Trading algorítmico e inteligencia artificial en el Mercado de Capitales

Programa de Formación de la Bolsa de Comercio de Rosario 2020.

Autor: Patricio Iwanowski.

Tutor: Carlos Flury.

I. Resumen

El objetivo de este trabajo es explicar en qué consiste el trading algorítmico, las ventajas y riesgos que conlleva y las nociones generales de cómo se desarrolla un sistema automático para operar en el Mercado de Valores.

Comenzamos analizando cada uno de los elementos necesarios en cada sistema de algo-trading: el acceso y procesamiento de los datos del mercado, el análisis previo a operar, el modelo de armado del portafolio, el modelo de ejecución de operaciones, y, por último, el análisis posoperativo.

Una vez introducido el trading algorítmico, el trabajo explica de forma teórica los pasos a seguir para construir y validar dos estrategias simulando la aplicación de las mismas en los datos históricos de diversos activos financieros y analizando los resultados obtenidos. Luego, se trata el tema de optimización de parámetros de dichas estrategias, analizando nuevamente el impacto que esto tiene en los resultados.

Por último, se introducen conceptos de inteligencia artificial y se analizan las aplicaciones de estos en el trading algorítmico.

Comentado [FC1]: Ojo que esto debe haber quedado viejo (especificamente lo de la API y la empresa que ofrece algoritmos a sus clientes). Con tal de ajustarlo a lo que viene despues (las dos estrategias de momentum y rev a la media) esta OK

II. Índice

III. Introducción

El trading algorítmico, también llamado algo-trading, se refiere a cualquier uso de un programa para automatizar alguna o todas las partes del proceso de operar en cualquier mercado financiero. La ventaja de estos sistemas, es que operan a una velocidad y con una precisión muy superior a las que podría lograr cualquier ser humano.

Esto puede hacerse con distintos objetivos, los más comunes son:

- Generar rentabilidad y obtener ganancias.
- Ofrecer liquidez en un mercado.
- Disminuir el impacto en el precio de mercado de las operaciones propias si la posición que uno tuviera fuese lo suficientemente grande, como es el caso de los grandes fondos e instituciones financieras.

El trading algorítmico ha crecido rápidamente en los últimos años en todo tipo de instrumentos financieros, siendo responsable de más del 80% del volumen de mercado de acciones en Estados Unidos en la actualidad, y su potencial de aplicación es aún muy grande (Reuters and Bloomberg, 2011). La naturaleza competitiva del trading algorítmico, la escasez de expertos y el vasto potencial de rentabilidad hacen que los detalles de implementación sean difíciles de encontrar.

Los requisitos ideales para el trading algorítmico son:

- Un libro de órdenes centralizado que incluya todas las ofertas de compra y venta, con el precio y la cantidad de cada una de ellas.
- Mercado con mucha liquidez.
- Información de los protocolos financieros para la comunicación de información por computadora.

IV. Beneficios del trading algorítmico:

- Las operaciones se ejecutan a los mejores precios posibles.
- Las órdenes se colocan en el mercado instantáneamente y con precisión, por lo que hay una alta probabilidad de que la ejecución sea a los niveles deseados.
- Costos de transacción reducidos.
- Se revisan múltiples condiciones de mercado simultáneamente.
- Se reduce el riesgo de errores manuales al posicionar órdenes en el mercado.
- Se puede realizar un backtesting sobre el algoritmo utilizando datos históricos y en tiempo real para verificar la viabilidad de la estrategia.
- Se elimina la posibilidad de errores humanos basados en factores psicológicos y emocionales.

V. Requerimientos técnicos

Para implementar una estrategia es necesario construir un programa de computadora y realizarle un backtesting, esto es, una prueba del algoritmo en un periodo histórico del pasado para analizar el desempeño que este hubiera tenido, y las ganancias o pérdidas que hubiésemos obtenido. El desafío es convertir la estrategia en un proceso computarizado que tenga acceso a una cuenta en un bróker y pueda colocar órdenes en el mercado. Los requerimientos son:

- Conocimiento de programación computacional, programadores contratados o adquirir un software pre-armado.
- Conectividad a la red.
- Acceso a una o varias plataformas de trading.
- Acceso a datos del mercado.
- Habilidad e infraestructura para realizar el backtest del algoritmo.

Hay riesgos y desafíos inherentes al sistema, como lo son problemas de conectividad, demoras entre los tiempos de lanzamiento de orden y los tiempos de ejecución, y lo más importante, algoritmos imperfectos. Entre más complejo sea el algoritmo, más estricto y meticuloso debe ser el backtesting requerido antes de la implementación.

VI. Componentes de un sistema de trading algorítmico.

- ✓ **Acceso y limpieza**: de los datos económicos financieros y/o sociales que guiaran el proceso.

- ✓ **Análisis previo al trading**: en esta etapa pueden utilizarse diferentes técnicas de *data science* para analizar las propiedades de la información de los activos que nos permitirán identificar oportunidades en el mercado.

- ✓ **Generación de señales**: se identifican los activos que se acumularan en el portafolio basado en el análisis previo, determinando qué comprar, qué vender y en qué cantidad.

- ✓ **Ejecución de operaciones**: es la manera en que se ejecutarán y/o enviarán las órdenes de mercado para los activos seleccionados. Determina cómo se operará en el mercado.

- ✓ **Análisis posterior al trading**: analiza los resultados de la operación, por ejemplo, la diferencia entre el precio cuando se tomó la decisión de comprar o vender y el precio cuando esta efectivamente se ejecutó (Philip Treleaven, Michal Galas, y Vidhi Lalchand, 2013: pág. 3).

El **análisis previo**, a su vez, tiene 3 componentes principales, estos son:

- El **Modelo Alpha** es el modelo matemático diseñado para predecir el comportamiento futuro del instrumento financiero con el cual el sistema de trading algorítmico planea operar. Analiza información histórica y en tiempo real para identificar oportunidades potenciales de operar con rentabilidad. Pueden estar basados en teorías, donde el desarrollador elige una serie de hipótesis sobre el comportamiento de los activos y luego confirma sus hipótesis con un modelo o de manera empírica, donde utiliza un algoritmo que analiza los datos del activo e identifica patrones subyacentes en ellos, sin ninguna idea preconcebida sobre el comportamiento del mismo (Philip Treleaven, Michal Galas, and Vidhi Lalchand, 2013: pág. 4).

  Las 3 principales técnicas que se utilizan son:

  - Análisis fundamental: evalúa variables que pueden afectar el valor del activo, incluyendo factores macroeconómicos (como la economía en general y las condiciones de la industria) y específicos de la compañía (como reportes financieros).

  - Análisis técnico: se enfoca principalmente en el análisis de la tendencia del activo y en la identificación de la formación de patrones en los gráficos de precios y volumen (lo que se conoce como chartismo).

  - Análisis cuantitativo: aplica métricas computacionales basadas en la estadística, la física o el *machine-learning* para capturar, predecir, o explotar factores de la información financiera, económica y o social en el trading.

- El **Modelo de Riesgo**, el cual evalúa los niveles de exposición asociados con el instrumento financiero que estamos analizando para adquirir y aquellos que ya tenemos en nuestro portafolio (Philip Treleaven, Michal Galas, and Vidhi Lalchand, 2013: págs. 4 y 5). Se busca optimizar el ratio riesgo/beneficio. Hay dos enfoques para hacer esto:

  - Limitar el tamaño del riesgo: se limita la exposición aplicando penalidades o constantes a los diferentes activos, o bien, midiendo su volatilidad y su dispersión con medidas estadísticas como el desvío estándar de sus precios.

  - Limitar el tipo de riesgo: eliminar tipos enteros de exposición. Esto a su vez, puede hacerse con modelos teóricos o empíricos, como se mencionó anteriormente.

- El **Modelo de Costo de Transacción**, que calcula los costos potenciales asociados con la operación del instrumento financiero. Estos incluyen comisiones, deslizamiento de precios entre que se coloca la orden y esta se ejecuta, impacto de mercado, entre otros (Philip Treleaven, Michal Galas, and Vidhi Lalchand, 2013: pág. 5).

Modelo de construcción del portfolio

Los resultados del modelo Alpha, del modelo de gestión de riesgo y del modelo de costo de transacción se utilizan como entradas para el proceso de generación de señales de trading, en el cual se determina la cantidad de cada instrumento financiero a adquirir o mantener para formar una cartera óptima, es decir, que maximice los potenciales beneficios mientras limite el riesgo a valores tolerables y disminuye los costos transaccionales. Los modelos de construcción de portfolio pueden dividirse en dos categorías:

Basados en reglas: utilizan heurísticas definidas por el programador para determinar cómo acumular los instrumentos financieros. Esencialmente hay 4 clases de modelos:

- Igual ponderación o peso a cada posición: esencialmente consiste en tener la misma cantidad en el portafolio de cada activo seleccionado.
- Ponderación de igual riesgo: consiste en que cada activo contribuya en igual medida al riesgo del portafolio, es decir, se balancean las posiciones seleccionadas de acuerdo a la volatilidad de cada activo (en términos sencillos se compra más de ese activo si es menos volátil y menos si es más volátil).
- Ponderación basada en el Alpha (rendimiento) del activo: consiste en adquirir más de los activos que mejor Alpha tengan en un cierto periodo de tiempo, por ejemplo, un año.
- Ponderación por árbol de clasificación: consiste en definir las ponderaciones de cada activo según diferentes clasificaciones para cada tipo de activo en un árbol de decisión, por ejemplo, se le puede dar una ponderación a cada sector (industrial, tecnológico, energético, financiero, etc.) y así sucesivamente.

De optimización: utilizan un algoritmo con una función objetivo que itera hasta alcanzar la cartera que cumpla una cierta condición, por ejemplo, maximizar la tasa de retorno.

VII. Estrategias de trading algorítmico:

Se refiere a la naturaleza de todo el espectro de actividades que lleva a cabo un sistema de este tipo, desde la recolección de los datos hasta el análisis final, pasando por todas las etapas intermedias. Las más comunes son:

- ***Momentum***: siguen las tendencias de los movimientos de precios de mercado a través de medias móviles, canales, soportes y resistencias e indicadores técnicos como el índice de fuerza relativa RSI o el MACD (convergencia-divergencia de la media móvil exponencial). Son estrategias simples de implementar, un ejemplo popular es el uso de medias móviles simples o exponenciales de 50 y 200 días, con la regla de comprar un activo cuando la media de 50 cruza al alza a la de 200, y vender cuando el mismo cruce ocurre a la baja.

- **Reversión a la media**: se basa en el concepto de que los precios altos o bajos de un determinado activo son temporales y eventualmente deben volver a sus valores promedio. Buscan identificar y definir rangos de precio y preparar un

algoritmo para operar automáticamente cada vez que el precio rompa hacia adentro o hacia afuera de dicho rango. Un derivado muy popular de esto es el trading de pares o *pairs trading*, que busca operar en dos activos que históricamente han estado muy correlacionados en momentos en los que esta relación se distorsiona o diverge, asumiendo que este par va a volver a converger en el futuro.

- **Oportunidades de arbitraje**: comprar un activo en un mercado a un precio relativamente bajo y venderlo en otro mercado al mismo tiempo a un precio relativamente más alto que el anterior para obtener una ganancia libre de riesgo.

- **Basadas en modelos matemáticos**: se puede operar según algún modelo matemático probado. El ejemplo más famoso es el portafolio delta-neutral. Delta es una letra griega utilizada para calcular como se vería afectado nuestro portafolio ante un cambio en el precio de un activo de una unidad monetaria. La estrategia consiste en combinar posiciones múltiples de un activo y derivados financieros de ese activo como opciones o futuros para balancear el delta de nuestro portafolio y que así este no se vea afectado si el precio sube o baja, sino que se busca rentabilidad por aumentos o disminuciones en la volatilidad del precio.

VIII. Back-testing

Antes de implementarlo, es necesario someter la estrategia desarrollada a un minucioso proceso de prueba conocido como *backtesting*. Se aplica la estrategia a los datos históricos que determina el rendimiento que la estrategia hubiese tenido de haber sido implementada en el mercado en esas condiciones pasadas. Aunque el *backtesting* no garantiza el rendimiento futuro del algoritmo, nos ayuda a entender las vulnerabilidades de la estrategia desarrollada a través de una simulación con datos históricos o generados aleatoriamente, sin arriesgar capital operativo, lo cual nos permite aprender de la historia y corregir o compensar estas debilidades haciendo los ajustes correspondientes a él o los modelos donde estos se encuentren y volviendo a probar la nueva estrategia mediante otro *backtesting* para analizar los cambios que estos tuvieron en su desempeño. Es recomendable utilizar una gran cantidad de datos históricos para este proceso, de manera tal que el mercado atraviese distintos ciclos y condiciones diferentes a lo largo del tiempo y pueda evaluarse la versatilidad del sistema para adaptarse a estos cambios, como pueden ser cambios de tendencia, cambios en la volatilidad, cambios en las condiciones del sector del activo financiero, etc. Por ejemplo, para evaluar un algoritmo que opere el índice S&P 500, podríamos usar los últimos 10 años de cotizaciones para el *backtesting*. Debe observarse que estos datos sean de igual temporalidad que la estrategia que se pretende, si se va a correr el algoritmo cada día, para buscar y generar señales para el día siguiente, se deben utilizar las cotizaciones diarias.

En el *backtesting* pueden compararse varias estrategias distintas con diferentes parámetros cada una. Naturalmente la primera idea que resulta lógica seria elegir aquella con mejores rendimientos. Esta idea rápidamente es descartada al tener en cuenta el riesgo, por supuesto que lo que buscamos es maximizar los retornos de nuestra inversión, pero lo óptimo es hacerlo minimizando nuestra exposición al riesgo,

especialmente al tener en cuenta que nada nos garantiza que una estrategia que tuvo buenos resultados en el pasado continuara haciéndolo en el futuro. Para un correcto análisis, se deben tener en cuenta las siguientes medidas:

- ***Drawdown***: también conocido como retroceso de la curva de resultados, el *drawdown* mide el retroceso actual en la curva de resultados respecto al anterior máximo en dicha curva y es una forma de medir el riesgo de nuestro sistema de trading (ya sea automático o no). Ningún sistema es perfecto y tarde o temprano atraviesa por una racha de pérdidas. Como se puede intuir, el análisis del *drawdown* (tanto en profundidad como en tiempo) es sumamente importante a la hora de evaluar si un determinado sistema de trading es susceptible de ser operado o no, puesto que uno de los principales errores que cometen muchos inversores es preocuparse únicamente de lo que el sistema gana históricamente, sin tener en cuenta lo que ha tenido que sufrir el sistema para poder obtener esos resultados finales (Michael L. Halls-Moore, 2015).

- ***Sharpe-Ratio***: es una medida del exceso de rendimiento por unidad de riesgo de una inversión. Si definimos Rt como los retornos de cada día t, es decir, los cambios relativos porcentuales de nuestra cuenta, entonces el *Sharpe-Ratio* está dado por la media de estos retornos (el promedio de todos los Rt), sobre el desvió estándar de los mismos

  Sharpe-Ratio = media (Rt) / Desvió estándar (Rt)

  Es decir, se trata de nuestra ganancia promedio sobre una medida de nuestro riesgo y la volatilidad a la cual estamos expuestos. Naturalmente, es deseable que esta ratio sea lo más alto posible y de estrategias similares elegiremos aquel con mayor *Sharpe-Ratio* para probar fuera de la muestra. También resulta muy útil para decidirnos por la implementación o el descarte de nuestra estrategia comparar esta medida contra un *benchmark*, por ejemplo, posicionarnos en long en un índice conocido como el S & P y *holdear* (mantener) la posición durante todo el periodo, o los bonos del tesoro de EEUU que se consideran inversión libre de riesgo, para ver cuánto más conveniente resulta nuestra estrategia diseñada respecto a otras más sencillas, comunes y/o conservadoras.

- **Simulación de Montecarlo**: el sistema se diseña con datos del pasado (porque no tenemos otros datos disponibles, y el pasado es lo único que conocemos), y el problema subyacente de esta práctica es que es poco probable que los datos futuros sean idénticos a los datos del pasado que utilizamos para desarrollar el sistema.

  La simulación de Montecarlo se puede utilizar para crear múltiples secuencias aleatorias de datos, en este caso, precios con los cuales operaria nuestro algoritmo. Todas esas secuencias alternativas son igual de probables y como resultado nos dan múltiples curvas de capital también todas igualmente probables. A partir de estas curvas de capital podemos estimar la probabilidad de obtener determinados rangos de beneficios, de *drawdown*, *sharpe-ratio* y de otros tipos de ratios estadísticos, lo cual nos permite sobretodo analizar el riesgo y poder gestionarlo mejor.

Ventajas de la simulación de Montecarlo:

- Analizar el riesgo: Puedo saber cuáles son los niveles de *drawdown* más probables.
- Determinar el tamaño de la posición que hará que nuestra curva de capital crezca más mientras se limita el *drawdown* a un nivel aceptable.
- Ayuda a estimar cuándo un sistema ha dejado de funcionar.
- Conocer las características del sistema: Cuanto más conocemos como se comporta el sistema, más confianza podremos tener en él porque sabremos qué es lo que podemos esperar de su rendimiento.

Desventajas:

- Si la muestra con la que trabajamos no es representativa de poco vale que podamos aleatorizar la secuencia, no nos dará resultados fiables.
- Montecarlo asume la independencia entre los datos, por lo que no gestiona correctamente los sistemas donde existe una alta correlación en los inputs. En casos de una marcada tendencia en la curva de precios de un activo, los precios están fuertemente correlacionados.

IX. Sesgos en el *back-testing*.

Desafortunadamente, el *backtesting* está afectado por sesgos que, si bien pueden disminuirse, jamás pueden eliminarse del todo. Lo que es aún peor, estos sesgos suelen tender a mejorar los resultados del rendimiento del sistema mostrados por el *back-testing* en lugar de disminuirlos. Teniendo esto en cuenta, el *back-testing* debe tomarse como una cota superior aproximada del rendimiento que puede esperarse y el riesgo que estamos corriendo, esto es, un escenario optimista.

Los 4 principales sesgos que se presentan son: el sesgo de sobre optimización, el de mirar hacia el futuro, el de supervivencia y, por último, el sesgo cognitivo.

Sesgo de optimización: es probablemente el más malicioso de todos. Se trata de utilizar demasiados parámetros para definir la estrategia de entrada y salida de una manera que resulte muy atractiva y rentable en los datos de entrenamiento de la estrategia. Sin embargo, como bien sabemos las condiciones pasadas no se mantienen contantes indefinidamente, sino que tienden a cambiar, por lo que el rendimiento con datos reales, inciertos y aleatorios probablemente arroje resultados mucho peores a los del backtesting (Michael L. Halls-Moore, 2015). Una forma de disminuir esto, es minimizar en lo posible la utilización de parámetros distintos en nuestra estrategia, dividir los datos históricos en datos de entrenamiento y datos de prueba fuera de la muestra, verificando de esta manera los resultados con datos que a priori eran desconocidos y comparar para saber si hubo demasiado sobreajuste y un pobre rendimiento fuera de la muestra (Rogelio A. A. Morán, 2017). Otra posibilidad es utilizar simulación de Montecarlo, para verificar nuestra estrategia contra corridas de datos realmente aleatorios.

Sesgo por mirar hacia el futuro: como su nombre lo indica, este sesgo es introducido en nuestro algoritmo cuando introducimos en el backtesting del mismo (que utiliza

información histórica) datos que se supone que aun serian inciertos si el algoritmo estuviese operando en tiempo real. Es decir, ocurre si llegamos al punto en el tiempo N, e incluimos datos de algún periodo N+k, con k mayor a cero (Michael L. Halls-Moore, 2015). Este sesgo puede introducirse de maneras muy sutiles, como, por ejemplo, por errores en los índices de los *arrays* que utiliza nuestro programa, por utilizar datos futuros para calcular algún parámetro operacional del sistema (como por ejemplo optimizar una regresión lineal entre dos series de datos completas y luego probar el desempeño con estos datos futuros que no deberían estar incluidos), etc. Lo apropiado seria utilizar los datos con uno o varios periodos de retraso para asegurarnos de eliminar este sesgo.

Sesgo de los supervivientes: este sesgo se introduce al tener en cuenta para aplicar nuestra estrategia solo aquellos activos financieros que han sobrevivido a lo largo del tiempo. Por ejemplo, si tuviéramos en cuenta solo las acciones de empresas que no desaparecieron en la crisis financiera del 2008, estaríamos introduciendo este sesgo por tener en cuenta solo empresas que el tiempo nos demostró que se desempeñaron de una mejor manera que otra, información que no hubiésemos tenido al ejecutar nuestro algoritmo en tiempo real en ese periodo, y probablemente hubiésemos sufrido pérdidas en ese periodo operando con esos activos que el backtesting no refleja (Michael L. Halls-Moore, 2015). Es un caso particular del sesgo por mirar hacia el futuro, ya que, indirectamente, estamos introduciendo información del futuro en nuestro back testing. Se pueden conseguir bases de datos libres de este sesgo para probar nuestras estrategias, o también puede utilizarse un periodo de tiempo menor que no incluya estos periodos de gran *drawdown* del mercado donde solo algunos activos sobrevivieron.

Sesgo cognitivo: se trata de factores psicológicos que afectaran los resultados de nuestra implementación si no se tienen en cuenta, por ejemplo, soportar el *drawdown* sin desconectar el algoritmo. Si el back-test nos arroja como resultados que la estrategia es ganadora, pero que podemos esperar un *drawdown* del 25%, y luego en la vida real al ver un *drawdown* de esta magnitud nos asustamos y no dejamos operar al algoritmo, podríamos estar arruinando una estrategia que de otro modo probablemente hubiera sido ganadora (Michael L. Halls-Moore, 2015). Es necesario interpretar correctamente los resultados del back-testing para saber que esperar, y como debemos comportarnos mientras nuestro algoritmo se ejecuta a lo largo del tiempo.

X. Construcción y comparación de estrategias Alpha

A continuación, procederemos a crear 2 sistemas básicos de algo-trading, basados en diferentes estrategias y comparando los resultados obtenidos en cada uno. Las estrategias utilizadas serán de tipo *Momentum* y otra de reversión a la media.

- ## Estrategia de Momentum:

Utilizaremos solo un indicador en esta estrategia, la media móvil (MM). Procederemos a explicar los fundamentos y utilización de la misma.

La media móvil se define como “cálculo para analizar un conjunto de datos para crear series de promedio”. Por lo tanto, una media móvil es una lista en la que cada número es un promedio de datos. Este promedio puede ser sencillamente dividir la suma de los precios que se consideren sobre la cantidad de precios que la conforman, dando lugar

a una media móvil simple, o bien pueden darse diferentes valoraciones o ponderaciones a cada precio o valor. Podríamos decir entonces que lo que hace la media móvil es suavizar el ruido de las variaciones del precio de mercado de un activo, para poder comprender mejor la tendencia detrás de estas variaciones.

Por lo tanto, podemos utilizar una media larga de n sesiones y una media corta de m sesiones para tratar de anticiparnos a un cambio de tendencia. La media larga va a tener una adaptación lenta, es decir, va a ser más difícil observar cambios en ella ya que tiene en cuenta un conjunto mayor de datos. Todo lo contrario que la media móvil de período más corto, la cual tiene en cuenta un menor número de datos y por lo tanto varía con mayor rapidez ante cambios en el precio. Teniendo esto en cuenta, podríamos decir que una es una media “lenta” y la otra una media “rápida” (Rogelio A. A. Morán, 2017).

Una estrategia de trading muy efectiva que utiliza esto es conocida como Cruce de Medias Móviles, que consiste en emplear la combinación de dos MMs, una para corto plazo y otra para largo plazo. En esta estrategia, se crea una señal cuando la media móvil más corta cruza a la media móvil más larga de abajo hacia arriba, lo cual generaría una señal de compra. Mientras que, si la media móvil más larga cruza a la media móvil más corta de arriba hacia abajo, se genera una señal opuesta que nos indica que debemos salir del mercado, es decir, vender nuestra posición.

La gran ventaja de este tipo de herramientas es que son simples y sencillas de implementar y verificar su desempeño incluso manualmente, ya que generan pocas señales de trading incluso en largos periodos de observación (Michael L. Halls-Moore, 2015).

Se creó una estrategia algorítmica como la mencionada anteriormente utilizando las medias móviles de 100 días y 400 días en lenguaje Python, cuya regla naturalmente es comprar el activo cuando la media de 100 cruza al alza la media de 400 y vender cuando ocurre lo opuesto. El capital inicial simulado es de 100000 dólares.

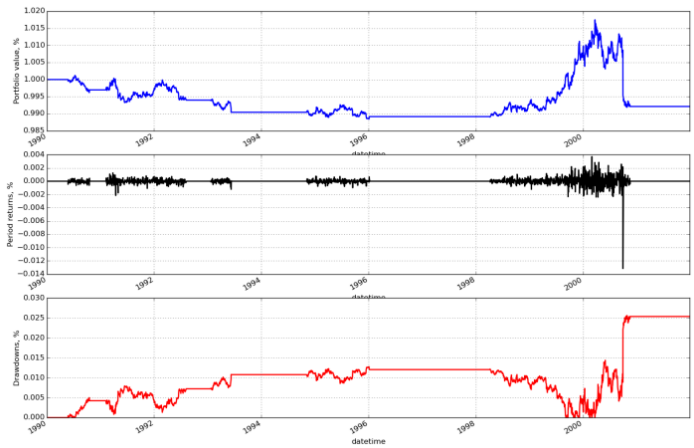
Para tener en cuenta el modelo de costo de transacción en la estrategia, todas las ordenes se lanzaron a precio de mercado, los cual afectara nuestro rendimiento por el costo de *slipagge* (diferencia de precio en el mercado entre que se genera la señal y se lanza y ejecuta nuestra orden) y las comisiones de Interactive Brokers, lo cual nos da un resultado verosímil si la estrategia se aplica con cuentas relativamente pequeñas, que no tenga impacto de mercado. Si tuviésemos impacto de mercado como en el caso,

por ejemplo, de un gran fondo de inversión, el modelo de costo de transacción se torna mucho más complejo.

Los resultados que arroja la aplicación de esa estrategia en las acciones de Apple en el periodo de tiempo desde el 1 de enero de 1990 hasta el 1 de enero de 2002 son los siguientes:

Capital final= 99211 dólares
Comisiones pagadas= 13 dólares
Retorno total = -0.79%
Sharpe Ratio = -0.09
Drawdown máximo = 2.56%
Duración del *drawdown* = 2312 periodos (de un día cada uno)
Señales generadas = 10

A continuación, se muestra un gráfico de dichas medidas a lo largo de este periodo:



Evidentemente, los resultados obtenidos para esta estrategia fueron muy malos y la misma no es apta para aplicarse en la realidad. Esto nos demuestra los peligros de operar en el mercado con estrategias demasiado sencillas, incluso en acciones de gran reputación y tendencia alcista como lo es Apple, e incluso en un largo horizonte de inversión.

Naturalmente, podemos hacernos varias preguntas a partir de esta prueba, la primera de ellas: ¿podría esta estrategia haber arrojado resultados distintos si se seleccionaran medias móviles con distinta cantidad de periodos considerados en cada una? La respuesta es que si, ya que podríamos optimizar los parámetros de esta y de cualquier otra estrategia de trading algorítmico para que den mejores resultados en un activo en particular, aunque corriendo el riesgo de sobre optimización que mencionamos anteriormente. El tema de la optimización es tratado más adelante en este trabajo.

Luego, podemos preguntarnos también, ¿qué pasaría si este mismo par de medias móviles se aplicase a otro activo distinto que tuviese más Momentum en sus tendencias alcistas y nos permitiese capturar mayores beneficios? Obviamente las estrategias de Momentum dan mejores resultados en unos activos que en otros, lo ideal, seria aplicar este tipo de estrategias a acciones que muestren un fuerte comportamiento tendencial, es decir, movimientos marcadamente sostenidos en una dirección durante un periodo relativamente largo de tiempo. Un coeficiente estadístico que puede resultarnos de utilidad para esto es el exponente de Hurst, el cual se utiliza para describir la "memoria de largo plazo" entre las observaciones de una serie de tiempo, es decir, que los eventos de un periodo influyen en todos los siguientes (Oscar Yesid Quintero Delgado y Jonathan Ruiz Delgado, 2011). Cuando aplicamos la estimación de Hurst a una serie de precios (lo cual puede hacerse en Python con una función que está presente en la librería pandas), podemos llegar a las siguientes valoraciones:

- Hurst = 0.5 ⇒ La serie es aleatoria
- Hurst < 0.5 ⇒ La serie tiene una dinámica de reversión a la media
- Hurst > 0.5 ⇒ La serie es tendencial.

Así también podremos conocer cuál es la intensidad de la dinámica: Si la estimación de Hurst tiene un valor cercano a 1 estaremos frente a una serie con un comportamiento fuertemente tendencial. Por el contrario, si Hurst es cercano a cero tendremos mucha probabilidad de encontrarnos frente a serie que tiende a revertir a la media (Oscar Yesid Quintero Delgado y Jonathan Ruiz Delgado, 2011). Sería interesante entonces, ver como resulta esta estrategia al aplicarse a un activo cuyas series de tiempo para un cierto número de periodos arrojen un exponente de Hurst cercano a 1. Esta es una posible mejora al sistema de trading propuesto, se deja como ejercicio para el lector interesado realizar la prueba y ver como resulta, así como también pensar en otras posibles mejoras para una estrategia de trading algorítmico basada en Momentum.

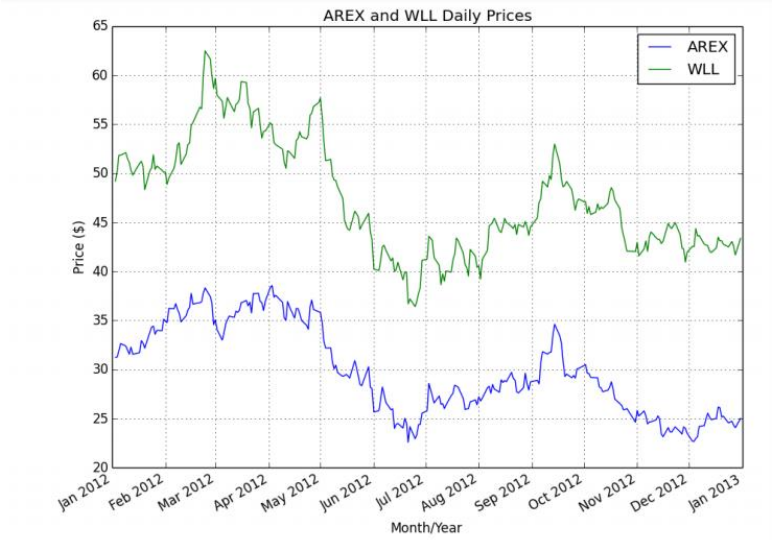
Queda claro que tenemos trabajo por hacer para mejorar estos resultados en la siguiente estrategia.

- ## Reversión a la media:

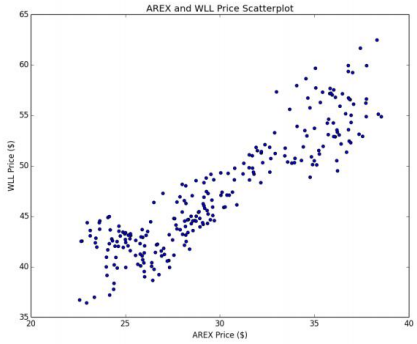
La siguiente estrategia consiste en buscar dos activos financieros que estén correlacionados, y posicionarnos en este par cuando esta correlación se rompa, asumiendo que en el futuro estos activos volverán a sus valores medios de correlación. Dicho en términos prácticos, supongamos que tenemos 2 activos que suelen desempeñarse de manera muy similar, pero en el último tiempo uno de ellos ha experimentado una suba de precio mientras el otro ha experimentado una baja, lo que haremos será tomar una posición en corto en el primero (apostando a la baja del precio) y una posición en largo en el segundo (apostar a la suba). Claramente, si se cumple nuestra hipótesis inicial, los precios de estos activos volverán a su valor medio por la correlación que se observaba entre ellos y obtendremos una ganancia.

Esta estrategia no puede aplicarse a cualquier par de activos, es necesario primero estudiar estadísticamente el comportamiento de las series de precios de los mismos. A simple vista pueden verse los gráficos de precio de dos activos financieros para identificar candidatos potenciales para este tipo de estrategias. A continuación, se

muestra el grafico de precios de las acciones de WILL y AREX, ambas de los mismos sectores energéticos (y por lo tanto expuestos a condiciones de mercado similares), desde enero del 2012 a enero del 2013, donde puede observarse que ambas series de precio presentan un comportamiento similar:



Si colocamos los precios de ambas en un gráfico de dispersión, puede observarse lo que parece ser una relación lineal entre ambos:



Esta relación lineal entre variables puede aproximarse mediante un método de machine learning de aprendizaje supervisado llamado regresión lineal. Esta aproximación tiene la siguiente forma:

$y(t) = \beta x(t) + E(t)$

Donde x(t) e y(t) son los precios de dichas acciones en un determinado momento t, β es el coeficiente que expresa la relación entre ambos y E(t) es el error que se puede

observar en cada una de estas predicciones en cada momento, también llamado residuo. Si la relación que planteamos es correcta, la serie de estos residuos en el tiempo debería ser un proceso aleatorio puro, esto es, una sucesión de variables aleatorias no correlacionadas, con media cero y varianza constante (Rogelio A. A. Morán, 2017). Se lo llama ruido blanco. Podemos utilizar este concepto para evaluar el modelo que hemos planteado ya que "si un método o modelo representa exactamente los patrones de la serie, es decir, toda la dinámica sistemática de la misma, la diferencia entre los valores de la serie y el pronóstico, esto es, el error de pronóstico, debe ser ruido blanco" (Rogelio A. A. Morán, 2017).

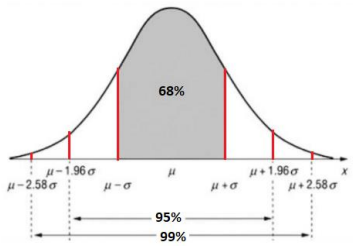
Se puede estimar el parámetro β que mejor ajuste nuestra regresión lineal a los datos observados a través del método de mínimos cuadrados ordinarios o MCO, el cual nos brindará la recta de regresión que nos proporcione un valor lo menor posible de la suma de los cuadrados de los residuos, esto es, la recta que mejor ajuste a los datos observados (Rogelio A. A. Morán, 2017). Se obviará el paso a paso de esta técnica ya que escapa a los objetivos del proyecto y puede hacerse ejecutando una simple función de una librería importable en Python llamada pandas. El parámetro de entrada para aplicar en Python el método MCO es el número de periodos del pasado a considerar para crear la regresión. Inicialmente tomaremos este número como 100, aunque veremos más adelante que este parámetro se puede optimizar.

Vamos a definir nuestras entradas al mercado en largo o en corto para el par seleccionado en función del análisis de los residuos. Cuando el residuo obtenido se aleja significativamente de cero hacia abajo (un valor negativo) vamos a ingresar al par en largo (nos posicionamos en largo en el activo representado por y(t) a la vez que nos posicionamos en corto en el activo representado por x(t)), saliendo de ambas posiciones cuando el residuo deje de ser significativamente lejano de cero porque entonces ya nos estaríamos acercando nuevamente a la situación normal. Dicho en términos más sencillos, los precios de estos activos se han alejado de su diferencia normal hacia abajo, entonces nos posicionamos en el mercado apostando a que dicha diferencia se acerque nuevamente a sus valores normales. Por ejemplo, supongamos que la diferencia de precio entre AREX y WILL suele ser de 10 dólares, pero en el momento vemos que AREX vale 50 dólares y WILL vale 100 dólares, es decir, vemos una diferencia de 50 dólares, lo que haremos es comprar AREX y *shortear* WILL apostando a que esta diferencia se achique y vuelva a valores más normales.

Lo opuesto también es válido, cuando el residuo sea significativamente distinto de cero en un valor positivo, tomaremos una posición en corto en el par (nos posicionamos en corto en el activo representado por y(t) a la vez que nos posicionamos en largo en el activo representado por x(t)), saliendo de ambas posiciones en el momento que el residuo se acerque de nuevo a cero en una medida que consideremos suficiente. Dicho en términos más sencillos, los precios de estos activos se han alejado de su diferencia normal hacia arriba, entonces nos posicionamos en el mercado apostando a que dicha diferencia se acerque nuevamente a sus valores normales.

Para ello utilizaremos el estadístico z, el cual mide la diferencia entre un estadístico observado y su parámetro hipotético de población en unidades de la desviación estándar. La conversión de una observación a un valor z se denomina estandarización. Para estandarizar una observación de una población, reste la media de población a la observación de interés y divida el resultado entre la desviación estándar de la población. El resultado de estos cálculos es el valor z asociado con la observación de interés.

Puede utilizar el valor z para determinar si puede rechazar la hipótesis nula. Para determinar si puede rechazar la hipótesis nula, compare el valor z con su valor crítico, que se puede encontrar en una tabla normal estándar en la mayoría de los libros de estadística. En este caso, vamos a calcular en cada observación de los precios de nuestro par el estadístico z del residuo que se genera y en función de la probabilidad que había de observar ese valor de z, podremos tomar posición en el mercado si observamos un valor muy grande de z positivo o negativo, dicho de otro modo, en los casos en que acabamos de ver un valor muy anormal de z y la probabilidad nos indica que el próximo valor puede ser más cercano al valor medio, es decir, nos alejamos demasiado de la media y estamos en el umbral de presenciar un fenómeno de reversión a la media.



El valor z para una serie de residuos se calcula con la siguiente formula:

Z = (E(t) – media(E(t))) / desv.est(E(t))

Esto es se le sustrae al residuo el valor medio de la serie de residuos, y se lo divide por la desviación estándar de la misma serie.

Entonces, definiremos dos valores de z, uno más grande que nos servirá para entrar al mercado cuando observemos un residuo que lo exceda en valor absoluto, y otro menor que nos servirá para salir de la posición del mercado que hemos tomado. Primero tomaremos estos dos valores de z de manera arbitraria, pero más adelante veremos que estos 2 también son parámetros que se pueden optimizar. Para esta prueba tomaremos zmayor=3 y zmenor=0.5. Cuando el ultimo z observado sea mayor a 3 (en valor absoluto, es decir, mayor a 3 o menor a -3) entraremos al mercado del par ya sea en corto o en largo dependiendo de si el valor z es positivo o negativo respectivamente.

Resumiendo lo expuesto anteriormente, para esta estrategia construiremos una regresión lineal utilizando el método OLS con los últimos 100 periodos de tiempo, y se capturaran los residuos de la diferencia entre el valor de esta regresión y el verdaderamente observado en el mercado, para calcular el valor z de cada residuo observado, y en función de este posicionarnos en el par de acciones en el mercado. Para esta estrategia las acciones consideradas en el par son AREX y WILL y los periodos utilizados serán los minutos (no días como en la estrategia anterior). Se cargan los datos del par en la estrategia a partir del 8 de noviembre del 2007 a las 10:41:00hs (es importante que ambas series de tiempo empiecen exactamente en el mismo momento) hasta el 11 de marzo del 2014 a las 16:01:00hs. El capital inicial simulado es nuevamente de 100000 dólares. Para el modelo de costo de transacción se toman las mismas consideraciones que en la estrategia anterior, es decir, se tiene en cuenta el

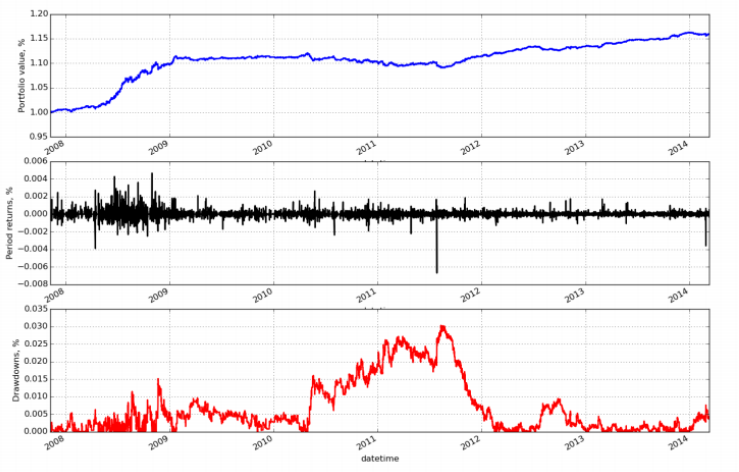
slippage por lanzar órdenes de mercado y se incluyen las comisiones pagadas, no así el impacto de mercado ya que suponemos que nuestra cuenta no lo tiene o este es mínimo. A continuación, se muestran los resultados del backtest de dicha estrategia:

Capital final= 115903.3 dólares
Comisiones pagadas= 9721.4 dólares
Retorno total = 15.9%
Sharpe Ratio= 1.89
Máximo *drawdown* = 3.03%
Duración del máximo *drawdown* = 120718 periodos (de un minuto cada uno)
Señales generadas = 7478

Claramente esta estrategia es muy superior a la anterior, ya que hemos obtenido un retorno de casi el 16 por ciento, contra un *drawdown* máximo de solo el 3%, todo teniendo en cuenta además costos por slippage y comisiones, es decir, en una prueba considerablemente realista. El Sharpe Ratio obtenido es mayor a 1, es decir, que por cada unidad extra de riesgo obtenemos un retorno superior a esta unidad, lo cual a priori es muy positivo para la estrategia.

Vale aclarar además que esta estrategia nos generó muchas más operaciones que la anterior y por lo tanto, también hemos pagado mucho más dinero en comisiones. En la estrategia anterior realizábamos solo 10 operaciones pagando una comisión de solo 13 dólares, mientras que en este caso operamos 7478 veces, generando un costo por comisiones de 9721 dólares. Sin embargo, a pesar de este elevado costo que hemos pagado esta estrategia nos ha dado un retorno de casi 16000 dólares, ya descontando las comisiones, mientras en el caso anterior perdíamos dinero. Una posibilidad interesante sería la de buscar un bróker con comisiones menores a las de IB, ya que, de encontrarlo, esta estrategia mejoraría significativamente y podríamos observar métricas de rendimiento aún mejores.

A continuación, se muestra un gráfico de dichas medidas a lo largo de este periodo:



XI. Optimización

En el capítulo anterior vimos como construimos dos modelos para realizar predicciones en los precios de los activos financieros para luego desarrollar sistemas de trading basados en ellos. Podemos observar que había uno o más parámetros en estos modelos: en el primer caso se trataba de la cantidad de periodos a tener en cuenta en cada una de las medias móviles, en el segundo los parámetros eran el número de periodos a tener en cuenta para construir la línea de regresión y los 2 valores z para determinar la entrada y salida del mercado. Para ambos casos, estos parámetros se determinaron arbitrariamente, sin embargo, podemos mejorar el desempeño de nuestras estrategias optimizando sistemáticamente estos parámetros a través de distintos métodos (Michael L. Halls-Moore, 2015).

Sin embargo, antes de realizar esta optimización debemos tener dos factores en consideración. El primero es la sobre optimización o sobreajuste, mencionado anteriormente, el cual consiste en obtener un modelo que se desempeñe muy bien dentro de la muestra de datos que se utilizó para entrenarlo, pero nos ofrece resultados pobres al implementar el modelo con datos reales, inciertos a la hora de realizar esta optimización y por lo tanto, poco rentables. El segundo es el costo computacional que puede llegar a tener en muchos casos esta optimización si se involucran demasiados parámetros en la misma.

Teniendo esto en cuenta, vamos a proceder a optimizar los parámetros para la segunda estrategia que estudiamos anteriormente ya que esta fue la que mejores resultados nos había brindado y por lo tanto la que más interés tenemos en mejorar para aplicarla. Esta estrategia intradiaria para el trading de pares de acciones tiene 3 parámetros que podemos optimizar: el número de periodos a tener en cuenta para el OLS, el z mayor que nos dará las entradas, y el z menor que nos indicará las salidas. Lo que haremos será considerar un rango de valores para cada parámetro y mediante el producto cartesiano de estos 3 rangos de valores, considerar todas las combinaciones posibles de los 3 parámetros en esos rangos, guardar este producto cartesiano en un diccionario de Python y simular nuestra estrategia para cada combinación posible de parámetros. Este proceso se conoce en el área de machine learning como *grid search* (Michael L. Halls-Moore, 2015). Como el proceso de backtesting que llevamos a cabo para cada posibilidad es intenso para una CPU normal, pondremos un rango de solo 3 opciones para cada parámetro, dándonos un total de 27 combinaciones posibles para probar.

Parámetro 1: Periodos tenidos en cuenta por el algoritmo OLS = {50, 100, 200}

Parámetro 2: Valor z para la entrada = {2.0, 3.0, 4.0}

Parámetro 3: Valor z para la salida = {0.5, 1.0, 1.5}

A continuación, se muestra la tabla de resultados para cada una de las 27 simulaciones:

Parámetro 1	Parámetro 2	Parámetro 3	Retorno total (%)	Retorno anualizado (%)	Sharpe Ratio	Máximo DD	Duración del DD
50	2.0	0.5	213.96	20.19	1.63	42.55	255568
50	2.0	1.0	264.9	23.13	2.18	27.83	160319
50	2.0	1.5	167.71	17.15	1.63	60.52	293207
50	3.0	0.5	298.64	24.9	2.82	14.06	35127
50	3.0	1.0	324.0	26.14	3.61	9.81	33533
50	3.0	1.5	294.91	24.71	3.71	8.04	31231
50	4.0	0.5	212.46	20.1	2.93	8.49	23920
50	4.0	1.0	222.94	20.74	3.5	8.21	28167
50	4.0	1.5	215.08	20.26	3.66	8.25	22462
100	2.0	0.5	378.56	28.62	2.54	22.72	74027
100	2.0	1.0	374.23	28.43	3.0	15.71	89118
100	2.0	1.5	317.53	25.83	2.93	14.56	80624
100	3.0	0.5	320.1	25.95	3.06	13.35	66012
100	3.0	1.0	307.18	25.32	3.2	11.57	32185
100	3.0	1.5	306.13	25.27	3.52	7.63	33930
100	4.0	0.5	231.48	21.25	2.82	7.44	29160
100	4.0	1.0	227.54	21.01	3.11	7.7	15400
100	4.0	1.5	224.43	20.83	3.33	7.73	18584
200	2.0	0.5	461.5	31.97	2.98	19.25	31024
200	2.0	1.0	461.99	31.99	3.64	10.53	64793
200	2.0	1.5	399.75	29.52	3.57	10.74	33463
200	3.0	0.5	333.36	26.58	3.07	19.24	56569
200	3.0	1.0	325.96	26.23	3.29	10.78	35045
200	3.0	1.5	284.12	24.15	3.21	9.87	34294
200	4.0	0.5	245.61	22.06	2.9	12.52	51143
200	4.0	1.0	223.63	20.78	2.93	9.61	40075
200	4.0	1.5	203.6	19.55	2.96	7.16	40078

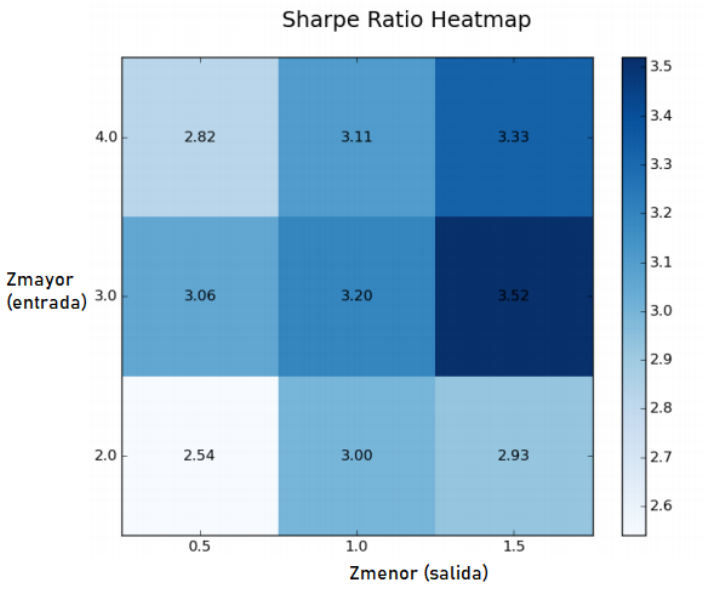
El mejor Sharpe Ratio de 3.71 lo obtenemos para la combinación de parámetros p1=50 z1=3 y z2= 1.5 lo cual resulto en un retorno total del 294.91% y un máximo drawdown del 8.04%.

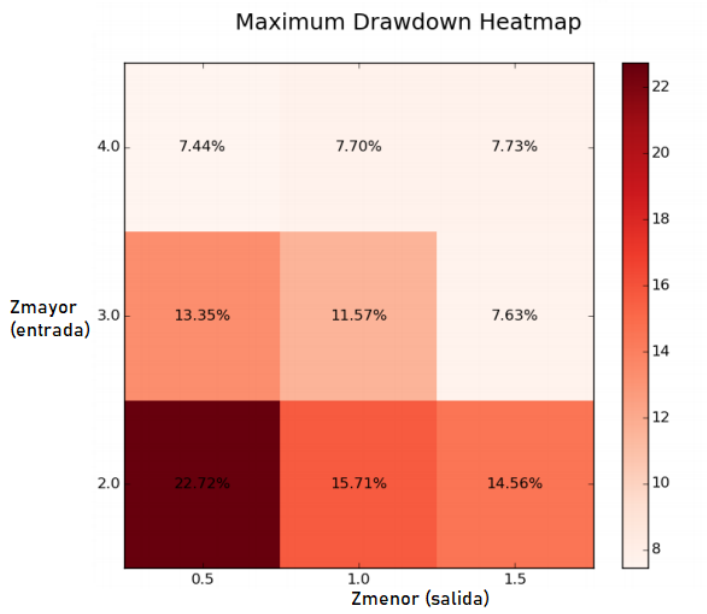
El máximo retorno total lo obtuvimos para los parámetros p1= 200, z1=2 y z3=1, obteniendo un retorno total de 461.99%, un Sharpe Ratio de 3.64 y un máximo drawdown del 10.53%.

Todas nuestras posibles estrategias devolvieron retornos positivos (de más del 150%) y todas con un Sharpe Ratio mayor a uno, lo cual también habla muy bien acerca del desempeño en general de la estrategia elegida para este par.

Ahora, vamos a tratar de visualizar los resultados de esta optimización utilizando herramientas de la librería Matplotlib de Python. Un problema para hacerlo es que nuestro problema tenía 3 dimensiones (los 3 parámetros a optimizar en conjunto) por lo que la visualización de las soluciones también sería tridimensional y por lo tanto, difícil de comprender. Sin embargo, existen soluciones parciales para este problema, por ejemplo, dejar fijo un parámetro en un valor y graficar los diferentes resultados para las variaciones de los otros dos, dándonos un gráfico de dos dimensiones, cuya visualización es más sencilla.

Vamos a fijar el número de periodos del pasado con el cual hacemos nuestra regresión lineal a 100, y luego graficar las variaciones del máximo *drawdown* y *Sharpe Ratio* para el rango de los otros 2 parámetros.





Para la estrategia que utiliza 100 periodos en la regresión, los resultados son muy claros. El *Sharpe Ratio* se maximiza en la región superior derecha, con parámetros de entrada y salida mayores para la estrategia, mientras que el *drawdown* se minimiza aproximadamente en la misma región. Esto nos ayuda a comprender que evidentemente lo mejor para la implementación de esta estrategia seria considerar valores z de entrada y salida relativamente altos.

XII. Inteligencia artificial.

El termino inteligencia artificial se refiere al uso de computadoras para reproducir o imitar funciones cognitivas de los seres humanos (Söhnke M. Bartram, Jürgen Branke y Mehrshad Motahari, 2020). Existen infinidad de aplicaciones para la IA, y naturalmente, muchas de ellas son en el ámbito de las finanzas. La IA puede tomar decisiones basadas en gran cantidad de información y datos de los diferentes activos financieros, analizando todos estos datos complejos de los mercados financieros a los cuales tiene acceso y actualizando los algoritmos de forma automática en base a los resultados anteriores, mejorando así la precisión en el proceso y la toma de decisiones, respecto a un algoritmo que siga estrictamente reglas prefijadas y nunca refine su operatoria.

En otras palabras, mediante la inteligencia artificial podemos crear algoritmos que aprendan constantemente no solo del pasado sino también del presente, en tiempo real, y utilicen ese aprendizaje para mejorar su operatoria a futuro, sin necesidad de la constante intervención humana para monitorear el funcionamiento del algoritmo y desarrollar mejoras en el mismo. Obviamente, esta clase de algoritmos puede llegar ofrecer resultados muy superiores a aquellos que solo siguen reglas precargadas sin ningún tipo de aprendizaje, ya que, como mencionamos anteriormente, ninguna estrategia funciona para todos los escenarios a lo largo de toda la historia, y por lo tanto, es necesario observar constantemente las circunstancias del mercado y mejorar nuestros algoritmos a medida que estas cambian. La IA sin embargo, viene a plantear la pregunta, que pasaría si los algoritmos pudiesen mejorarse solos, y adaptarse por sí mismos a las condiciones cambiantes del mercado.

Esto puede parecernos una utopía muy lejana. Sin embargo, el lector se sorprenderá al saber que ya hemos desarrollado una estrategia de este tipo en este mismo trabajo. Así es, la estrategia de reversión a la media de pares que creamos y optimizamos en capítulos anteriores involucra inteligencia artificial, ya que utiliza un modelo predictivo llamado regresión lineal, que, si recuerdan bien, re recalculaba constantemente periodo a periodo, observando el pasado para crear una nueva recta de regresión que modelara la relación entre los precios de ambos activos, y, si esta relación fuese a cambiar, este cambio se vería reflejado en las nuevas rectas de regresión, y por lo tanto nuestro modelo siempre daría una aproximación genuina de la relación de estos activos en sus precios, inclusive si esta varia a lo largo del tiempo (mientras que esta relación se mantenga lineal, claro está, lo cual es la gran debilidad de esta estrategia). Como verán, no es necesario realizar algo muy complejo o extravagante para involucrar a la inteligencia artificial en nuestros algoritmos, aunque obviamente existen aplicaciones mucho más complejas y sofisticadas que la mencionada anteriormente.

El caso expuesto en el párrafo anterior se trata de un modelo de *machine learning*, un campo particular dentro de la IA que, como su nombre lo indica, permite que las máquinas aprendan sin ser expresamente programadas para ello, lo cual permite hacer sistemas capaces de identificar patrones entre los datos para hacer predicciones (Söhnke M. Bartram, Jürgen Branke y Mehrshad Motahari, 2020).

Existen 2 grandes corrientes en el *machine learning*:

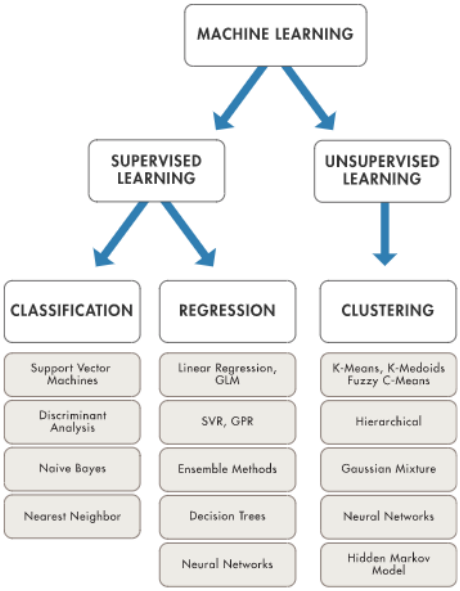
- **'Aprendizaje supervisado'**: se produce cuando se entrena a las máquinas con datos etiquetados. Este es el caso del modelo de regresión lineal, en el cual alimentamos al modelo con precios de dos activos, etiquetados como X e Y en nuestro modelo de regresión. El aprendizaje supervisado emplea técnicas de clasificación y regresión para desarrollar modelos predictivos:

    - Las **técnicas de clasificación** predicen respuestas discretas; por ejemplo, si un correo electrónico es legítimo o es spam, o bien si un tumor es cancerígeno o benigno y, un caso más de nuestro interés, si el precio de la rueda siguiente de un cierto activo será por encima o por debajo del precio de cierre de la rueda anterior, por ejemplo. Algunos algoritmos habituales para realizar la clasificación son: máquina de vectores de soporte (SVM), *k*-vecino más cercano, clasificadores bayesianos y análisis discriminante (Michael L. Halls-Moore, 2015).

    - Las **técnicas de regresión** predicen respuestas continuas. Algunos algoritmos habituales de regresión son: modelo lineal, modelo no lineal, regresión LASSO, regularización, regresión por pasos, árboles de decisión y redes neuronales (Michael L. Halls-Moore, 2015).

- **'Aprendizaje no supervisado'**: las máquinas no identifican patrones en bases de datos etiquetadas, sino que buscan similitudes. En este caso, los algoritmos no están programados para detectar un tipo específico de datos, sino que buscan ejemplos que se parezcan y puedan agrupar.

    - El ***clustering*** es la técnica de aprendizaje no supervisado más común. Se emplea para el análisis de datos exploratorio, con objeto de encontrar patrones o agrupaciones ocultos en los datos. Algunos algoritmos habituales para realizar el *clustering* son: k-*means* y k-*medoids*, *clustering* jerárquico, modelos de mezclas gaussianas, modelos de Markov ocultos, mapas auto organizados, *clustering* difuso de c-*means* y *clustering* sustractivo (Michael L. Halls-Moore, 2015).



El objetivo de este capítulo era introducir el concepto de inteligencia artificial y sus aplicaciones en el trading algorítmico, en especial aquellas relacionadas con el *machine learning*, pero no se entró en demasiada profundidad ya que excedería a los límites de este trabajo. Queda como tarea pendiente para el lector investigar más sobre el tema si este capítulo cumplió su objetivo de disparador motivacional.

XIII. Conclusión.

A lo largo de este trabajo hemos visto varios conceptos importantes en el trading algorítmico, culminando el mismo en la creación, prueba y optimización de distintas estrategias. Claramente, desarrollar un algoritmo que opere eficientemente no es tarea sencilla e, incluso al hacerlo, también queda claro que este no será efectivo en el mercado eternamente ya que eventualmente las condiciones del mismo cambiaran, por lo que es vital monitorear el desempeño contantemente y buscar riesgos y mejoras en nuestro sistema. Para ello resulta muy útil la creatividad para aplicar conceptos de la matemática y la estadística de maneras que puedan darnos una ventaja, como hemos observado anteriormente.

También han quedado claro los riesgos de desarrollar un pobre algoritmo, ya que fácilmente podríamos perder dinero si nuestra estrategia no nos ofrece una verdadera ventaja competitiva en el mercado. No debemos dejarnos llevar por los resultados obtenidos al sobre optimizar parámetros ya que de hacerlo tendremos resultados muy desalentadores al operar en la realidad, la optimización de parámetros debe hacerse siempre comprendiendo lo que esto implica para nuestra estrategia. Tampoco sería correcto tomar a la ligera el backtesting de nuestro algoritmo, ya que hemos visto la facilidad con la que podemos introducir sesgos, u obviar factores importantes como los costos transaccionales, que podrían llevar una estrategia de parecer ganadora a ser completamente perdedora, y eso que no hemos enfrentado en este trabajo la difícil tarea de considerar el impacto de mercado, lo cual sería vital si nuestro algoritmo fuese a implementarse institucionalmente.

Ha quedado claro además, que el trading algorítmico no se trata solo de cuanta rentabilidad esperamos obtener sino también del riesgo que estamos dispuestos a correr para hacerlo, teniendo presente no solo la profundidad del drawdown que podemos esperar de nuestra estrategia sino también la duración, ya que claramente no es lo mismo soportar un drawdown durante una semana o un mes que durante varios años, lo cual probablemente resultaría muy doloroso para la mayoría de la gente y nos haría perder el control de nuestras emociones y detener la operación de un algoritmo que quizás hubiese funcionado a largo plazo, pero la persona no lo deja funcionar por no comprender el concepto del drawdown en todas sus dimensiones. Esto también nos lleva a la conclusión de que el algoritmo se debe personalizar no solo para el mercado y los activos donde operará, sino también para la persona encargada de su gestión y su perfil psicológico, es decir, debemos asegurarnos que el algoritmo que creemos sea adecuado al riesgo que estamos dispuestos a correr. Un ejemplo claro de esto sería un algoritmo que opere para un fondo común de inversión, donde muy probablemente la prioridad se centre en disminuir el drawdown más que en maximizar las ganancias, ya que de sufrir muchas pérdidas el fondo podría enfrentarse a retiros masivos de capital debido al miedo de sus subscriptores y la estrategia se estropearía.

Muchos creen que el trading algorítmico es la forma de evitar errores humanos a la hora de operar en el mercado de capitales, y, si bien esto es parcialmente cierto, queda claro que la posibilidad de errores humanos no queda del todo eliminada, ya que hemos visto que estos pueden aparecer en reiteradas oportunidades a lo largo del proceso de desarrollo e implementación de una estrategia eficiente.

XIV. Bibliografía:

Oscar Yesid Quintero Delgado y Jonathan Ruiz Delgado (2011). Estimación del exponente de Hurst y la dimensión fractal de una superficie topográfica a través de la extracción de perfiles. Artículo de investigación publicado por la Universidad Distrital "Francisco José de Caldas".

Michael L. Halls-Moore (2015). Successful Algorithmic trading.

Philip Treleaven, Michal Galas, and Vidhi Lalchand (2013). Algorithmic Trading Review. Communications of the ACM, Volume 56, Chapter 11.

Rogelio A. A. Morán (2017). Pronósticos de demanda.

Söhnke M. Bartram, Jürgen Branke, and Mehrshad Motahari (2020). Artificial intelligence in asset management. CFA Institute Research Foundation.

Referencias electrónicas:

Shobhit Seth, (2021). Basics of Algorithmic Trading: Concepts and Examples. Consultado el 20 de marzo de 2021: https://www.investopedia.com/articles/active-trading/101014/basics-algorithmic-trading-concepts-and-examples.asp

Rakesh Sharma (2021). Quantitative trading. Consultado el 22 de marzo del 2021: https://www.investopedia.com/terms/q/quantitative-trading.asp

Gordon Scott (2021). Application programming interface-API. Consultado el 24 de marzo de 2021: https://www.investopedia.com/terms/a/application-programming-interface.asp