y cerramos la conexión con la base de datos.

Después dirigimos al usuario a la página INDEX.

Texto

Descripción generada automáticamente

Al final también he creado una pagina de GRACIAS (thanks) y redirijo al usuario cuando quiera salir de la aplicación.

Envio el fichero comprimido en ZIP para que se copie directamente.

¡¡¡¡Muchas gracias!!!!