UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CIENCIAS

Métodos de machine learning aplicados a la construcción de ETFs para su uso en estrategias de arbitraje estadístico.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Actuario

PRESENTA:

Carlos Torres Salinas

TUTOR

MSc. Claudio Cuevas Pazos

CIUDAD UNIVERSITARIA, CD. MX., 2022

Resumen

El trabajo consiste en la construcción de portafolios de inversión que repliquen el índice *S&P 500*[1], utilizando métodos de *machine learning*, o aprendizaje de máquina, para poder utilizarlos en un algoritmo de operación bursátil conocido como *pair trading*, el cual pertenece al ramo de las estrategias de arbitraje estadístico. Para hacer uso de esta estrategia, se necesitan dos activos altamente correlacionados, los cuáles se obtendrán con los métodos previamente mencionados.

La motivación detrás de este trabajo de investigación deriva del estudio de los nuevos métodos utilizados en mercados financieros para realizar operaciones de compra/venta de activos, como lo son los algoritmos de *trading* y los modelos de *machine learning*.

Los modelos de *machine learning* se basan en el aprovechamiento de grandes cantidades de datos, así como del uso de funciones matemáticas para la construcción de modelos computacionales, los cuales pueden ser usados para realizar proyecciones, cálculo de riesgo y selección de variables, entre otras funciones utilizadas en los mercados financieros.

Por otro lado, los algoritmos de *trading* son conjuntos predefinidos de reglas para la ejecución, dirigidos a un objetivo específico, permitiendo la ejecución sistemática de operaciones. Existen varios tipos de algoritmos, y cada algoritmo puede ser personalizado gracias a los parámetros que utilizan.

La investigación necesaria para el presente trabajo incluye conocer los algoritmos de *trading*, clasificaciones y su uso actual. Asimismo, es necesario adentrarse en el arbitraje estadístico y específicamente en el algoritmo de *pair trading*. También resulta indispensable entender los modelos de *machine learning* que sirven para aprovechar el flujo de datos, su modelación estadística y su implementación computacional.

Finalmente, se analiza el rendimiento de una estrategia de inversión de arbitraje estadístico, entre nuestros portafolios y un portafolio comercialmente disponible, y se comparan rendimientos.

La hipótesis del trabajo consiste en analizar la posibilidad de crear un portafolio que replique exitosamente el índice utilizando modelos de *machine learning* con un subconjunto de las acciones del índice. Para ello, se utilizará un modelo de aprendizaje supervisado y uno de aprendizaje no supervisado, lo cual veremos a detalle en capítulos siguientes. Posteriormente, se construirá la estrategia de *pair trading* para comprobar si existe la posibilidad de generar rendimientos positivos con dicho algoritmo.

[1] El *S&P 500* es un índice bursátil con 500 empresas de las más grandes de Estados Unidos.

Dedicatoria

La dedicatoria sería demasiado extensa si abarcara a todas las personas importantes en mi vida, así que sólo haré algunas menciones, principalmente a mi familia, que a lo largo de mi vida, han confiado en mí para lograr este objetivo.

A mi papá, quien me enseñó el camino de la rectitud, del esfuerzo y del trabajo duro. Que siempre buscó sacar a toda la familia adelante y se dedicó en cuerpo y alma a ella. Por él, quien siempre quiso que fuera un hombre de bien y triunfara en la vida, le dedico este logro. Nunca olvidaré lo que para mí ha sido su enseñanza más grande: *Los mejores días de tu vida están por venir*.

A mi madre, quien comenzó este camino junto a mí, desde el día en que entré a mis primeros años de educación básica y que estuvo conmigo incontables noches haciendo tarea, estudiando y leyendo conmigo. Me llevó de la mano por un camino de compromiso y entrega. Jamás se dio por vencida, jamás se rindió cuando yo no podía más. Ella es quien mantiene la unión familiar y confía siempre en nuestros sueños y nos ayuda a volverlos propósitos cumplidos.

A mi hermano, la persona que más admiro en el mundo, por ser mi compañero, mi mejor amigo, mi apoyo. Porque entre bromas y juegos siempre me demostró que yo podía lograr cualquier meta que pusiera frente a mí. Me apoyó para levantarme cuando caía, quien me mostró que era capaz de más cosas de las que algún día imaginé.

A mi tío, el Ing. Luis Ramos y mi tía Soco Torres, quienes siempre vieron mucho potencial en mí. Me regalaron muchos libros y siempre me invitaron a estudiar, a ser mejor, a tener un título y en pocas palabras, me ayudaron a ser quien soy hoy. Son como unos segundos padres para mí.

A mis primos Juan Pablo Sunderland y Karla Saavedra, por su constante apoyo y cariño, así como su motivación para que confíe en mis capacidades. Además, Juan Pablo fue el primero en llevarme a una casa de bolsa, y despertó en mí la pasión por el medio financiero y las inversiones.

También le dedico este trabajo con mucho cariño, a mi sobrina Lia, para que sepa que podrá lograr todo lo que se proponga en su vida, si pone esfuerzo y dedicación en ello.

Finalmente, una mención especial para el Dr. Ricardo Gallardo por su constante apoyo a lo largo de los años, así como su insistencia para que concluyera este proyecto.

A ellos, y al resto de mi familia y amigos, quienes siempre han creído en mí, y en mi capacidad para lograr este proyecto, les dedico este trabajo que es la culminación de nuestro esfuerzo y dedicación, así como un reflejo de su confianza y entrega. Sin ellos, esto no hubiera sido posible.

Agradecimientos

Un capítulo entero no alcanzaría para agradecer a todas las personas que me apoyaron en este esfuerzo a lo largo de los años: familiares, amigos, profesores y compañeros de profesión. Sin embargo, quisiera dedicar esta sección a mis maestros: desde mis inicios como estudiante hasta mi consolidada formación profesional. Gracias a ellos por dedicar su vida a la enseñanza y compartir un poco de su conocimiento conmigo.

Particularmente, quiero agradecer al MSc. Claudio Cuevas Pazos por guiarme y asesorarme a lo largo de este trabajo, por confiar en mí y en mis conocimientos, así como en mis habilidades para desarrollar el presente escrito. Porque fue él quien me introdujo en el mundo de las finanzas cuantitativas y los algoritmos de *trading*, y fue ahí en donde encontré mi pasión por la carrera. Porque gracias a él disfruto de la investigación en estos temas y siempre buscaré ir más allá en mi conocimiento de los mismos. Gracias, porque nunca se cansó de revisar mi trabajo, ayudarme a resolver dudas y a encaminarme para que este trabajo llegara a buen término. Siempre me apoyó y me presionó para terminar esta labor que es la cumbre de toda una carrera de esfuerzo y sin él, no hubiera logrado.

También quiero agradecer al Dr. Heber Longhurst, *CFA*, porque quizás sin saberlo, su constante ejemplo de entrega al estudio y la investigación, me inspiraron para dar mi mayor esfuerzo en este trabajo, buscando siempre que fuera un trabajo con bases sólidas y buenos argumentos, para hacer de él un trabajo digno de mi grado.

Finalmente, agradezco a los sinodales que se tomaron el tiempo de revisar mi trabajo, señalar las correcciones pertinentes y ayudarme para que el presente escrito fuera lo mejor que podía llegar a ser.

A ellos, y todas las personas que me han acompañado desde el comienzo de mis estudios básicos hasta esta consolidación del estudio profesional, muchas gracias.

Índice

Capítulo 1

Introducción

1.1 Motivación y metodología

El avance tecnológico global ha alcanzado todos los sectores productivos de la población: desde el industrial hasta el de servicios. En este último se han dado mejoras gracias a las nuevas herramientas tecnológicas, los programas de computación y las nuevas disciplinas como la inteligencia artificial, la ciencia de datos, *big data*, entre otros.

Ya sea por la complejidad de los problemas a solucionar, por la cantidad de datos que influyen para la toma de decisiones, o por la velocidad y precisión con la que se realizan operaciones en la computadora, estos avances han tenido un auge en una gran cantidad de aplicaciones: desde las matemáticas, con las soluciones numéricas, hasta el sector financiero, con nuevos modelos, formas de ejecución como los algoritmos de operación bursátil, o algoritmos de *trading* y el procesamiento de información. Esto provocó la necesidad de establecer una nueva forma de plantear modelos matemáticos ayudados por métodos computacionales para estar siempre a la vanguardia. En el ámbito financiero, estos modelos se utilizan para cálculos numéricos complejos, proyecciones, medidas de riesgo, valuaciones y algoritmos, entre muchas otras cosas.

El presente trabajo se enfoca en la creación de un portafolio que replique un índice bursátil por medio de métodos de aprendizaje de máquina, conocidos más comúnmente como *machine learning* en inglés. Posteriormente, se utilizan dichos portafolios en un algoritmo de *trading* con el fin de generar rendimientos positivos.

Los métodos de *machine learning* consisten en encontrar una función matemática capaz de relacionar una serie de datos independientes con una serie de datos objetivo, aprovechando las interacciones entre ellos y siempre buscando minimizar el error entre el dato estimado y el dato real por medio del ajuste de parámetros y la iteración de procesos.

Para lograrlo, se introduce una serie histórica de variables $\mathbf{X}$ independientes en el modelo, con el fin de estimar una variable dependiente $\mathbf{y}$ conocida. Con esto, el modelo puede encontrar una función que pueda describir los datos de salida en términos de los datos de entrada, e iterar sobre los parámetros de manera que, el error entre el dato producido por el modelo y el dato objetivo real, sea mínimo.

Los parámetros, así como la forma en que se encuentra la función óptima para relacionar las variables, cambia entre los muchos tipos de modelos que existen. Por ejemplo, se pueden fijar parámetros preestablecidos por el usuario o construir un algoritmo con la tarea de encontrar los parámetros óptimos.

Entre estos modelos se encuentran las regresiones lineales y logísticas, los árboles de decisión, los *random forest* o bosques aleatorios, K vecinos cercanos, los modelos de regularización como las regresiones Lasso y Ridge, las máquinas de soporte vectorial y otros modelos conocidos como *deep learning* o aprendizaje profundo entre los cuales están los distintos tipos de redes neuronales. En todos ellos, una computadora construye una función óptima bajo ciertas métricas a partir de la relación que encuentre entre datos de entrada y salida, conocidos como datos de entrenamiento.

Se utilizan datos del mercado de capitales de Estados Unidos buscando replicar el índice bursátil S&P 500 a partir de un subconjunto de las acciones que lo componen para generar un portafolio con menos acciones pero que, en teoría, deben tener una correlación elevada con un portafolio real que replique el índice.

Al tener una correlación elevada con un portafolio utilizado por el mercado, se puede aplicar un algoritmo de *pair trading*, un algoritmo de arbitraje estadístico basado en la teoría de reversión a la media, buscando generar rendimientos positivos.

El trabajo se implementa en *Python* pues es, en mi opinión, un lenguaje de programación que posee un buen balance entre sintaxis amigable y potencia computacional. Además de que es considerado como uno de los mejores lenguajes de programación para ciencia de datos y *machine learning*, debido a la amplia gama de bibliotecas útiles para este fin[1].

1.2 Objetivo e hipótesis

El objetivo general del presente trabajo se centra en la construcción de portafolios que permitan replicar el índice bursátil S&P 500 de Estados Unidos, utilizando modelos de *machine learning*, para posteriormente, utilizar dichos portafolios en estrategias de inversión basadas en algoritmos de arbitraje estadístico, buscando generar rendimientos positivos.

Primero, se sientan las bases para entender el mercado financiero y se da un repaso por conceptos como la microestructura de mercado, los tipos de órdenes y la definición de un *Exchange Traded Fund* o ETF. Lo siguiente es ver una clasificación de algoritmos de *trading* explicando a detalle el algoritmo que se utiliza en el presente trabajo: *pair trading*, un algoritmo del tipo oportunista.

Posteriormente, se analiza la teoría detrás de los modelos estadísticos de aprendizaje computacional para poder construir el portafolio de réplica, construyendo dos modelos de aprendizaje supervisado: uno paramétrico y uno no paramétrico. Por último, se realiza la construcción del algoritmo de *trading* y el análisis de resultados obtenidos.

[1]Un ejemplo es la encuesta de *Kaggle* del 2018 donde el lenguaje más utilizado por los usuarios fue Python, con un 83% de los usuarios, el segundo lugar fue SQL con 44% y el tercero fue R con un 36%. https://www.kaggle.com/kaggle/kaggle-survey-2018

La hipótesis del trabajo, se fundamente en que los métodos de *machine learning* sirven para construir un portafolio que replique el índice bursátil S&P 500, utilizando un subconjunto de las acciones que lo componen. Además se tiene la hipótesis de que, si el portafolio replica el índice de manera correcta, entonces tiene una correlación suficiente con un portafolio comercialmente disponible que ya replique dicho índice; y se puede construir una estrategia de arbitraje estadístico que sea rentable.

Para comprobar las hipótesis, se construyen los portafolios basados en modelos de aprendizaje de máquina y se analizan ciertas métricas de valuación, de las cuáles se hablará más adelante, así como la correlación con el índice subyacente.

Si existe correlación entre el portafolio generado con respecto al mercado y, aunado a esto, las métricas de medición son similares a los portafolios comercialmente disponibles — cuya finalidad es igualmente la de replicar el índice—, entonces se considera que la primera hipótesis es correcta y se procede a aplicar el algoritmo de *trading*. Finalmente, se analizan los rendimientos para comprobar la segunda hipótesis.

1.3 Estructura del trabajo y literatura de referencia

El primer paso para lograr el objetivo es entender la teoría detrás de los modelos de los algoritmos de *trading*, el por qué funcionan y cómo están clasificados. Aquí se da una breve descripción de los tipos de algoritmos según su objetivo, cómo está compuesto el mercado, algunos costos relacionados a las operaciones y, finalmente, se trata el tema del algoritmo a utilizar. Para la bibliografía principal, ver [1].

También es importante entender sobre aprendizaje estadístico, conocer sobre los métodos aplicables, su construcción y su clasificación según el problema del que se trate. Aquí se analizan las ventajas y desventajas de algunos modelos y el costo entre sesgo y varianza, así como el de interpretabilidad y rigidez. Posteriormente se estudian esos modelos a profundidad, incluyendo su motivación, fundamento matemático, aplicación y parametrización. Además, se incluye un análisis de selección de variables para mejorar estimaciones. Para la bibliografía principal, ver [2].

Finalmente, se aterriza toda la teoría en un problema práctico que se realiza con el apoyo del lenguaje de programación *Python*. En este programa se incluye la construcción de los modelos y posteriormente los algoritmos de *trading*, así como resultados y conclusiones del trabajo y un apartado para investigación futura.

Para la construcción de los algoritmos utilizando bibliotecas en *Python* se utilizan diversos libros e información obtenida en cursos presenciales y en línea, así como blogs de ayuda para programadores y las mismas páginas de ayuda de las bibliotecas.

La literatura consultada, así como páginas de ayuda y demás recursos utilizados para la presente investigación, se pueden revisar en la bibliografía.

Parte I

Mercados financieros y algoritmos de *trading*

Capítulo 2

Bases de mercados financieros

2.1 Microestructura de mercado

El campo de la microestructura de mercado se concentra en el proceso actual de *trading*[1], al analizar cómo los mecanismos específicos afectan tanto a los precios observados como a los volúmenes de operación. La microestructura de mercado ayuda a explicar muchos de los costos que previenen que los activos alcancen sus valores dados por el análisis fundamental.

La microestructura de mercado cubre temas como:

- Estructura y diseño de mercado.
- Investigación de mecanismos de operación o *trading*.
- Análisis y medición de costos de transacción.

2.2 Fundamentales

Los mercados existen para que sea posible realizar transacciones. Esto se logra al reunir diferentes participantes en un mismo sitio, y permitirles el intercambio de activos. La liquidez de un activo representa la facilidad con la que éste se puede convertir en efectivo. Realizar transacciones en mercados altamente líquidos es mucho más sencillo y más eficiente que hacerlo en mercados con poca liquidez.

2.2.1 Función del mercado

El propósito fundamental de un mercado es reunir a compradores y vendedores. Los mercados pueden ser categorizados en **primarios**, donde se encuentran las nuevas ofertas de instrumentos, y **secundarios**, donde toman lugar el resto de las transacciones.

La investigación de microestructura de mercado ha estado enfocada principalmente en la eficiencia de los mercados secundarios. En particular, examinando el diverso rango de estructuras de mercado y mecanismos de *trading*.

[1] El término trading en inglés se utiliza para referirse a la compra y venta de activos financieros.

2.2.2 Participantes

Por convención, el *buy side* corresponde a los clientes tradicionales: inversionistas individuales e institucionales, mientras que el *sell side* representa a los corredores, los distribuidores u otros intermediarios financieros. Los corredores actúan como agentes que facilitan las operaciones actuales, mientras que los distribuidores —o creadores de mercado, en inglés *market makers*— operan en su propio beneficio tratando de obtener ganancias por ofrecer liquidez al mercado.

2.2.3 Liquidez

Operar significa, generalmente, convertir un activo en efectivo o viceversa. La liquidez del activo representa cuánto costaría esta conversión. La liquidez puede verse en términos de la inmediatez al verse reflejada en qué tan rápido se puede operar un activo al mejor precio posible. Así, el precio de un activo está estrechamente vinculado a su liquidez.

Un mercado o activo líquido debe tener un costo menor por inmediatez; por ejemplo, costos de transacción. Asimismo, tendrán un mayor volumen de operación.

Los mercados de acciones más líquidos del mundo son: La bolsa de Nueva York —el *New York Stock Exchange* o NYSE—, el NASDAQ —empresas de tecnología en Estados Unidos—, y la bolsa de Londres —*London Stock Exchange* o LSE—. Dado que se utilizarán activos que se operan en la bolsa de Nueva York para el presente trabajo, no se consideran costos de liquidez asociados a la operación de los títulos.

2.3 Estructura y diseño de mercado

Para que los mercados funcionen de manera correcta, su diseño debe acomodarse a las necesidades de inversionistas, distribuidores y especuladores. Es por ello que un mercado exitoso permite a inversionistas operar cuando quieran; y minimiza los costos de las órdenes de operación, mientras siguen siendo redituables para distribuidores y especuladores.

2.3.1 Mecanismos de operación

Existen tres tipos de mecanismos de operación, los cuales se dividen en mercados por cotización *(quote-driven)*, por órdenes *(order-driven)* o una mezcla de ambos *(hybrid)*.

En un mercado movido por cotizaciones, los operadores deben realizar su transacción con un *dealer* (o un *market maker)*, que informa sobre los precios a los cuales está dispuesto a comprar y vender una cierta cantidad (la cotización del activo). Se puede escoger entre: tomar la oferta, aceptando el precio de compra o de venta, renegociar o simplemente dejarla pasar.

Por otro lado, un mercado potenciado por órdenes permite a los operadores participar por igual; colocando órdenes de compra y/o venta en un libro de órdenes, que posteriormente, son cruzadas con otras órdenes utilizando un conjunto consistente de reglas.

Los precios son establecidos por órdenes actuales, enviadas por otros operadores. El *best bid* es el precio **más alto** por el cuál alguien está dispuesto a realizar una compra, mientras que el *best offer* es el precio **más bajo** por el cual alguien está dispuesto a realizar una venta.

Una compra ocurrirá únicamente cuando una orden de compra iguale o mejore el precio de la mejor venta, mientras que una venta se realizará cuando la orden de venta iguale o mejore el precio de la mejor compra. De esta forma, en lugar de responder a la cotización de compra-venta de un creador de mercado, se reacciona a la liquidez disponible en el libro de órdenes.

Es importante mencionar que **no existe una garantía** de que una orden se ejecutará al precio exacto al cual se coloca. Al momento en que se envía la orden, la orden con la cual se hubiera podido cruzar, pudo haber sido cancelada, cambiada o cruzada con otra orden que llegó antes de la nuestra. Por ello, es posible que nuestra orden se ejecute a un precio distinto (que potencialmente nos perjudicaría) o que no se ejecute.

También se debe destacar que existen dos tipos de órdenes: una que garantiza ejecución pero no precio (*market order*) y otra que garantiza establecerse en un precio, mas no asegura su ejecución (*limit order*). Es por ello que los mejores precios de compra y venta establecidos en un mercado de órdenes, son más bien indicativos del mercado pero no están garantizados, a diferencia de la cotización de compra-venta de un comerciante.

2.3.2 Frecuencia de operaciones

La frecuencia de operaciones determina en qué momento el cruce de órdenes se convierte en ejecuciones. Generalmente, los mercados permiten los siguientes tipos:

- *Trading* continuo (*continuous*)
- *Trading* periódico (*periodic*)
- *Trading* por solicitud (*request-driven*)

El *trading* continuo provee medios eficientes y convenientes de ejecución; sin embargo, dicha inmediatez puede llevar a una volatilidad en el precio. En el caso del *trading* periódico, es generalmente programado para ocurrir en uno o más momentos específicos del día, por lo que permite una formación más considerada del precio. Además, permite que se acumule liquidez. El *trading* por solicitud significa que se requiere pedir una cotización a un generador de mercado.

El mercado de acciones de Estados Unidos es un mercado de *trading* continuo y es el tipo de operación que se asume para la realización del apartado práctico del presente trabajo.

2.3.3 Costos de transacción

A continuación se citan los más relevantes:

- ***Comisiones***: Representan la compensación del corredor por proveer el servicio de operación.
- ***Impuestos***: Aplicados a cualquier ganancia de capital realizada.
- ***Costo de spread***: Compensa a los operadores por proveer liquidez. El *spread*, diferencia entre precio de compra y precio de venta, generalmente refleja la liquidez de un activo.
- ***Costo de retraso***: Refleja cualquier cambio de precio del activo, entre la decisión inicial de invertir y el momento en que la orden fue enviada efectivamente para su ejecución.
- ***Impacto en el mercado***: Esto representa cuánto efecto tiene nuestra orden en el precio del activo. Las órdenes más grandes resultan en un mayor impacto. Sin embargo, el efecto decrece significativamente con activos de mayor liquidez.
- ***Tendencia de precio***: Una tendencia a la alza implica que los costos aumentarán cuando se compre un activo, mientras que se ahorrará al venderlo y viceversa.
- ***Riesgo de timing***: Refleja incertidumbre, en particular por la volatilidad tanto en el precio del activo como en su liquidez. Entre más volátil es un activo, es más probable que el activo se mueva lejos del precio de decisión y se incurra en costos mayores. De igual manera, si la liquidez cae en un instante, los costos por el impacto de mercado se incrementarán.
- ***Costo de oportunidad***: Las órdenes no siempre se llenan completamente, por lo que, esto representa el costo de la oportunidad perdida; pues los precios del día siguiente pueden estar más alejados del precio de decisión.

Para el apartado práctico del presente trabajo no se toman en cuenta costos de transacción, pues se asumen condiciones ideales para operar. Esto no es necesariamente cierto en todos los casos, por lo que, si se desea llevar este trabajo de lo teórico a lo práctico, se debe hacer un estudio cuidadoso de los costos de transacción, ya que estos pueden llevar una estrategia que varía de ser redituable a tener pérdidas considerables.

Particularmente, se pueden tomar en cuenta los costos de comisiones e impuestos para una investigación futura del presente trabajo, según el corredor que se esté utilizando y las leyes del país donde se opere con relación a impuestos sobre ganancias de capital. Estos son costos determinados en un porcentaje del volumen, de la ganancia de capital o de activos manejados, por lo que se puede tener un estimado previo a las operaciones.

Los costos de *spread*, impacto de mercado y de oportunidad, dependerán de la liquidez al momento de realizar la operación. Si hay poca liquidez, habrá un *spread* entre la compra y la venta, el impacto en el libro de órdenes puede ser mayor y quizás no se alcance a llenar la orden en ese día. Estos costos dependen enteramente del momento en que se opera en el mercado.

Los costos por retraso, *timing* y tendencia dependerán de la situación particular del mercado en cuanto a volatilidad, liquidez y tendencia en precios. Si es un día muy volátil, el retraso de un microsegundo puede cambiar drásticamente el precio; y si la tendencia es en contra de la operación, puede ser un costo importante a considerar. Al igual que los costos anteriores, estos dependen completamente de la situación del mercado al momento de las operaciones.

2.4 Órdenes

Una orden es simplemente una instrucción para comprar o vender una cantidad específica de un activo dado. A grandes rasgos se pueden distinguir entre dos tipos de órdenes principales:

- **Orden de mercado**: Son órdenes dirigidas a realizar una operación de manera inmediata al mejor precio posible. Demandan liquidez y, si bien pueden ser ejecutadas completamente, puede existir un riesgo por la incertidumbre del precio final de ejecución.

- **Orden limitada**: Son órdenes que tienen un precio fijo incorporado que no debe ser violado, con un precio máximo para compras y uno mínimo para ventas. De esta forma, estas órdenes garantizan el precio al cual se ejecutaría la operación, si es que llega a ejecutarse, pues arriesgan el que no exista un cruzamiento de la orden por el precio dado, fallando en su ejecución.

Las condiciones que se pueden aplicar a cualquier orden para permitir que el operador tenga control sobre muchas características, incluyen, entre otras cosas:

- Cómo y cuándo se vuelve activa la orden.
- Su duración o tiempo de vida.
- Si puede ser completada parcialmente.
- Si puede ser enviada a otros mercados (bolsas).
- Si puede tener conexión con otras órdenes.

Un amplio rango de estilos de *trading* pueden ser alcanzados sólo combinando estas condiciones con los tipos de órdenes. Estas estrategias siguen evolucionando al día de hoy y algunos mercados incluso ofrecen estrategias similares a algoritmos de inversión.

Para el presente trabajo se decidió optar por una orden de mercado al cierre del día. Esto es, una orden que no tiene un precio límite pero debe llenarse por completo y su precio debe ser el mismo que el precio de cierre, el cual se calcula normalmente como el promedio ponderado de las operaciones de los últimos 20 minutos del día. A este tipo de órdenes se les conoce como orden al cierre, o al PPP[2], en inglés conocida como *market on close*.

[2]PPP es por Precio Promedio Ponderado de los últimos 20 minutos del día. El precio de cierre se calcula así para evitar que algún operador coloque una orden en el último minuto del día esperando marcar un precio muy alto o muy bajo, a conveniencia propia.

Ahora es necesario entender el mercado sobre el cual se va a trabajar. En este caso es el mercado accionario, comúnmente conocido en inglés como *Equity market*.

2.5 Mercado accionario

El mercado de capitales se divide en mercado de renta variable que corresponde a las acciones, mercado de renta fija, o mercado de deuda, que es la parte de bonos y mercado de derivados.

El mercado de renta variable, también conocido como mercado accionario, *stock market* o *equity market*, es una de las áreas más importantes[3] de la economía, pues permite a las compañías tener acceso a capital de inversión y a los inversionistas les permite ser dueños de una parte de una compañía con el potencial de realizar ganancias basadas en su rendimiento futuro o por medio de dividendos.

Cada acción representa una porción del total del valor de la compañía y son inversiones a futuro, pues los compradores esperan que los activos de la compañía incrementen su valor o recibir un pago de dividendos para compensar el riesgo de quiebra.

Determinar el valor justo de las acciones no es trivial. La teoría del valor presente dice que el valor del activo corresponde a la suma del valor presente de sus flujos futuros. Esto puede depender de los dividendos a pagarse, que a su vez dependen de las ganancias de la empresa, lo que puede ser muy difícil de pronosticar.

Existen bolsas de valores en todo el mundo. En México se tienen dos: La Bolsa Mexicana de Valores, BMV (1894) y la Bolsa Institucional de Valores, BIVA (2017). En Estados Unidos existen 13 bolsas de valores, siendo las principales la *New York Stock Exchange* o *NYSE* (1792), y el *NASDAQ* (1971), el primer mercado electrónico en el mundo.

2.5.1 ETFs

Los ETFs, llamados así por el nombre en inglés de ***Exchange Traded Funds***, son un tipo de activo compuesto por la colección de otros activos como acciones, bonos u otros. Están listados en las bolsas de valores y se operan como cualquier acción en el mercado.

Se pueden utilizar para replicar los movimientos de un índice accionario como puede ser el *S&P 500*, el *NASDAQ Composite*, el IPC (México), entre otros. Existen muchos índices que buscan replicar alguna de las bolsas de Estados Unidos. Las principales empresas que crean los fondos que replican esos índices, son *BlackRock* y *Vanguard*.

También pueden gestionarse activamente para generar rendimientos superiores al índice buscando distintas estrategias de inversión, entre las cuales puede ser: invertir en algún sector determinado, tener más de algún tipo de empresas o invertir por factores, entre otros.

Existe también una estrategia híbrida entre las estrategias pasivas y activas, conocida como indexación mejorada (o *enhanced indexing* en inglés) que consiste en utilizar un subconjunto de emisoras buscando replicar un índice, pero a un menor costo.

[3]Investopedia: Equity Markets https://www.investopedia.com/terms/e/equitymarket.asp

2.5.2 Indexación mejorada

Es una estrategia de inversión que se encuentra en un punto medio entre: una estrategia de inversión pasiva, donde únicamente se compra un ETF que replique el índice buscando obtener el mismo rendimiento que el índice al cual se está replicando; y una estrategia de inversión activa, donde se compran y venden acciones para tratar de vencer el índice de referencia, o *benchmark*.

Se busca vender determinadas acciones que, según la regla que se utilice, no aporten al objetivo deseado, utilizando el efectivo adquirido por esas ventas para comprar más posición en aquellas que aportan más al objetivo.

Es una estrategia activa en la que se busca construir un ETF con un subconjunto de empresas de un índice. Pueden escogerse como las empresas más grandes, empresas de un sector, empresas que paguen dividendos, descartar las de peor rendimiento para aumentar posición en las de mejores rendimientos, entre otras reglas. Para efectos del presente trabajo se escogen las empresas según las que indique el modelo de *machine learning* a utilizar, vendiendo las acciones que menos aporten a la correlación con el índice, y dejando las que tengan mayor aporte.

De esa forma, se busca vender las acciones que menos peso tienen para la réplica del índice y conservar las que más aportan a la explicación de la variabilidad del mismo, teniendo como resultado un ETF que replique al mercado pero con un menor número de activos.

Esto representa muchas ventajas. Por un lado, se obtiene un activo que, en teoría, debe tener una correlación elevada con otros ETFs que replican el mercado. Por otro, es más barato de rebalancear[4]. Además de eso, las comisiones por manejo son menores que un portafolio con mayor número de activos.

Particularmente a la estrategia de crear un índice propio y operarlo, se le conoce como ***index construction enhancements***.

Hay muchas formas de construir estos portafolios de "ETFs mejorados". Para el presente trabajo se utilizan los modelos de *machine learning* que vienen en capítulos subsecuentes.

2.5.3 Métrica de medición para un ETF réplica

Como ya se mencionó, el presente trabajo busca replicar el índice *S&P 500*, utilizando *enhanced indexing* al construir un ETF utilizando métodos de *machine learning*. La medida más común para saber si un ETF replica de manera correcta al índice subyacente para el cual fue creado, es el ***tracking error***[5].z

[4]En el mercado financiero se le llama rebalanceo de un portafolio a la compra y venta de acciones que se lleva a cabo cada cierto tiempo para asegurarse que el objetivo del ETF se mantenga. En el caso de querer replicar un índice, el rebalanceo se hace para asegurarse que el portafolio no se aleje del benchmark a replicar.

[5]Otras medidas pueden ser la Beta del portafolio con respecto al índice de réplica, la R^2 y la volatilidad. De estas medidas se habla en capítulos siguientes.

El *tracking error* es la divergencia entre el comportamiento del precio de una posición o un portafolio, en este caso un ETF, y el comportamiento del precio de un índice de referencia. Se reporta como la diferencia porcentual en la desviación estándar, lo cual representa la diferencia entre el rendimiento que obtiene un inversionista contra lo que hubiera obtenido de haber estado en el *benchmark*.

Se calcula como:

$$\textit{Tracking error} \;=\; \textit{Desviación estándar}\,(P \;-\; B) \tag{2.1}$$

Donde P es el rendimiento del portafolio, en este caso el ETF, y B es el rendimiento del *benchmark*, en este caso un índice.

En el presente trabajo se crean modelos para replicar al *S&P 500*, el cual es el índice con 500 empresas de las más grandes por capitalización de mercado, listadas en el *NYSE* o en el *NASDAQ*. Uno de los ETFs que replica dicho índice, creado por *State Street Global Advisors*, es el ***SPY US Equity***.

Se compara el *tracking error* entre los portafolios generados y el *S&P 500* para saber si es una réplica fiable, al menos para el año en revisión. También se realiza una comparación entre el *tracking error* de los portafolios generados con el *tracking error* del *SPY US Equity* para saber si el portafolio generado tiene una métrica similar al ETF existente en el mercado.

En el siguiente capítulo se ve la estrategia de inversión a utilizar, el algoritmo de trading, su función y uso, y el por qué es relevante saber del *SPY US Equity*.

Capítulo 3

Algoritmos de operación y arbitraje estadístico

3.1 Introducción

Un algoritmo es un conjunto de instrucciones para una tarea determinada. Por tanto, un algoritmo de *trading* define los pasos requeridos para ejecutar una operación de manera específica. Algunos tienen como objetivo el igualar o superar un *benchmark*, otros tratan de minimizar los costos de transacción; mientras que otros buscan operar de manera más oportunista.

Si bien las reglas individuales pueden ser bastante sencillas, la amplia gama de diferentes eventos y posibilidades que se deben cubrir por un algoritmo promedio, pueden volverlo complejo rápidamente. Una forma común de abordar esta complejidad ha sido dividir el problema en dos: decisiones a nivel macro y decisiones a nivel micro.

El nivel macro realiza elecciones estratégicas basadas en objetivos generales. El nivel micro considera detalles más tácticos como las especificaciones de cada envío de órdenes. El presente trabajo se enfoca en las especificaciones de un algoritmo a nivel macro.

3.2 Categorizando algoritmos

Existen muchas formas de clasificar los algoritmos. La más común, considerando el objetivo de cada algoritmo, los clasifica en tres categorías:

- *Impact-driven*
- *Cost-driven*
- Oportunistas

3.2.1 Algoritmos *Impact-driven*

Buscan minimizar el impacto que la operación tiene en el mercado. En otras palabras, tratan de reducir el efecto que la operación tiene sobre el precio de un activo.

Los algoritmos *impact-driven* evolucionaron partiendo de estrategias donde se divide la orden, a lo largo del tiempo establecido para operar. Al dividir una orden grande en otras más pequeñas, se trata de reducir el efecto que la operación tiene sobre el precio del activo, minimizando los costos por impacto en el mercado.

Los algoritmos basados en precio promedio, como el ***TWAP*** (*Time Weighted Average Price*) donde se divide una orden en partes iguales a lo largo del tiempo de operación, o el ***VWAP*** (*Volume Weighted Average Price*) donde se divide una orden ponderada por el volumen a lo largo del tiempo de operación; representan la primer generación de este tipo de algoritmos. Aunque están destinados a minimizar los costos de impacto, su enfoque principal son sus respectivos *benchmarks*. Se basan en un horario fijo, por lo que su objetivo es completar la ejecución de la orden en un tiempo establecido sin importar las condiciones de mercado.

La progresión natural de estos enfoques estáticos ha sido la adopción de métodos más dinámicos, resultando en un cambio gradual hacia algoritmos más oportunistas de *trading*.

Un ejemplo son los algoritmos ***POV*** (*Percentage of volume*) basados en el volumen actual del mercado y que, como su nombre indica, operan porcentajes del volumen actual. Estos algoritmos evolucionan en algoritmos de mínimo impacto que toman un enfoque sigiloso, apuntando a no generar ningún impacto de mercado.

3.2.2 Algoritmos *Cost-driven*

Buscan reducir los costos generales de una operación. Por lo tanto, necesitan tener en cuenta el impacto de mercado, riesgo del momento de ejecución (*timing risk*) e incluso factores como tendencias en precios.

Para reducir los costos de transacción, se debe encontrar un balance entre el impacto de mercado y el riesgo del momento de ejecución. Esto fue llamado como el **dilema del *trader***, por Kissel and Glantz (2003): operar muy agresivamente resultaba en un impacto de mercado considerable, mientras que operar de manera muy pasiva resultaba en aumentar el riesgo del momento de ejecución.

Los primeros algoritmos de costo de transacción evolucionaron partiendo de algoritmos de impacto de mercado, al incluir nuevas consideraciones, como el momento de ejecución. Entre algunos ejemplos encontramos el ***Implementation shortfall***, que busca minimizar la diferencia entre el precio promedio operado y el *benchmark* a utilizar; el ***Adaptative Shortfall***, cuya idea es similar aunque de manera más oportunista según tendencias en precios y liquidez del mercado; y el ***Market-on-close***, cuya meta es operar similar al precio de cierre del mercado.

Si se quiere utilizar la estrategia del presente trabajo para órdenes que puedan ocasionar impactos importantes en el mercado, en caso de operar cerca del horario de cierre, se puede recurrir al algoritmo de ejecución de *Market-on-close* para determinar los mejores momentos para operar una gran cantidad de títulos y así tener un precio similar al del cierre, sin incurrir en un impacto significativo en el mercado.[1]

3.2.3 Algoritmos oportunistas

Los algoritmos oportunistas toman ventaja cuando las condiciones del mercado son favorables buscando generalmente precio, liquidez o involucrados en *pair/spread trading*.

Los algoritmos en línea con el precio, generalmente derivaron de algoritmos de impacto de mercado como el **VWAP** o **POV**, agregando sensibilidad al precio, permitiendo modificarse y ser más agresivos cuando el precio es favorable para su posición.

Por otro lado, ***Pair trading*** es una estrategia de mercado-neutral, más conocida como *market neutral strategy*, por lo que el riesgo es de menor preocupación. En estos casos, la razón entre activos o el *spread*, es la base para la toma de decisiones.

3.3 Arbitraje Estadístico y *Pair Trading*

En finanzas, el arbitraje estadístico es una clase de estrategias de *trading* donde se utilizan modelos de ***reversión a la media***[2]. El caso de ***pair trading*** es un tipo de estrategia de arbitraje estadístico. Al ser una estrategia de tipo *market-neutral*, involucra comprar un activo mientras se vende otro simultáneamente. Se llama estrategia de mercado neutral porque el riesgo de cada activo debe cubrirse o cancelarse mutuamente. Por ejemplo, si el mercado va a la baja, la pérdida de la posición larga —posición de compra— se debe cubrir con la ganancia de la posición corta —posición de venta—.

Para que esta estrategia funcione, los precios de ambos activos a operar deben estar "suficientemente correlacionados"[3], para que ambos suban y bajen en la misma proporción y realmente sea una estrategia de mercado neutral.

La estrategia de *pair trading*, siguiendo el modelo de arbitraje estadístico, consiste en encontrar la valuación relativa entre activos. Se asume que el *spread* o la razón entre dos activos que estén altamente correlacionados, generará oscilaciones alrededor de su media. Para el caso de bonos se utilizan *spreads*, que es la diferencia entre dos tasas, y para el caso de acciones, se utilizan razones o *ratios*.

[1] Más información sobre el algoritmo *Market-on-close* en "*Algorithmic Trading & DMA - An introduction to direct access trading strategies*", de Barry Johnson. Ver Bibliografía.

[2] En finanzas, "reversión a la media" es un término para la suposición de que el precio de un activo va a tender a regresar al precio promedio con el tiempo.

[3] "*Algorithmic Trading & DMA ...*", capítulo 5, sección 5.6, pág. 150.

Cuando una razón entre dos activos altamente correlacionados se desvía de su media, existe una oportunidad de *trading*. Comúnmente se utilizan dos desviaciones estándar de la media para las señales de entrada a una operación; de esa forma, si la razón está más alejada de la media que dos desviaciones estándar, se realiza la compra y venta de activos, cerrando la operación cuando la razón vuela a la media.

Para el caso del presente trabajo, el algoritmo consiste en medir la razón entre un portafolio que replique el índice, creado utilizando modelos de *machine learning*, y un ETF común para el mercado que hace réplica del *S&P 500*, el *SPY US Equity*.

Lo primero es construir el portafolio. Una vez que se tiene el portafolio, lo siguiente es analizar sus métricas para comprobar que es un ETF que efectivamente replica el índice. Posteriormente, comprobar que existe una correlación entre dicho portafolio y el ETF del mercado.

Luego, se analiza su razón y su comportamiento en el tiempo, para determinar si existen opciones de *trading* por medio de la estrategia de *pair-trading*. Finalmente, se implementa la estrategia y se analizan los resultados.

Parte II

Aprendizaje estadístico y modelos de *machine learning*

Capítulo 4

Aprendizaje estadístico

4.1 Introducción

De manera general, supongamos que se observa una respuesta numérica Y y una cantidad p de diferentes variables predictivas, $X_1, X_2, ...X_p$. Se asume que debe existir una cierta relación entre Y y $X = (X_1, X_2, ...X_p)$, que puede ser descrita en la forma general:

$$Y = f(X) + \epsilon \tag{4.1}$$

Aquí, f es una función fija, pero desconocida, de $X_1, ..., X_p$ y ϵ es un *término de error*, el cual es independiente de las variables del vector **X** y con media cero. De esta forma, f representa la información sistemática que **X** provee sobre **Y**, que también es variable aleatoria. Sin embargo, esa función f que conecta la variable de entrada con la variable de salida, es generalmente desconocida.

4.1.1 ¿Por qué estimar *f*?

En esencia, el aprendizaje estadístico se refiere a un conjunto de enfoques para poder estimar f. Las principales razones para querer estimar f son: *predicción* e *inferencia*.

Predicción

En muchas situaciones, el conjunto de *inputs* X está disponible, pero el *output* Y no se puede obtener fácilmente. En estos casos, dado que el término del error tiene media o, se puede predecir Y usando:

$$\hat{Y} = \hat{f}(X), \tag{4.2}$$

donde $\hat{f}$ representa el estimado para f, y $\hat{Y}$ representa la predicción de la variable objetivo Y. Aquí, $\hat{f}$ es tratada como una *caja negra* de manera frecuente, en el sentido de que no es de interés el conocer la forma exacta de $\hat{f}$, siempre que se produzcan predicciones precisas para Y.

La precisión con la que $\hat{Y}$ predice a Y depende de dos cantidades que se conocen como el *error reducible* y el *error irreducible*. En general, $\hat{f}$ no es un estimador perfecto para f, y esto introduce un error. Este error es *reducible* porque se puede mejorar la precisión de $\hat{f}$ al utilizar el modelo más apropiado de aprendizaje estadístico para estimar f.

Sin embargo, incluso si fuera posible encontrar una forma perfecta para estimar f, tal que la respuesta estimada fuera $\hat{Y} = f(X)$, la predicción aún tendría un error. Esto debido a que Y es también función de ϵ, el cual, por definición, no se puede predecir utilizando X, pues son independientes. Por tanto, la variabilidad asociada a ϵ también afecta la precisión de las predicciones. Esto es conocido como el error *irreducible*, pues, sin importar qué tan bien se estime f, no se puede reducir el error dado por ϵ.

Se considera un estimador dado $\hat{f}$ y un conjunto de variables predictivas X, que produzca la predicción tal que $\hat{Y} = \hat{f}(X)$. Se asume por un momento que $\hat{f}$ y X son fijas. Entonces, es fácil mostrar que:

$$\begin{aligned} E\ (Y - \hat{Y})^2 &= E\ [f(X) + \epsilon - \hat{f}(X)]^2 \\ &= \underbrace{[f(X) - \hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{Var(\epsilon)}_{\text{Irreducible}} \end{aligned} \tag{4.3}$$

donde $E\ (Y - \hat{Y})^2$ representa la esperanza del cuadrado de la diferencia entre el valor de predicción y el valor actual de Y, y Var(ϵ) representa la varianza asociada con el término de error ϵ.

Inferencia

En algunos casos, es de interés entender la forma en que Y se ve afectada por los cambios en $X_1, ...X_p$. Se quiere entender la relación entre X y Y, o más específicamente, cómo cambia Y como función de $X_1, ...X_p$, más no necesariamente realizar una predicción para Y. En este caso, no se puede tratar a f como una caja negra, pues se necesita saber su forma exacta.

Algunas preguntas que podrían ser de interés para este tipo de análisis son:

- *¿Qué variables predictivas están asociadas a la variable de respuesta?* Identificar las pocas y *más importantes* variables entre un gran conjunto de posibles variables puede ser extremadamente útil.
- *¿Cuál es la relación entre la variable de respuesta y cada una de las variables predictivas?* Algunas variables podrían tener una relación positiva con Y y otras, una relación negativa.

Dependiendo si el objetivo último es la predicción, la inferencia o una combinación de ambas, diferentes métodos para estimar f pueden ser apropiados. Por ejemplo, un *modelo lineal* puede ser simple e interpretable para inferencia, pero podría no producir predicciones tan precisas como otra clase de modelos. En contraste, algunos enfoques altamente no lineales podrían proporcionar predicciones bastante precisas para Y, pero a costa de un modelo mucho menos interpretable para el cual sería muy complejo realizar una inferencia.

4.1.2 ¿Cómo estimar *f*?

Se asume que se tiene un conjunto de n diferentes datos conocidos por cada variable. A estos datos se les llaman *datos de entrenamiento* porque se utilizan estas observaciones para entrenar, o enseñar, al modelo cómo estimar *f*. Sea x_{ij} el valor de la observación *i-ésima* para la variable *j-ésima*, donde $i = 1, 2, ...n$ y $j = 1, 2, ...p$. De la misma forma, sea y_i la variable de respuesta para la observación *i-ésima*. Entonces, la información de entrenamiento consiste en $\{(\mathbf{X_1}, y_1), (\mathbf{X_2}, y_2), ..., (\mathbf{X_n}, y_n)\}$ donde $\mathbf{X_i} = (x_{i1}, x_{i2}, ..., x_{ip})$.

Se quiere encontrar una función $\hat{f}$ tal que $y \approx \hat{f}(X)$ para cualquier observación $(\mathbf{X}, \mathbf{y})$. De manera general, los métodos de aprendizaje estadístico para esta tarea pueden ser clasificados como *paramétricos* o *no paramétricos*.

Métodos paramétricos

Involucran un enfoque basado en el modelo de dos pasos:

1. Primero, se asume la forma funcional de *f*. Una vez que se asume la forma de *f*, el problema de estimarla se simplifica bastante, ya que se reduce a estimar los coeficientes de la función.
2. Una vez que el modelo ha sido seleccionado, se requiere un proceso que utilice los datos de entrenamiento para *ajustar* o *entrenar* el modelo. La forma más común de hacerlo es utilizando *mínimos cuadrados*.

El enfoque basado en el modelo se conoce como paramétrico, pues reduce el problema de estimar *f* a estimar un conjunto de parámetros de una función asumida previamente. La mayor desventaja del enfoque paramétrico es que el modelo que se tome para la función, usualmente no coincide con la verdadera y desconocida forma de *f*.

Métodos no paramétricos

Los métodos no paramétricos no hacen supuestos explícitos sobre la forma funcional de *f*. En vez de eso, buscan estimar una función f que se acerque lo más posible a los datos sin ser muy rígida o muy variable. Estos enfoques pueden tener una gran ventaja sobre los enfoques paramétricos pues, al no suponer una forma particular para *f*, tienen el potencial de ajustar mejor los datos con un amplio rango de posibles formas para *f*.

Sin embargo, los modelos no paramétricos sufren de una desventaja importante: dado que no reducen el problema de estimar f a la estimación de sus parámetros, se requiere un gran número de observaciones para poder estimar f de manera precisa.

4.1.3 La compensación entre la precisión e interpretabilidad

De todos los métodos posibles, algunos son menos flexibles o más restrictivos, en el sentido de que, pueden producir sólo un rango relativamente pequeño de formas para estimar *f*. Por ejemplo, una regresión lineal es un enfoque relativamente inflexible, pues sólo puede generar funciones lineales.

Otros métodos, como los de interpolación multivariable (como los *splines* y el método de vecinos más cercanos) son considerablemente más flexibles porque pueden generar un rango mucho mayor de posibles formas para estimar f.

En este punto, se puede realizar la pregunta de *¿por qué alguien escogería un método mucho más restrictivo, en lugar de uno mucho más flexible?* Si el interés está principalmente en la inferencia, entonces los modelos más restrictivos son mucho más interpretables. En contraste, los enfoques muy flexibles, como los *splines* o los *boosting methods*, pueden llevar a estimaciones tan complicadas para f que sería difícil entender cómo cualquier variable predictiva individual está asociada con la variable de respuesta.

Cuando el objetivo es hacer inferencia, hay ventajas en usar métodos de aprendizaje estadístico simples y, relativamente, inflexibles. En otros escenarios, donde sólo se esté interesado en predicción y la interpretabilidad del modelo simplemente no sea de interés, se puede esperar que lo mejor sería utilizar el modelo más flexible que se tenga a disposición. Sin embargo, es importante notar que **ese no siempre es el caso**. Esto se debe a que, el uso de modelos flexibles, puede llevar a un **sobreajuste** potencial, al querer tocar todos los puntos de la muestra, provocando estimaciones erróneas.

4.1.4 Aprendizaje supervisado y no supervisado

La mayoría de los problemas de aprendizaje estadístico pueden clasificarse en una de las siguientes dos categorías: aprendizaje supervisado y aprendizaje no supervisado.

En el **aprendizaje supervisado**, para cada observación de las variables $x_i, i = 1, ..., n$, existe una variable de respuesta asociada y_i. Se quiere ajustar un modelo que relacione la respuesta con las variables de predicción, con la mira en predecir la variable de respuesta para un conjunto de observaciones futuras (predicción), o bien, para entender la relación entre las variables de predicción y la variable de respuesta (inferencia).

En contraste, el **aprendizaje no supervisado** describe una situación en la que, para cada observación $i = 1, ..., n$, se observa un vector de variables de x_i, pero no existe una respuesta asociada y_i. En estos casos, se puede buscar entender las relaciones entre las variables o entre observaciones de una misma variable. Una herramienta de aprendizaje estadístico que se utiliza en este tipo de problemas es un análisis de *clusters*. El objetivo es averiguar, con base en las observaciones $x_1, ..., x_n$, si se pueden obtener distintos grupos; y determinar si nuevas observaciones se pueden categorizar en esos grupos. Esto puede ser útil ya que los grupos podrían diferir entre ellos con respecto a una característica de interés.

Problemas de regresión vs. problemas de clasificación

A grandes rasgos, podemos dividir las variables en *categóricas*, formadas por K clases, o *numéricas*, tomando valores en los números reales, R.

Se hace referencia a problemas donde la respuesta es numérica, como problemas de *regresión*, cuando se tiene una función $f : R -> R^m$ y a problemas de *clasificación* cuando se involucran respuestas categóricas, de funciones $f : R -> \{C_K\}_k$.

4.2 Evaluación de la precisión del modelo

Dado que no existe un modelo que sea mejor que cualquier otro para todos los posibles conjuntos de datos, entonces una tarea importante es decidir cuál método produce los mejores resultados para un conjunto de datos a estudiar.

4.2.1 Medición de la calidad del ajuste

Para poder evaluar el rendimiento de un método de aprendizaje estadístico en un conjunto de datos, se debe tener una manera de medir qué tan bien coinciden las predicciones con los datos observados. Esto se refiere a la necesidad de cuantificar la medida en la cual el valor de respuesta predicho para una observación dada está cerca del valor de respuesta real para esa observación.

En el ajuste de regresión, la medida más comúnmente usada es el *error cuadrático medio*, ECM, dado por:

$$ECM = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{4.4}$$

donde $\hat{f}(x_i)$ es la predicción que $\hat{f}$ da para la i-ésima observación. El error cuadrático medio será pequeño con relación a los datos, si las respuestas predichas están cercanas a las respuestas reales; y será grande, respecto a los datos, cuando las respuestas predichas y las reales, difieran sustancialmente.

Se ajusta un método de aprendizaje estadístico a las observaciones de entrenamiento $\{(x_1, y_1),\ (x_2, y_2),\ ...,\ (x_n, y_n)\}$, y se obtiene un estimado para $\hat{f}$. Posteriormente se pueden calcular $\hat{f}(x_1), \hat{f}(x_2), ..., \hat{f}(x_n)$. Si los resultados son aproximadamente iguales a $y_1, y_2, ..., y_n$, entonces el *error cuadrático medio del entrenamiento* es bajo. Sin embargo, el verdadero interés no está en si $\hat{f}(x_i) \approx y_i$, sino que, se busca que, para una serie de datos que no se encontraba en los datos de entrenamiento, es decir, una observación previamente sin observar (x_0, y_0), $\hat{f}(x_0)$ se aproxime lo más posible a y_0. Se escoge el método que tenga el menor *error cuadrático medio de prueba*, y no tanto el menor error cuadrático medio de entrenamiento. En otras palabras, si se tiene un gran número de observaciones fuera de los datos de entrenamiento, se calcula:

$$Promedio\ (\hat{f}(x_0)\ - y_0)^2 \tag{4.5}$$

que es el error cuadrático medio para las observaciones de prueba (x_0, y_0). Se selecciona el modelo para el cual el promedio de esta cantidad, el ECM de prueba, es lo menor posible.

Se debe mencionar que no siempre el método que minimiza el ECM de entrenamiento, minimiza también el ECM de prueba. Al ir ajustando la suavidad y flexibilidad de una curva, se pueden producir diferentes ajustes para los datos. Se puede obtener una curva para la cual se tenga el ECM de entrenamiento como una función de la flexibilidad o, más formalmente, de los *grados de libertad*. Los grados de libertad son la cantidad que resume la flexibilidad de la curva. Una curva más restrictiva tiene menos grados de libertad que una curva más ondulada. **El ECM de entrenamiento disminuye monotónicamente conforme los grados de libertad aumentan**.

Así, si se quiere tener un ECM de entrenamiento mínimo, se pueden usar muchos grados de libertad y tener una curva extremadamente volátil que pase por todos y cada uno de los puntos del conjunto de entrenamiento. Sin embargo, como se mencionó anteriormente, esto provocaría un sobreajuste y no serviría para realizar proyecciones sobre esos datos, debido a la elevada volatilidad presente en el modelo.

Es por eso que es más importante el ECM de prueba, pues el modelo no puede pasar por los puntos que aún no se conocen o que no se usan para el entrenamiento. Similar al ECM de entrenamiento, el ECM de prueba disminuye conforme se incrementan los grados de libertad. Sin embargo, en un punto, el ECM de prueba llega a un punto de inflexión y comienza a incrementar de nuevo, dando como resultado un ECM de prueba elevado.

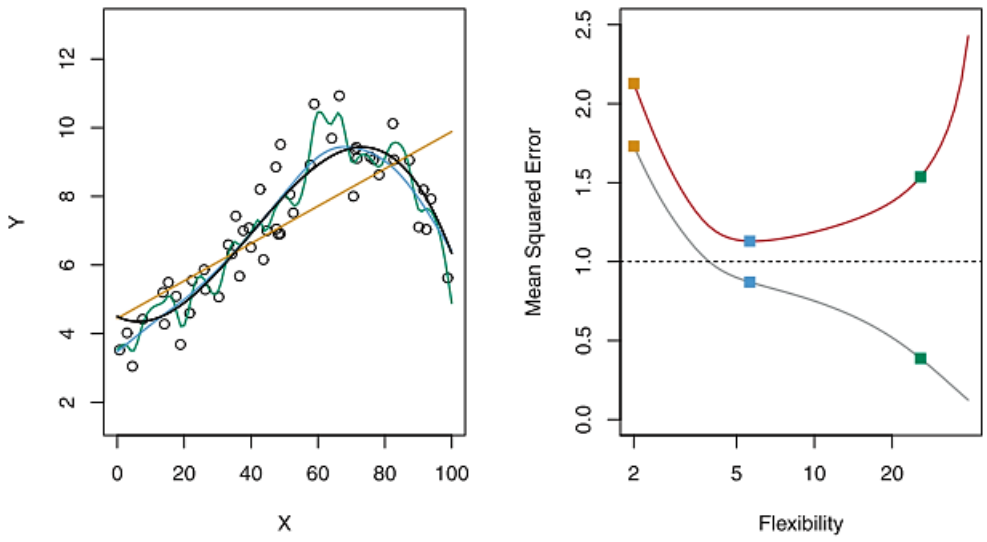


Figura 4.1. *Izq: Se tienen datos simulados de una función f, en negro. Se muestran 3 estimaciones para f: Una regresión lineal (línea naranja) y dos ajustes de splines suavizados (curvas verde y azul). Der: Se muestra el ECM de entrenamiento (curva gris), el ECM de prueba (curva roja) y el mínimo posible de los ECM de prueba para todos los métodos (línea punteada). Los cuadros representan los puntos para los ECM de entrenamiento y de prueba de los tres modelos utilizados en la imagen del lado izquierdo.*

Esta propiedad en la que el ECM de entrenamiento disminuye de manera monótona conforme aumentan los grados de libertad, y el ECM de prueba tiene una *forma de U*, tal como se ve en la imagen anterior[1], es fundamental en el aprendizaje estadístico que se mantiene sin importar el conjunto de datos ni el método que se utilice.

Cuando un modelo tiene un ECM de entrenamiento bajo, pero un ECM de prueba elevado, entonces se dice que se están ***sobreajustando*** los datos. Eso sucede porque el procedimiento trabaja para encontrar tendencias pero, al sobreajustar los datos, se pueden suponer patrones que son ocasionados por ruido o de manera aleatoria y que no necesariamente son propiedades de la función desconocida *f*. El ECM de prueba es elevado porque esos patrones de los datos de entrenamiento no existen realmente en el conjunto de datos de prueba. El sobreajuste se refiere específicamente a cuando el ECM de prueba disminuye al utilizar un modelo menos flexible.

[1]La imagen y el subtítulo son tomadas del libro: *"Algorithmic Trading & DMA - An introduction to direct access trading strategies"*, de Barry Johnson (2010).

En la práctica, se puede emplear una gran variedad de enfoques para estimar este punto mínimo. Un método importante es el de validación cruzada (*cross validation*), que es un método para estimar el ECM de prueba utilizando los datos de entrenamiento.

4.2.2 Compensación entre sesgo y varianza

La forma de U que se observa en la curva del ECM de prueba, equivale al resultado de dos propiedades en competencia de los métodos de aprendizaje estadístico. Es posible mostrar que el ECM de prueba esperado, para un valor dado x_0, puede descomponerse en la suma de tres cantidades fundamentales[2]: la *varianza* de $\hat{f}(x_0)$, el *sesgo cuadrado* de $\hat{f}(x_0)$ y la varianza del término de error ϵ. Esto es:

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = Var(\hat{f}(x_0)) + \left[Sesgo\left(\hat{f}(x_0)\right)\right]^2 + Var(\epsilon) \tag{4.6}$$

Donde $Sesgo = E[\hat{f}(x_0)] - f(x_0)$.

Se puede observar que para poder minimizar el error de prueba esperado, se debe seleccionar un método de aprendizaje estadístico que, de manera simultánea, alcance un bajo sesgo y una baja varianza. También se observa que, tanto la varianza como el sesgo al cuadrado, son cantidades no negativas, por lo que el ECM de prueba nunca será menor que Var(ϵ), el error irreducible.

La **varianza** se refiere a la cantidad en la que $\hat{f}$ cambiaría si se estima usando un conjunto de datos de entrenamiento distintos. Dado que se utilizan los datos de entrenamiento para estimar $\hat{f}$, distintos conjuntos de datos de entrenamiento resultarían en distintas funciones $\hat{f}$. Idealmente, el estimado para f no debería variar demasiado entre conjuntos de entrenamiento. Sin embargo, si el método tiene una elevada varianza, entonces pequeños cambios en el conjunto de entrenamiento, podrían resultar en grandes cambios para $\hat{f}$. En general, los modelos más flexibles presentan mayor varianza.

Por otro lado, el **sesgo** se refiere al error que se introduce por aproximar la función a un problema real. Por ejemplo, una regresión lineal asume que existe una relación lineal entre Y y $X_1, X_2, ...X_p$. Es poco probable que cualquier problema real tenga una relación tan simple como la relación lineal, y por tanto, al utilizar ese modelo, se introduciría un sesgo al estimar la función f. Generalmente, los métodos más flexibles tienen menor sesgo.

Al usar métodos más flexibles, incrementa la varianza y disminuye el sesgo. La tasa relativa de cambio entre estas dos cantidades es la que determina si el ECM de prueba incrementa o disminuye. Inicialmente, el sesgo tiende a decrecer más rápido que lo que incrementa la varianza. Consecuentemente, el ECM de prueba esperado, disminuirá. Sin embargo, en algún punto, incrementar la flexibilidad tendría poco impacto en el sesgo, pero comenzaría a tener un impacto significativo en la varianza. Cuando esto ocurre, el ECM de prueba incrementa.

[2]Se puede demostrar matemáticamente, pero está fuera del alcance de este trabajo

Esta relación entre el sesgo y la varianza es lo que se conoce como la compensación sesgo-varianza (en inglés *bias-variance trade-off*). El reto está en encontrar un método para el cual los valores de la varianza y el sesgo al cuadrado, se minimicen.

En una situación de la vida real, en el que la función f no se puede observar, es imposible calcular el ECM de prueba, el sesgo, o la varianza para un método de aprendizaje estadístico. Por ello, se utilizan métodos como el de validación cruzada para estimar el ECM de prueba utilizando los datos de entrenamiento.

Además de los métodos vistos, existe literatura sobre métodos de clasificación, aunque están fuera del enfoque de este trabajo. En ambos conjuntos, regresión y clasificación, escoger el nivel adecuado de flexibilidad es crucial para el éxito de cualquier método de aprendizaje estadístico.

Capítulo 5

Modelos paramétricos

5.1 Introducción

Como se mencionó en el capítulo anterior, existe una gran cantidad de modelos de aprendizaje estadístico, o *machine learning*, que se podrían utilizar para el manejo óptimo de grandes cantidades de datos, así como de tendencias y características. A continuación, se presentan algunos modelos de los muchos posibles que pueden aplicarse al mercado financiero. Se dividen en modelos paramétricos (en el presente capítulo) y no paramétricos (en el siguiente capítulo). Para el presente capítulo se utilizan **modelos de regresión lineal simple y múltiple**, y una extensión a **modelos de regresión polinómica**.

Retomando el objetivo del trabajo de investigación, se quiere construir un algoritmo en el cual se conocen sus valores numéricos, tal que a las variables de entrada les corresponde una variable de salida. Dado que se quiere llegar a un valor numérico, y no categórico, se utilizan modelos de regresión, y no de clasificación[1]. Por tanto, para fines del presente estudio, se utilizarán modelos de aprendizaje estadístico **supervisado** de **regresión**.

Se tomará un modelo paramétrico, en el cuál se asume la forma de la función f. Para el caso de la regresión lineal, se asume una relación lineal entre las variables de entrada y las de salida. Este supuesto se va volviendo más flexible conforme se cambia a regresión polinómica y se aumentan los grados de libertad.

En capítulos subsecuentes, se realiza la construcción del modelo y se compara su efectividad en cuanto a rendimientos contra un portafolio base que funciona como *benchmark*.

[1]También es posible utilizar modelos categóricos en finanzas. Un ejemplo para mercados puede ser dividir un estimado en si el precio sube (1) o baja (0). Sin embargo, están fuera del enfoque del presente trabajo.

5.2 Regresión lineal

El método de regresión es un modelo paramétrico pues asume que la función f, que relaciona variables predictivas con variables a predecir, tiene una forma determinada. Para el caso de la regresión lineal, asumiremos que la función es, precisamente, lineal, y por tanto, la relación entre variables de entrada y de salida, se asume lineal.

La regresión lineal es un modelo que permite estimar los valores de una variable numérica como una función lineal de las variables de entrada o predictores. De esta forma, la función lineal se expresa de la siguiente manera:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{5.1}$$

donde:

- $\hat{y}$ es la estimación del modelo para la variable y
- x_i las variables de entrada o predictores
- β_i las ponderaciones o pesos de las variables
- β_0 la estimación del modelo cuando todas las entradas son nulas (sesgo)

Se deben obtener los valores de β_i de tal forma que $\hat{y}$ sea lo más parecido a y. Para esto, se utiliza una función de costo, comúnmente la función de mínimos cuadrados, la cual se busca minimizar.[2] Estos coeficientes pueden obtenerse por un método analítico o utilizando un gradiente descendiente.

De forma gráfica, una regresión lineal simple se representaría de la siguiente manera:

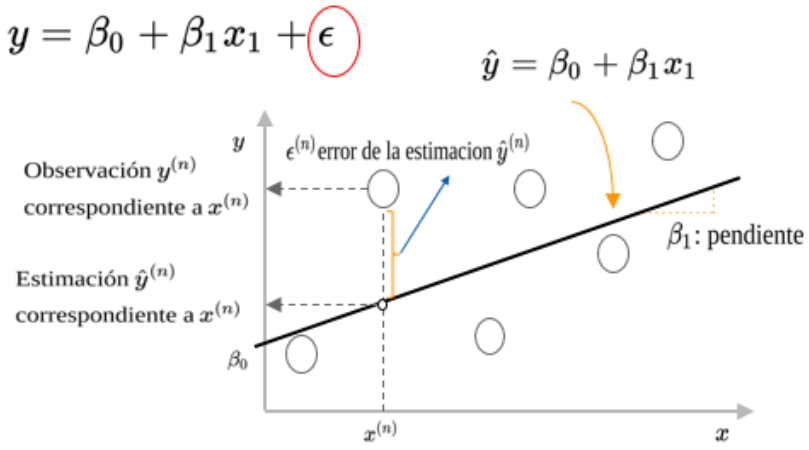


Figura 5.1. *Regresión lineal de ejemplo.*

[2] Se puede ver la fórmula en el capítulo 2 del presente trabajo, subsección 2.2, fórmula 2.4.

5.2.1 Obtención de coeficientes de forma analítica.

Se utiliza el método de mínimos cuadrados, que es el método que utiliza el modelo de Python del presente trabajo. Es un método basado en la teoría de probabilidad y permite obtener los coeficientes de forma directa. Sólo es válido cuando **no hay correlación entre las variables de entrada**. La fórmula es la siguiente:

$$\beta_i = \frac{\sum_{n=1}^{N}(x_i^{(n)} - \bar{x}_i)(y^{(n)} - \bar{y})}{\sum_{n=1}^{N}(x_i^{(n)} - \bar{x}_i)^2} \tag{5.2}$$

$$\beta_0 = \bar{y} - \sum_{i=1}^{D}\beta_i \bar{x}_i \tag{5.3}$$

5.2.2 Obtención de coeficientes por gradiente descendiente.

Es un método iterativo que permite reducir la función de costo, por ejemplo, el ECM en cada iteración. Los pasos son:

1. Se inicializa el valor de los β_i de forma aleatoria y se determina un nivel para el "salto" del gradiente η.
2. Repetir:
   (a) Se calcula el $\hat{y}$ con los coeficientes β_i actuales.
   (b) Se actualizan los coeficientes β_i de la siguiente manera:

$$\beta_i = \beta_i - \eta \sum_{j=1}^{N}(\hat{y}^{(n)} - y^{(n)})x_i^{(n)} \tag{5.4}$$

$$\beta_0 = \beta_0 - \eta \sum_{j=1}^{N}(\hat{y}^{(n)} - y^{(n)}) \tag{5.5}$$

donde η es un híper-parámetro que controla el tamaño del salto que hará que cambie β_i por cada iteración que se realice. Se suele utilizar $\eta = 0.1$

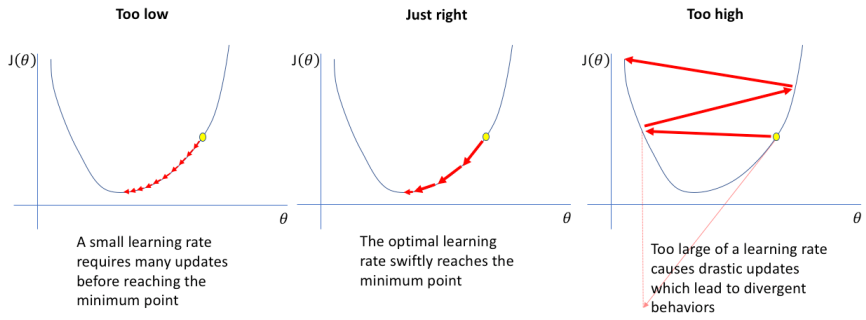


Figura 5.2. *Ejemplo del comportamiento del gradiente descendiente, con diferentes tasas de aprendizaje η.*

Normalmente, se puede utilizar gradiente descendiente cuando el costo computacional de calcular un valor de manera analítica —utilizando error cuadrático medio, u otro método— es muy elevado; o cuando se tienen múltiples variables objetivo.

5.2.3 Suposiciones para el modelo de regresión lineal.

La regresión lineal es un modelo de regresión que ofrece ciertas ventajas:

- Interpretabilidad de los resultados.
- Facilidad de uso e implementación.
- Poco coste computacional.

Por otro lado, para su correcta implementación, se debe cumplir con los siguientes supuestos:

- La relación entre los predictores y la salida debe ser lineal o casi lineal.
- La varianza de los términos de error debe ser constante (heterocedasticidad).
- Es muy sensible a *outliers* y a la colinealidad entre los predictores.
- Correlación entre los errores de la muestra.
- Deben ser variables independientes, idénticamente distribuidas (i.i.d.)

Estos supuestos pueden presentar limitaciones en cuanto al número de problemas reales que se pueden resolver por medio de la regresión lineal. Para trabajar esas limitaciones, se pueden utilizar las siguientes recomendaciones o extensiones al modelo:

1. Para evitar que sólo se considere la relación lineal entre predictores y la salida, se pueden considerar modelos de regresión polinómica. En estos casos, conforme se aumenta el grado de la regresión, se obtiene mayor flexibilidad (aunque también un mayor riesgo al sobreajuste).

2. Específicamente para problemas financieros, se asume que, en condiciones de mercado normales, la varianza de los términos de error es constante.

3. Para tener variables i.i.d. se pueden aplicar transformaciones a los datos de entrada, de manera que se elimine la correlación entre ellos. Para datos financieros, se utilizan generalmente los rendimientos logarítmicos para lograr este punto y no los precios directamente.

4. El caso de los *outliers* se puede tratar desde diferentes frentes. Por un lado, analizar el dato particular y verificar que no es un error en el dato. En segundo lugar, se pueden considerar otras variables que puedan explicar ese dato y que pueda ser importante para los predictores.

5. Para el problema de colinealidad de predictores se puede eliminar un predictor que tenga una alta correlación con otro. Para verificar estas colinealidades se puede construir una matriz de correlación entre los predictores, y limpiar los datos antes de realizar las pruebas dentro del modelo, o utilizar algún método de regularización.

5.2.4 Precisión del modelo.

Lo siguiente es estudiar la relación entre el valor estimado y el valor real, para determinar la precisión del modelo. Para ello, podemos hacer uso de alguna de las siguientes funciones:

- El error residual estándar (*RSE* por *residual standard error* en inglés)
- El coeficiente de correlación *r*
- El estadístico R^2

El **RSE** indica un valor absoluto del fallo en el ajuste del modelo, sin embargo, está calculado sobre las unidades de la variable de salida Y, por lo que no siempre se puede saber qué grado constituye un buen o mal RSE. Por otro lado, el **coeficiente de correlación** nos indica únicamente el grado de correlación **lineal** que existe entre las variables, por lo que si se quiere cambiar de una regresión lineal a una regresión polinomial, no sería posible utilizar ese coeficiente.

Finalmente, la $\mathbf{R^2}$ toma una forma de *proporción*, la proporción de la varianza explicada, tomando siempre valores menores a 1. Además es independiente de la escala de las variables y se puede probar que $R^2 = r^2$ en un espacio para una regresión lineal simple.

El estadístico R^2 calcula como:

$$R^2 \;=\; 1 \;-\; \frac{RSS}{TSS} \;=\; 1 \;-\; \frac{\sum(y_i - \hat{y}i)^2}{\sum(y_i - \bar{y})^2} \tag{5.6}$$

RSS es la suma de residuales al cuadrado (*residual sum of squares* en inglés), que sería la suma de las diferencias al cuadrado entre nuestra variable real de salida y la variable estimada de salida. TSS es la suma total de cuadrados entre nuestro valor real de salida y la media, (*total sum of squares* en inglés).

Visto de otra forma, TSS mide el total de la varianza en la respuesta Y, y por tanto se puede entender como la variabilidad inherente en la respuesta antes de que la regresión sea aplicada. Por otro lado, RSS mide la variabilidad que se queda sin explicar luego de aplicar la regresión. Es por eso que R^2 mide la proporción de la variabilidad en Y que puede ser explicada por X.

Un R^2 cercano a 1 indica que existe una gran proporción de la variabilidad que puede ser explicada por la regresión, y existiría una menor proporción del error irreducible, por lo que es preferible a un valor R^2 cercano a 1, el cual indica que la regresión explica gran parte de la variabilidad de la respuesta.

Sin embargo, R^2 no siempre es el mejor método de comprobación de un modelo. Por ejemplo, para regresión lineal múltiple, la R^2 sigue aumentando conforme se aumentan variables, dando un falso sentido de correlación, y por eso se utiliza la R^2 ajustada, la cual se ajusta según el número de variables y observaciones. En capítulos siguientes se ven otros métodos para la comprobación de la precisión del modelo.

Hasta el momento, se puede aplicar la misma teoría para una regresión lineal simple y una regresión lineal múltiple. Es importante señalar que, en una regresión lineal simple puede existir un valor R^2 cercano a 1, el cual indique que existe una correlación lineal entre las variables, cuando en realidad lo que suceda es que existe una tercer variable (o más) que expliquen el motivo de la correlación entre las variables estudiadas.

De aquí se concluye que *correlación no implica causalidad*, es decir, el que dos variables estén correlacionadas no implica que el movimiento en una variable sea necesariamente el factor por el cuál se mueve la otra variable. Para eso, se utilizan más variables, y se busca eliminar variables entre las que existe colinealidad, pues estarían explicando la misma varianza de nuestra variable objetivo, generando ruido innecesario en el modelo y no permitiendo encontrar las variables realmente relevantes. A esto se le conoce como **dependencia condicional**.[3]

En capítulos siguientes se sientan las bases para el modelo no paramétrico y posteriormente se analizan otros métodos para estudiar la precisión de un modelo.

[3]Es importante señalar que el presente trabajo no se enfoca en encontrar las causas entre los movimientos de una variable y otra, sino en las relaciones entre variables para aprovecharlas y tomar mejores decisiones de inversión.

Capítulo 6

Modelos no paramétricos

6.1 Introducción

A diferencia de los métodos vistos en el capítulo anterior, los métodos no paramétricos no asumen una distribución de probabilidad, como en el caso de la regresión lineal donde se asume que el error se distribuye normal con varianza constante. Otra diferencia es que los modelos paramétricos tienen un conjunto finito de parámetros, mientras que los modelos no paramétricos pueden tener, potencialmente, un número de parámetros infinito.

En este capítulo se revisan modelos de **árboles de decisión**. El modelo a implementar en el trabajo es un modelo de ***random forest***, específicamente aquellos utilizados para regresión, es decir, estimación de variables continuas.

Los métodos basados en árboles involucran la segmentación del espacio de predictores en un cierto número de regiones. Para poder realizar predicciones, típicamente se utiliza la media (para árboles de regresión), o la moda (para árboles de clasificación), de las observaciones del conjunto de entrenamiento en la región correspondiente; donde el valor de la predicción es la media, o la moda, de las predicciones, según corresponda. Existe literatura que afirma que los árboles de decisión no producen resultados competitivos contra otros modelos (Ver [1]), por lo que resulta necesario ampliar el análisis al estudio de los *random forest*, el cual consiste en producir una X cantidad de árboles que produzcan X cantidad de respuestas, y tomar como respuesta final la media (para problemas de regresión) o la moda (para problemas de clasificación), utilizando un subconjunto de covariables. Esto puede resultar en un incremento dramático en la precisión de las predicciones, ante una cierta pérdida de interpretabilidad.

Los datos que se utilizan para cada árbol, son subconjuntos distintos de los datos de entrenamiento. Dado que los datos están estructurados como series de tiempo, se deben tomar bloques de manera que no se rompa la secuencia entre los mismos. Para ello, recorremos a *Cross Validation* y *bootstrapping* por bloques.

6.2 Métodos con árboles de decisión

Un árbol de decisión es un conjunto de reglas que segmentan el espacio de predictores en varias regiones perpendiculares a los ejes. Es un modelo no lineal que puede utilizarse para regresión y clasificación. En el presente trabajo se utilizan árboles de decisión para **regresión**, donde se toma la **media** de las etiquetas correspondientes a la región de predictores; a diferencia de su uso para clasificación, donde se toma la moda dentro del espacio de predictores segmentado.

A cada segmento, o región, le corresponde una predicción fijada durante la construcción. La salida del árbol para una muestra está determinada por la región del espacio de predictores a la que corresponde.

El entrenamiento de un árbol es el proceso de segmentación de espacio de predictores y de asignación de una salida a cada región (u hoja) del árbol.

El caso particular de un modelo **CART**, *Classification And Regression Tree* es con una construcción recursiva. Muy genéricamente, se puede dividir en dos pasos:

1. Dividir el espacio de predictores en J regiones perpendiculares e independientes, de tal manera que sean mutuamente exclusivas y colectivamente exhaustivas. Esto es, que se incluyan todos los posibles valores $X_1, X_2, ..., X_p$ en las J regiones $R_1, R_2, ..., R_J$, sin que se repita un valor en dos regiones distintas.
2. Para cada observación que esté dentro de la región R_j, se realiza la misma predicción, que es la media de las variables de respuesta para las observaciones de entrenamiento en R_j

La siguiente pregunta sería: ¿cómo se construyen las regiones $R_1, ..., R_J$? En teoría podrían tener cualquier forma. Para una mayor interpretabilidad de resultados se toma un subconjunto perpendicular a los ejes, normalmente conocido como *cajas*. El objetivo es encontrar las cajas $R_1, ..., R_J$ que minimicen el RSS dado por:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \tag{6.1}$$

donde $\hat{y}_{R_j}$ es la media de respuestas de las observaciones dentro de la caja j.

Para optimizar la segmentación del conjunto de observaciones de entrenamiento, se utiliza un enfoque conocido como *recursive binary splitting*, el cual es una división binaria porque divide los grupos en dos, y es recursiva porque cada uno de esos grupos se subdivide de nuevo en dos grupos.

Primero, se divide el espacio de predictores en dos subgrupos, con un punto de corte en s, tal que por un lado estén $R_1(j, s) = \{\mathbf{X} | X_j < s\}$ y por otro $R_2(j, s) = \{\mathbf{X} | X_j \geq s\}$, escogiendo s de tal forma que exista la mayor reducción de RSS, y donde $\mathbf{X}$ es un vector de variables aleatorias $\mathbf{X} = (X_1, X_2, ...X_p)$.

El objetivo consiste en encontrar los valores j y s que minimicen:

$$\sum_{i: x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2 \tag{6.2}$$

donde $\hat{y}_{R_1}$ es el promedio de las respuestas para las observaciones de entrenamiento en la región $R_1(j, s)$, y análogo para $\hat{y}_{R_2}$.

Lo siguiente es aplicar la recursividad del método; esto implica repetir el proceso encontrando el mejor predictor y el mejor punto de corte, pero dentro de alguna de las regiones ya encontradas, en lugar del espacio completo de predictores. Este proceso continúa, no necesariamente de manera simétrica entre las regiones, sino con el objetivo de minimizar el RSS.

El proceso puede continuar hasta que se cumpla algún criterio de paro. Este puede ser: que ninguna región tenga más de n observaciones, o que el RSS alcance un nivel específico, o que el cambio en RSS por incurrir en una nueva iteración sea ínfimo, o bien, utilizando *tree pruning*, como se ve más adelante.

Una vez que se han creado las regiones $R_1, ..., R_j$, se realiza la predicción para la nueva observación. Para eso, se utiliza el promedio de las observaciones de entrenamiento correspondientes a la región en que la nueva observación pertenecería.

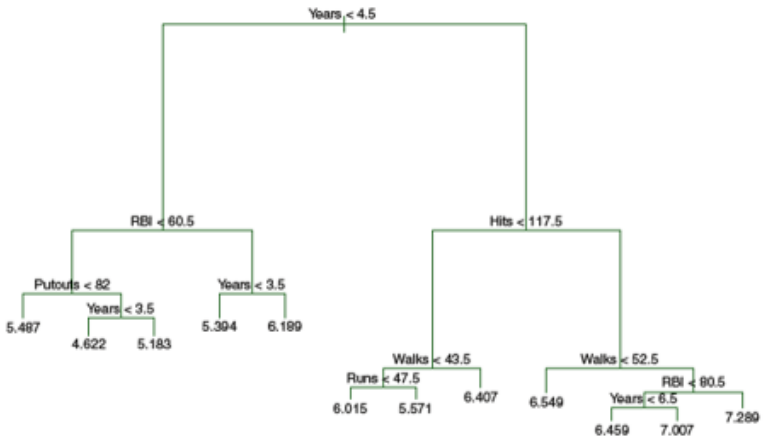


Figura 6.1. *Árbol de regresión completo, con 12 nodos. Ejemplo de la base de datos Hitters, tomado del libro de introducción al aprendizaje estadístico. Ver [2] en la bibliografía.*

Podado de árboles

Un árbol de decisión tal y como se ha definido, es un modelo que tiende a crear divisiones cada vez más pequeñas, lo que implica un riesgo de sobreajuste. Para ello, se puede utilizar la técnica de *podar el árbol*, o *tree pruning* en inglés.

Un árbol con menor número de *hojas*, esto es, menos regiones $R1, ..., R_j$, podría conducir a un modelo con menor varianza y mayor sesgo.

Una forma de lograr este objetivo de menos regiones, y por tanto menor varianza, sería utilizar una instrucción de paro de manera que sólo se haga una nueva división si la disminución en el RSS supera un límite establecido. Sin embargo, no es la mejor estrategia, pues podría ser que la siguiente división hubiera generado una disminución muy significativa en el RSS.

Por ello, una mejor estrategia es construir un árbol T_0, y *"podarlo"* para obtener un árbol más pequeño. El proceso ideal es conocido como *cost complexity pruning*. Se considera una secuencia de árboles indexada por un parámetro α. Para cada valor de α, existe un árbol más pequeño, T, que es subconjunto del árbol principal, T_0.

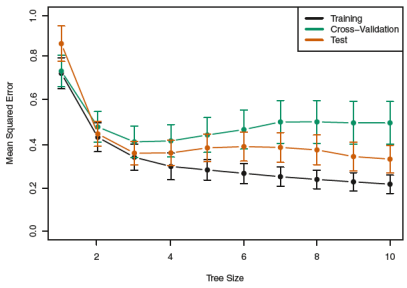
Esto es, $T \subset T_0$, tal que se minimice:

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T| \tag{6.3}$$

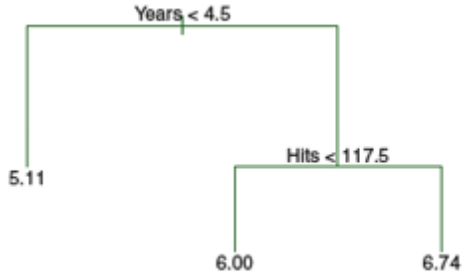
En la fórmula anterior, $|T|$ indica el número de *hojas* o nodos del árbol T; R_m es una región, o un subespacio del espacio total de predictores, correspondiente al m-ésimo nodo; y $\hat{y}_{R_m}$ es la respuesta predicha asociada al espacio R_m, esto es, el promedio de las observaciones de entrenamiento en R_m.

El término α controla el costo entre tener un árbol con $|T|$ nodos y el error de entrenamiento generado por ese árbol. Es decir, se prefiere un árbol con menos nodos y error de entrenamiento relativamente bajo, a uno con una cantidad excesiva de nodos, aunque tenga un error de entrenamiento menor. Para encontrar el valor de α, se pueden usar métodos de *cross-validation*.

La ecuación (4.3) es similar a algo conocido como *regresión lasso*, donde también se pone un costo a una regresión por aumentar su complejidad.



(a) *Se analizan los errores de entrenamiento, cross-validation y el error de prueba. El error mínimo ocurre en un árbol de tamaño tres.*



(b) *Aquí se ve el árbol podado, con sólo tres nodos.*

6.3 Árboles vs Modelos lineales

A continuación se presentan algunas ventajas y desventajas de utilizar modelos de árboles contra los modelos lineales.

Ventajas y desventajas de modelos de árboles

Algunas de las ventajas de los modelos de árboles por sobre modelos lineales son:

- Facilidad de interpretación.
- Se cree que se asemejan más al proceso de toma de decisiones de las personas.
- Facilidad de descripción gráfica, en especial si son pequeños.

Entre las desventajas, encontramos:

- Facilidad de generar sobreajuste si no se selecciona correctamente un criterio de paro.
- Se asume que el espacio se puede partir de manera perpendicular a los ejes.

Para evitar el sobreajuste, se tienen agregaciones de árboles de decisión. Esto es, generar conjuntos de árboles de decisión utilizando métodos como *bagging*, *random forests* y *boosting*, con los cuales se incrementa sustancialmente el rendimiento predictivo de los modelos de árboles.

6.4 *Bagging & Random Forests*

Para un conjunto de observaciones independientes $Z_1, ..., Z_m$, cada una con varianza σ^2, la varianza de la media $\bar{Z}$ de las observaciones está dada por $\frac{\sigma^2}{n}$. En otras palabras, promediar un conjunto de observaciones reduce la varianza del promedio $\bar{Z}$.

Realizando *bootstrapping*[1] de distintos conjuntos de entrenamiento de la población, se construye un modelo predictivo por separado para cada conjunto y se promedian las predicciones resultantes, disminuyendo la varianza de la media de las observaciones e incrementando la precisión predictiva (Ver [2]). Se puede calcular: $\hat{f}^1(x)$, ..., $\hat{f}^B(x)$, usando B conjuntos de entrenamiento separados, y promediarlos para obtener un modelo de aprendizaje estadístico con baja varianza dado por:

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x) \tag{6.4}$$

Sin embargo, no siempre es posible tener acceso a distintos conjuntos de entrenamiento. En lugar de eso, se pueden tomar distintas muestras del mismo conjunto, se entrena el método en el b-ésimo conjunto obtenido por *bootstrapping* para obtener $\hat{f}^{*b}(x)$, y finalmente promediar las predicciones para obtener:

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x) \tag{6.5}$$

[1] En estadística, es un análisis o una métrica en el que se utiliza muestreo simple con reemplazo

A este proceso se le conoce como ***bagging***. Para aplicar este proceso a árboles de decisión, se construyen B árboles regresivos usando B' conjuntos de entrenamiento obtenidos por *bootstrapping*, y se promedian las predicciones. Así, al promediar los B árboles se reduce la varianza del promedio de las observaciones.

El utilizar múltiples árboles en un proceso como *bagging*, aunque mejora la precisión en los estimados, disminuye la interpretabilidad del modelo. Se puede llegar a entender el modelo si se utiliza el RSS sobre las variables y qué tanto cambia al incluir o eliminar alguna de ellas.

Una desventaja de utilizar *bagging* es que se puede considerar una variable predictiva muy fuerte de manera que todos los árboles comiencen en la división de esa variable, creando árboles muy parecidos entre sí. Al ser árboles similares, se pierde el efecto de disminuir la varianza por medio de realizar el promedio de los árboles, pues sería casi como realizar un promedio sobre el mismo árbol.

Los ***random forests*** representan una mejora en el método anterior al incluir un cambio que elimina la correlación entre los árboles que se utilizan, y toma subconjuntos de las variables.

Se construye de la misma forma que con el método de *bagging*, utilizando muestras de entrenamiento obtenidas por *bootstrapping*. La diferencia radica en que, cada que se considere una nueva división en el árbol, se toma una muestra aleatoria de m predictores, misma que pertenece al conjunto total de p predictores, para considerarse como candidatos a ser divididos. Una nueva muestra aleatoria de m predictores se toma cada que se realiza una nueva división, generalmente $m \approx \sqrt{p}$.

La lógica detrás de esto es que no todos los árboles comenzarán por la misma variable, pues en algunos subconjuntos no se toma en cuenta. Así, se tienen árboles distintos entre sí, pues comienzan con diferentes variables iniciales y, por lo tanto, con menor correlación entre ellos, permitiendo que se reduzca la varianza como se vio anteriormente.

En promedio, $\frac{(p-m)}{p}$ de los cortes no considerarán el predictor más fuerte, con lo que otras variables pueden iniciar el árbol disminuyendo la correlación con otros árboles y, en consecuencia, haciendo que el promedio de los árboles resultantes sea menos variable y, por tanto, más confiable.

Se puede considerar el método de *bagging* como un caso especial de *random forests* con $m = p$. En ninguno de los métodos se genera un sobreajuste si se incrementa la cantidad de subconjuntos B a considerar, por lo que se toma un valor B, que sea lo suficientemente grande para encontrar patrones en los datos, mismos que hubieran sido difíciles de encontrar si no se hubieran hecho particiones de los datos.

Capítulo 7

Validación del modelo y selección de variables

7.1 Introducción

En capítulos anteriores se estudiaron las bases para el aprendizaje supervisado. Se vieron los tipos de modelos: paramétrico y no paramétrico; así como sus usos: modelos de regresión y modelos de clasificación.

Para poder construir el portafolio de réplica, lo siguiente sería plantearse: ¿cómo se pueden escoger las variables para el modelo, de manera que se utilicen las más significativas?, y ¿cómo determinar la precisión del modelo?

Como se vio en un capítulo anterior, el error de entrenamiento no siempre es similar al error del conjunto de prueba, por lo que, basarse en un modelo que tenga un error de entrenamiento bajo, no necesariamente garantiza los mejores resultados. Algunos métodos para seleccionar el mejor un subconjunto de variables y obtener los mejores resultados son:

- ***Subset selection***. Selección de un subconjunto de p variables, de un total de n variables, desde antes de realizar el ajuste del modelo.

- **Método de reducción de variables**. Consiste en realizar el ajuste del modelo desde las n variables y posteriormente ir reduciendo las variables a cero, con relación al estimado por mínimos cuadrados. Esta reducción (también conocida como regularización) tiene el efecto de reducir la varianza..

- **Reducción de dimensión**. Involucra el proyectar las variables a un subespacio M-dimensional, donde $M < p$. Esto se logra al calcular M combinaciones lineales diferentes, o proyecciones, de las variables. Posteriormente, estas M proyecciones son usadas como predictores para ajustar el modelo utilizando mínimos cuadrados.

En el presente trabajo, se utilizará el método de reducción de variables, por lo que únicamente se analiza ese caso.

7.2 Selección de variables y regularización del modelo

"El método de regularización es cualquier modificación que hagamos a un algoritmo de aprendizaje que tiene la intención de reducir su error de prueba, pero no su error de entrenamiento." (Goodfellow et al's "Deep Learning" Sección 5.2.2)

Algunas de las formas para seleccionar variables, por medio del método de selección de un subconjunto, son: el mejor subconjunto, o *best subset* en inglés, y la selección paso a paso, o *stepwise selection* en inglés.

Mejor subconjunto

Para utilizar este método, realizamos el ajuste por mínimos cuadrados al modelo para cada combinación del total de p predictores. Esto es, ajustamos p modelos que contienen exactamente una variable, luego todas las combinaciones $\binom{p}{2} = p * (p-1)/2$ modelos que contienen exactamente dos variables, luego tres, cuatro y así sucesivamente hasta un único modelo que contenga p. Luego, buscamos el que brinde mejores resultados, utilizando el criterio de *AIC*, *BIC* o R^2 *ajustada*.

Computacionalmente no es el mejor método, debido a que se deben ajustar 2^p modelos, y en particular para este trabajo, se utilizan muchas variables, por lo que no se entrará más a detalle en este método.

Selección por pasos

Existen tres tipos de métodos *stepwise selection*:

- ***Forward stepwise selection***.

  El método de *forward stepwise selection* comienza por realizar un ajuste sin utilizar ninguna variable y luego va agregando variables una por una hasta utilizar todas las variables del modelo. En particular, en cada paso, se considera la variable que más aporte a mejorar el ajuste del modelo.

  En concreto, los pasos para este método son:

  1. Comenzar con un modelo *nulo*, el cual no contiene variables, lo llamamos M_0
  2. Para $k = 0, 1, ..., p-1$:
     (a) Considerar todos los $p-k$ modelos que aumentan una nueva variable a M_k.
     (b) Escoger el *mejor* entre esos $p-k$ modelos, y lo llamamos M_{k+1}. El *mejor* modelo es aquel con menor RSS o mayor R^2.
  3. Seleccionar un único y mejor modelo entre los p modelos: $M_0, ..., M_p$, utilizando alguno de los métodos para determinar el ajuste del modelo: AIC, BIC, o R^2 ajustada.

Para este caso, se tienen $1 + p * (p + 1)/2$ modelos, lo cual es sustancialmente menor al método anterior, aunque no garantiza escoger la mejor entre todas las 2^p posibles opciones, debido a que se basa en la selección de la variable anterior para la siguiente. Por ejemplo, si el mejor modelo con una variable es utilizando X_1, entonces el mejor modelo con dos variables utilizará X_1 y otra variable, cuando en realidad el mejor posible modelo es quizás uno de dos variables, de las cuales ninguna es X_1.

- ***Backward stepwise selection***.

  Es la misma lógica que en el algoritmo de *forward stepwise selection*, sólo que aquí se comienza con un modelo *completo*, el cual incluye todas las p variables, y se van eliminando k variables, con $k = 1, 2, ..., p$, para encontrar el mejor modelo en cada uno de los casos, $M_0, ..., M_p$ y finalmente seleccionar el mejor entre los modelos anteriores.

  Tiene las mismas ventajas y desventajas que el método de *forward stepwise selection*, y se generan el mismo número de modelos.

- ***Hybrid selection***.

  Este método es una mezcla de los dos anteriores. Aquí se va agregando una variable, pero cada que se agrega una, se elimina otra para evitar variables que dejen de aportar al modelo. De esta forma, se trata de llegar al mismo resultado que utilizando el método del mejor subconjunto, pero con un coste computacional menor.

En el presente trabajo, se utiliza el método de ***backwards stepwise selection***.

7.2.1 Seleccionando el mejor modelo

Los métodos anteriores resultan en la creación de conjuntos de modelos, cada uno con un subconjunto de las p variables predictoras. Para poder implementar esos métodos, se debe seleccionar el modelo que sea el *mejor*. Cuando utilizamos RSS o R^2 estamos seleccionando el menor error de entrenamiento. Sin embargo, como se vio en capítulos anteriores, lo que nos interesa minimizar es el error de prueba, el cual no necesariamente se minimiza al reducir el error de entrenamiento, por lo que el uso de RSS o R^2 no es el apropiado para dicho objetivo.

Para poder estimar el error de prueba, y encontrar el modelo donde dicho error es mínimo, se utiliza alguno de los siguientes enfoques:

1. Estimar el error de prueba de forma *indirecta*, al realizar un ajuste al error de entrenamiento.
2. Estimar el error de prueba de forma *directa*, utilizando un subconjunto de validación o un enfoque de validación cruzada (*cross-validation*).

Ambos enfoques se consideran más adelante, aunque no se entra a detalle debido a que la mayoría de los indicadores pueden ser estimados con paquetes computacionales que se ven en el capítulo del desarrollo del modelo.

Estimación indirecta: AIC, BIC y R^2 ajustada

El criterio de AIC, *Akaike Information Criterion*, se define al ajustar el modelo por máxima verosimilitud. Aquí, lo que se busca es el modelo con el menor AIC, pues indica el menor error de prueba. Para el caso de una regresión lineal, con errores Gaussianos, máxima verosimilitud y mínimos cuadrados, son lo mismo. En este caso, el AIC está dado por:

$$AIC \ = \ -2 \ \ell(\theta_{mv}) \ + \ 2d \ = \ ... \ = \ \frac{1}{n\hat{\sigma}^2}(RSS \ + \ 2d\hat{\sigma}^2) \tag{7.1}$$

Donde n denota el número de observaciones y d la cardinalidad del subconjunto de covariables a utilizar.

El criterio de BIC, *Bayesian Information Criterion*, se deriva desde el punto de vista bayesiano, aunque termina siendo similar al AIC. Para un modelo por mínimos cuadrados con d predictores, el BIC está dado por:

$$BIC \ = \ -2 \ \ell(\theta_{mv}) + 2d \ln(n) \ = \ ... \ = \ \frac{1}{n}(RSS \ + \ \log(n)d\hat{\sigma}^2) \tag{7.2}$$

Se puede observar que el BIC cambia el valor de 2 en el AIC, por $\log(n)$, donde n es el número de observaciones. Dado que $\log(n) > 2$ para cualquier $n > 7$, el estadístico BIC generalmente pondrá mayor castigo a los modelos con muchas variables, prefiriendo así modelos con menor número de variables. En ambas fórmulas, d es el número de predictores y $\hat{\sigma}^2$ es un estimado para la varianza del error ϵ.

El estadístico de R^2 ajustado es similar al R^2 visto en capítulos anteriores pero ajustado para d variables, evitando así que el valor incremente conforme se incrementan las variables. Para un modelo con d variables, un R^2 ajustado se calcula como:

$$R^2 \ Ajustado \ = \ 1 \ - \ \frac{\frac{RSS}{(n \ - \ d \ - \ 1)}}{\frac{TSS}{(n \ - \ 1)}} \tag{7.3}$$

A diferencia del AIC y BIC, para los cuales un valor menor indica un modelo con menor error de prueba, un valor de R^2 ajustado mayor, indica un modelo con menor error de prueba. La lógica detrás de este estadístico es que, una vez que se han incluido las variables correctas en el modelo, agregar una variable adicional generaría ruido y provocaría una disminución en el estadístico. A diferencia del estadístico de R^2, R^2 ajustado implica un costo por agregar variables innecesarias.

Aunque las fórmulas presentadas son para un caso de un modelo lineal, es posible hacer el cálculo de los mismos estadísticos para diferentes modelos.

Estimación directa: Validación cruzada y *Bootstrapping*

- ***Cross-Validation: Validation Set***. En este método, se divide el conjunto de entrenamiento aleatoriamente en dos partes: Un conjunto de entrenamiento y un conjunto de validación. Se ajusta el modelo en el conjunto de entrenamiento y luego se prueba en el conjunto de validación. El ECM de validación proporciona un estimado cercano al que sería el ECM de prueba.

- ***Cross-Validation: Leave-One-Out Cross Validation (LOOCV)***. Es el mismo caso que el *Cross-Validation: Validation Set*, sólo que en lugar de dividir el conjunto en dos partes iguales, se deja una única observación fuera, para que funcione como el conjunto de validación. Con esto se pretende eliminar la variabilidad generada por escoger la mitad de variables de entrenamiento de manera aleatoria en el método anterior. Este método se va repitiendo dejando fuera una observación distinta en cada ocasión. Al final, el ECM de validación es el promedio de los ECM que se fueron obteniendo para cada observación.

- ***Cross-Validation: K-fold Cross Validation***. Similar a los anteriores, pero divide el conjunto en k grupos del mismo tamaño, tomando el primero como conjunto de validación y los restantes $k-1$ como entrenamiento. Podemos ver que el método anterior (LOOCV) es un caso especial de este método, donde $k = n$, pues divide en el conjunto en n valores (total de observaciones) y sólo se queda con 1. La ventaja de este método contra el anterior es el coste computacional, ya que no tiene que ir recorriendo las observaciones de una por una, sino con subconjuntos de $\frac{n}{k}$. Asimismo, brinda un mejor estimado para el ECM de prueba debido al equilibro entre sesgo y varianza, pues aunque LOOCV tiene menor sesgo, k-fold tiene menor varianza. Se calcula como el promedio de los k ECM formados.

- ***Bootstrap method***. Es cualquier prueba o métrica que se basa en muestreo aleatorio con reemplazo. Permite asignar medidas de precisión (definida en términos de sesgo, varianza, intervalos de confianza, predicción de error, entre otras) a un estadístico, basado en los estimados de la muestra aleatoria. Es muy útil en modelos no paramétricos.

Es importante mencionar que, para efectos del presente trabajo, se deben de hacer ajustes a estas medidas para poder ser utilizadas. Esto debido a que en una serie de tiempo no se pueden aplicar los métodos directamente, pues la serie de tiempo tiene un orden, el cual puede implicar que una observación y_n esté determinada por el conjunto de observaciones anteriores $y_1, ..., y_{n-1}$, pero no por observaciones futuras. *Cross-validation* implica poder cambiar el valor de prueba en cualquier punto de las observaciones, lo cual no es posible en una serie de tiempo ordenada.

Para poder realizar pruebas se utiliza el mismo principio que en K *cross-validation*, pero dividiendo las observaciones en conjunto de entrenamiento y prueba, utilizando ventanas móviles ordenadas en el tiempo. Por ejemplo, tomando las observaciones ordenadas $y_1, ..., y_k$ como observaciones de entrenamiento y probando para $y_{k+1}, ..., y_n$ como prueba. Posteriormente, recorrer la ventana en el siguiente paso a $y_2, ..., y_{k+1}$ como entrenamiento, y probar en $y_{k+1}, ..., y_{n+1}$. Y así sucesivamente.

Parte III

Desarrollo, resultados y conclusiones

Capítulo 8

Bases para el desarrollo

8.1 Introducción

En los siguientes capítulos, se construyen los portafolios utilizando métodos de machine learning y se utilizan diversas métricas para comprobar que los portafolios replican el índice. Posteriormente, se crea un algoritmo de inversión, con reglas que se verán más adelante. Finalmente, se analizan rendimientos y se llega a las conclusiones del trabajo.

En el presente capítulo, se sientan las bases para el desarrollo, tanto de los modelos, como del algoritmo de trading. Se utilizan datos de las empresas que componen al S&P 500, analizando para el año 2019, y posteriormente se hace una extensión al 2020, para ver el comportamiento del algoritmo en periodos de alta volatilidad[1].

Para poder analizar la estrategia para un determinado año, se debe correr el modelo para el año anterior, y así construir bandas de media y desviación estándar para utilizar en el año objetivo, y evitar lo que se conoce como *look-ahead bias*[2]. Es decir, no se puede construir una media y desviación estándar con precios de un determinado año, y luego operar sobre esos mismos precios.

Se utilizaron los siguientes datos en el ejercicio:

- Del 1ro de enero de 2015 al 31 de diciembre de 2018, correspondientes al entrenamiento de los modelos.
- Del 1ro de enero de 2019 al 31 de diciembre de 2019, correspondientes a la prueba de los modelos. Para el ejercicio de 2020, se recorrió la ventana de datos, tanto de prueba como de entrenamiento, un año.

Se tienen diversas métricas para evaluar los resultados. Para verificar que el portafolio replica de manera correcta al índice, se comparan métricas como: Beta, R^2 ajustada, *tracking error* y volatilidad. También se utilizan métricas de riesgo y rendimiento como el VaR, el CVaR, índice de Treynor e índice de Sharpe. Estas mismas métricas de rendimiento se utilizan para corroborar que la estrategia presenta resultados positivos.

[1] La alta volatilidad en 2020 se debió a la crisis derivada por el COVID 19.

[2] El *look-ahead bias* es un tipo de sesgo que ocurre cuando, en un estudio o simulación, se utilizan datos que en realidad no estaban disponibles durante el periodo de estudio.

8.2 Construcción de portafolios y estrategia de inversión

La construcción de portafolios consiste en lo siguiente:

1. Primero se toman los datos de precios de cierre de todas las emisoras del S&P 500 y se convierten a rendimientos diarios.
2. Se hace un análisis de correlación para eliminar variables que tengan elevada correlación entre sí. Si dos variables tienen elevada correlación entre ellas, se elimina la que tenga menor correlación con el índice SPY.
3. Se construye el modelo de machine learning para seleccionar las emisoras que estarán dentro del portafolio, con sus respectivos pesos.
4. Una vez que se tienen las variables del portafolio y sus pesos, se hace una suma producto entre el rendimiento y el peso, para obtener el rendimiento del portafolio total.
5. La serie de rendimientos del portafolio se aplica al primer precio observado del SPY, en el año correspondiente al análisis, y se obtiene una serie de precios del portafolio.

La estrategia de inversión consiste en lo siguiente:

1. Una vez que se tiene el portafolio modelo, se construye una valuación relativa entre este portafolio y el SPY Index, para los datos del año a analizar y datos de un año anterior, para evitar el *look-ahead bias*. La valuación relativa se construye dividiendo el precio de uno entre el precio del otro.
2. Se calcula la media y dos desviaciones estándar de la valuación relativa entre portafolios, utilizando los datos del año anterior al año del análisis.
3. Se comienza con un millón de dólares de capital, el cual se invertirá a la tasa libre de riesgo mientras se tenga en posición, esperando vender un millón de dólares en corto[3] un activo y utilizar el dinero para comprar el otro.
4. La razón del año de análisis se coloca dentro del rango de la media y desviaciones estándar del año anterior. Cuando la razón entre los activos exceda las dos desviaciones estándar, y vuelva a entrar al rango, se ejecuta una operación de compra de un portafolio, y venta del otro.[4]
5. La operación se cierra cuando la razón cruce la media. Al momento de cerrar la operación, se venden los títulos del ETF que estaba largo, y con ese dinero, se compran los títulos del ETF que estaba corto.
6. Se obtienen métricas de riesgo y rendimiento del portafolio como el índice de Treynor o el índice de Sharpe, así como el VaR.

[3]Una venta en corto consiste en vender un activo en el tiempo t_0, el cual no se tiene en posición, para comprarlo y entregarlo en el tiempo t_n. De manera direccional, se utiliza esperando que el precio del activo baje, para venderlo en t_0 a un precio mayor al que se comprará en t_n.

[4]Se compra el de menor precio y se vende el de mayor precio.

8.3 Métricas de evaluación del portafolio réplica

Para comprobar que un portafolio realmente replica a un índice es necesario obtener ciertas métricas para la evaluación del portafolio. Algunas de ellas pueden ser las siguientes:[5]

- **Beta**: Es una medida de volatilidad de un portafolio comparado contra un *benchmark*, generalmente un mercado. Para que sea significativa, debe existir una relación entre el activo y el mercado contra el cual se va a analizar.

  Describe efectivamente, la actividad de los rendimientos de un instrumento, mientras responde a los movimientos del mercado. Se calcula de la siguiente manera:

  $$\beta = \frac{Covarianza\ (Re,\ Rm)}{Varianza\ (Rm)}$$

  Donde *Re* es el rendimiento del instrumento, *Rm* el rendimiento del mercado, la covarianza determina la relación del cambio en los rendimientos del activo con respecto a cambios en los rendimientos del mercado, y la varianza el qué tanto se alejan los datos del valor promedio.

- **R^2**: Medida estadística que representa la proporción de la varianza de la variable dependiente, que es explicada por la variable independiente.

- **Tracking Error**: Divergencia entre el comportamiento del precio de un portafolio, y el de su benchmark. Se reporta como la desviación estándar de la diferencia de los rendimientos, que es la diferencia entre el rendimiento que un inversionista recibe y el del benchmark que trata de replicar. Se calcula de la siguiente manera:

  $$Tracking\ error\ = Desv.\ Est.\ (P\ -\ B)$$

  Donde P son los rendimientos del portafolio, y B los del benchmark.

- **Volatilidad**: La volatilidad de un portafolio se obtiene con su varianza y, por lo tanto, es una medida de su riesgo y fluctuación de precios en el tiempo. Comprar la varianza de dos portafolios nos permite entender las posibles fluctuaciones futuras que ambos podrían tener.

Además de estas medidas que sirven para corroborar si el portafolio replica de manera adecuada al índice, se obtienen otras métricas para conocer más acerca del portafolio. Estas métricas miden la relación que existe entre el riesgo y rendimiento del portafolio, así como su posible máxima pérdida esperada.

[5]Todas las definiciones se tomaron de investopedia

8.4 Métricas de riesgo y rendimiento de los portafolios

En el presente trabajo se tienen tres portafolios: Un portafolio generado con el modelo de regresión lineal, otro generado con el modelo de *random forest*, y el portafolio final que se compone de la inversión en efectivo y, si la estrategia así lo define, de la posición corta y larga de alguno de los portafolios anteriormente mencionados y el índice de mercado.

A diferencia de las métricas anteriores, las siguientes métricas aplican tanto para los portafolios de replica como para el portafolio con la estrategia final que se compone de la combinación de portafolios y la inversión de efectivo a la tasa libre de riesgo.

- **Valor en Riesgo**: El valor en riesgo, abreviado como VaR por sus siglas en inglés de *Value at risk*, es un valor que cuantifica las posibles pérdidas financieras de un portafolio. También se conoce como la máxima pérdida esperada. Puede ser calculada utilizando rendimientos históricos, método de varianza y covarianza[6] y métodos Monte Carlo de simulación. Un mayor VaR representa un portafolio con mayor posible pérdida esperada y, por tanto, mayor riesgo.

- ***Treynor ratio***: El índice de Treynor, o *Treynor ratio* en inglés, también es conocido como índice de recompensa-volatilidad, pues mide cuánto retorno en exceso se generó por cada unidad de riesgo tomada por el portafolio. La fórmula es la siguiente:

  $$Treynor\ ratio\ = \frac{r_p\ -\ r_f}{\beta_p}$$

  Donde r_p es el rendimiento del portafolio, r_f es la tasa libre de riesgo y β_p es la beta del portafolio.

- ***Sharpe ratio***: El índice de Sharpe, o *Sharpe ratio* en inglés, es una métrica financiera que sirve para conocer el promedio de rendimiento en exceso a la tasa libre de riesgo, por unidad de volatilidad o de riesgo. La volatilidad es la fluctuación de precios de un activo o un portafolio. Su fórmula es:

  $$Sharpe\ ratio\ = \frac{r_p\ -\ r_f}{\sigma_p}$$

  Donde r_p es el rendimiento del portafolio, r_f es la tasa libre de riesgo y σ_p es la desviación estándar del rendimiento en exceso del portafolio.

  Se busca que ambos índices de un portafolio sean positivos y superiores a los de su *benchmark*, pues indicaría que el portafolio tiene mayor rendimiento con respecto a su riesgo.

La diferencia principal entre ambos índices es que, el índice de Sharpe ayuda a entender el rendimiento de un portafolio comparado con su riesgo, mientras que el índice de Treynor explora el retorno en exceso generado por cada unidad de riesgo adicional en el portafolio.

[6] Este es el método utilizado en el presente trabajo.

8.5 Análisis y exploración de datos

8.5.1 Bibliotecas

Las paqueterías utilizadas en el presente trabajo son las siguientes:

- **MLModels_Carlos**: La biblioteca de funciones del presente trabajo. De ahí se mandan llamar las funciones con las que se construyen los modelos y otras necesarias para el presente trabajo. Se puede utilizar la función help() para ver la documentación de cada una de las funciones creadas y utilizadas en el trabajo.

- **NumPy**: Es un paquete fundamental para computación científica en Python. Una biblioteca de Python que provee objetos de arreglos multidimensionales, objetos derivados de estos arreglos (como matrices), y una gran gama de rutinas para realizar operaciones rápidas con arreglos, incluyendo operaciones matemáticas, operaciones lógicas, manipulación de forma, ordenar datos, seleccionar datos, álgebra lineal básica, operaciones estadísticas básicas, simulación de números aleatorios, entre otras. Ver [13, 14] en la bibliografía.

- **Pandas**: Es una herramienta de alto nivel para manipulación y análisis de datos, desarrollada por Wes McKinney. Está construida sobre la paquetería de NumPy y su estructura de datos clave es llamada DataFrame. Los DataFrames permiten guardar y manipular tablas de datos en filas de observaciones y columnas de variables. Es de código abierto, rápida, poderosa, flexible y fácil de usar; construida sobre el lenguaje de programación de Python. Ver [15, 16] en la bibliografía.

- **Scikit-learn**: Biblioteca de *machine learning* para Python. Incluye herramientas simples y eficientes para análisis predictivo de datos. Es accesible para todos y reutilizable en varios contextos. Está construida sobre NumPy, SciPy y matplotlib. De código abierto, utilizable comercialmente y con licencia BSD. Ver [17] en la bibliografía.

- **Statsmodels**: Es un módulo de Python que provee clases y funciones para la estimación de muchos modelos estadísticos diferentes, así como para conducir pruebas estadísticas y exploración estadística de datos. Tiene una lista extensiva de resultados estadísticos para cada estimador. Los resultados se prueban contra paquetes estadísticos existentes para asegurar que estén correctos. Ver [18] en la bibliografía.

- **Matplotlib**: Es una biblioteca comprensiva para la creación de visualizaciones estáticas, animadas e interactivas en Python. Sirve para desarrollar gráficos para publicación con unas cuantas líneas de código y utilizar figuras que pueden ampliarse con zoom, actualizarse, etc. También permite tomar control del estilo de los gráficos, como propiedades de las fuentes, estilo de líneas, entre otros. Además, se pueden exportar y guardar los gráficos resultantes en un gran número de formatos. Ver [19] en la bibliografía.

- **Plotly**: Permite a usuarios el importar, copiar y pegar, o transmitir datos, para ser analizados y visualizados. Es una biblioteca de gráficos científicos. Ver [20] en la bibliografía.

8.5.2 Carga y análisis de datos

Una vez que se importaron las paqueterías, se procede a la carga y análisis de los datos.

Lo primero que se tiene que hacer es leer los datos. Estos datos se pueden obtener de una terminal de Bloomberg[7], o de la página de Yahoo Finance[8] y se cargan en un Excel para leer en Python.

Se utiliza una variable para determinar el año objetivo para el cuál se realizará el estudio.

Los datos a cargar son:

- Precios de todos los activos que formaron parte del S&P 500 desde 2014 hasta el último disponible en 2020. De esta manera, se puede analizar cuatro años de historia más un quinto año como año objetivo, a partir de 2018 y también para 2019 y 2020.
- Empresas que formaron parte del S&P 500, del 2010 al 2020, para poder sacar el subconjunto de precios necesarios para el análisis y creación de modelos para un determinado año.

Obtenemos el subconjunto de activos y fechas, a partir de una variable de año objetivo. Finalmente, se calculan los rendimientos y se separan las variables independientes de la variable objetivo. Los log-rendimientos se calculan con la siguiente fórmula:

$$Rendimiento = Ln\left(\frac{P_{t+1}}{P_t}\right)$$

Equity Ticker	A US Equity	AAL US Equity	AAP US Equity	AAPL US Equity	ABBV US Equity	ABC US Equity	ABT US Equity	ACN US Equity	ADBE US Equity	ADI US Equity	...	XLNX US Equity	XOM US Equity	XRAY US Equity	XRX US Equity	XYL US Equity	YUM US Equity	ZBH US Equity	ZION US Equity
Dates																			
2014-01-01	40.8958	25.250	110.68	20.036	52.81	70.31	38.33	82.22	59.879	50.93	...	45.92	101.20	48.48	32.0630	34.60	54.3677	93.19	29.96
2014-01-02	40.1950	25.360	109.74	19.755	51.98	69.89	38.23	81.13	59.290	49.28	...	45.97	99.75	47.96	31.3780	34.16	53.9938	92.24	29.65
2014-01-03	40.7027	26.540	112.88	19.321	52.30	69.94	38.64	81.40	59.160	49.61	...	45.62	99.51	48.19	31.5888	34.47	54.3317	92.64	29.86
2014-01-06	40.5025	27.030	111.80	19.426	50.39	69.69	39.15	80.54	58.120	49.33	...	45.42	99.66	47.90	31.8522	34.41	54.2886	93.24	29.65
2014-01-07	41.0817	26.905	113.18	19.287	50.49	70.45	38.85	81.52	58.970	49.59	...	45.52	101.07	48.64	32.1157	34.51	55.0508	95.10	29.74

5 rows × 504 columns

Figura 8.1. *Precios de las acciones que alguna vez fueron parte del S&P 500, entre el 1ro de enero de 2014 y el 31 de diciembre de 2020.*

Para que el análisis para determinado año sea posible, es necesario comenzar con un año anterior, para tener un histórico de la razón entre un portafolio generado y el ETF de mercado.

Una vez que se cuenta con la razón histórica, se realiza de nuevo este proceso, pero para los años de análisis del presente trabajo: 2019 y 2020.

[7]Empresa mundial de información financiera y noticias, cuya terminal permite el uso y descarga de datos financieros globales

[8]Parte de la red de empresas de Yahoo! donde se pueden descargar datos financieros y económicos de manera gratuita.

8.5.3 Análisis de colinealidad

Como se vio en capítulos anteriores, la colinealidad entre variables puede llegar a ser un problema que genere un sobreajuste al modelo. Para ello, se creó una función que permite eliminar las variables que tengan una elevada correlación con otras en la serie de datos. La función que se construyó para el código se llama **correlaciones**. Los detalles de la función son los siguientes:

```
Help on function correlaciones in module mlmodels_carlos:

correlaciones(X, y, c)
    Esta función encuentra dos variables que tengan una
    correlación igual o mayor al valor de corte dada en
    los parámetros, selecciona la que tenga menos correlación
    con la variable objetivo y la elimina de los datos.

    Parameters
    ----------
    X: pd.DataFrame(N,M)
        Matriz de variables independientes

    y: pd.DataFrame(N, 1)
        Matriz de variable objetivo

    c: decimal, c en [0,1]
        Valor de corte de la correlación entre variables independientes

    Returns
    -------

    list: Arreglo de variables que tienen correlación menor al punto de corte
          para que se pueda obtener un subconjunto con este arreglo.
```

Los parámetros que utilizamos para la función son: el DataFrame de los rendimientos de las variables independientes X, la variable objetivo y, y un punto de corte para las correlaciones, que en el caso particular del trabajo fue 0.85[9]. Los datos deben ser tomados hasta un año antes del año objetivo, para evitar *look-ahead bias*.

Si existen dos variables con correlación igual o mayor al punto de corte, se elimina la que tenga menor correlación con la variable objetivo. Con eso eliminamos las variables conlineales y preparamos los datos para utilizar en los modelos.

Para 2018, luego de correr la función, se redujo el número de variables de 504 a 476. Esta función se vuelve a correr para 2019, donde se redujo el número de emisoras de 504 a 468, y para 2020, donde la reducción fue de 505 a 479.

Equity Ticker	A US Equity	AAL US Equity	AAP US Equity	AAPL US Equity	ABBV US Equity	ABC US Equity	ABT US Equity	ACN US Equity	ADBE US Equity	ADI US Equity	...	XEC US Equity	XLNX US Equity	XOM US Equity
Dates														
2014-01-02	-0.017285	0.004347	-0.008529	-0.014124	-0.015842	-0.005991	-0.002612	-0.013346	-0.009885	-0.032934	...	-0.042746	0.001088	-0.014432
2014-01-03	0.012552	0.045480	0.028211	-0.022214	0.006137	0.000715	0.010667	0.003322	-0.002195	0.006674	...	0.007236	-0.007643	-0.002409
2014-01-06	-0.004931	0.018294	-0.009614	0.005420	-0.037204	-0.003581	0.013112	-0.010621	-0.017736	-0.005660	...	-0.020556	-0.004394	0.001506
2014-01-07	0.014199	-0.004635	0.012268	-0.007181	0.001983	0.010846	-0.007692	0.012094	0.014519	0.005257	...	0.010530	0.002199	0.014049
2014-01-08	0.016230	0.026590	-0.007806	0.006306	-0.002578	0.009747	0.008969	0.007698	-0.001188	0.002417	...	-0.001997	0.008531	-0.003270

5 rows × 476 columns

Figura 8.2. *Log-rendimientos de las acciones en 2018, luego de correr el algoritmo para eliminar correlaciones. Se puede observar que ahora se tienen 476 columnas, correspondientes a 476 emisoras.*

Antes de comenzar a correr los modelos, se dividen las observaciones en datos de entrenamiento (80%) y datos de prueba (20%).

En los siguientes capítulos se desarrollan los modelos y se analizan resultados.

[9]Este parámetro es móvil y simplemente fue determinado aleatoriamente como un buen punto de corte.

Capítulo 9

Desarrollo del modelo paramétrico

9.1 Introducción

En el presente capítulo, se construye el modelo de aprendizaje supervisado paramétrico, una regresión lineal múltiple. Para la selección de variables se probará con un *Backward Stepwise* y la selección del mejor modelo será por medio de la R^2 ajustada.

La función que se construyó es llamada ***backwards_regression***. La documentación para la función es la siguiente:

```
Help on function backwards_regression in module mlmodels_carlos:

backwards_regression(X, y, target_metric='Adj_R2')
    Este código hace una búsqueda de las mejores
    variables utilizando backwards stepwise

    Parameters
    ----------
    X: pd.DataFrame(N, M)
        Matriz de variables independientes

    y: pd.DataFrame(N, 1)
        Matriz de variable objetivo

    target_metric: str
        Variable de comparación de modelos. Opciones:
            * Adj_R2 (Predeterminado)
            * AIC
            * BIC
            * MSE

    Returns
    -------
    [model, list, dict]:
        Modelo de regresión lineal ajustado,
        Lista de mejores parámetros,
        Resultados de las métricas.
```

La función comienza tomando un modelo que incluye todas las variables y ajusta una regresión. Se guardan sus métricas (por default se utiliza la R^2 ajustada para comparar). Posteriormente, se elimina una variable, se vuelve a ajustar el modelo y se guardan sus métricas de comparación.

Si tiene una métrica mejor, este modelo se toma como el nuevo mejor modelo, se guardan las nuevas métricas y se elimina definitivamente la variable previamente eliminada para la prueba. Si no tiene mejores métricas, se regresa la variable eliminada y se pasa a la siguiente. Este proceso se repite hasta encontrar el modelo con la mejor métrica.

Para construir la regresión se utiliza la función de OLS en Python, que significa *Ordinary Least Squares*. Esto se refiere a que el ajuste del modelo se realiza por mínimos cuadrados. La biblioteca que permite utilizar esta función es la de ***statsmodel***

9.2 Parámetros y resultados de la función

Los parámetros de la función son:

- El DataFrame de variables independientes del modelo. Para este caso, es el DataFrame de rendimientos de las acciones.
- El DataFrame de la variable objetivo. En este caso, los rendimientos del S&P 500.
- Finalmente, la métrica que se utilizará para seleccionar el mejor modelo. La métrica predeterminada es la $R^2 ajustada$.

La función nos regresa:

- El modelo de regresión lineal ajustado, utilizando los mejores parámetros. En este caso, sería el modelo utilizando un subconjunto de las acciones originales.
- La lista de mejores parámetros, que serían las acciones que componen al ETF mejorado.
- Los scores, o resultados, de las distintas métricas que pudimos seleccionar.

Una vez que se termina de correr el modelo, se puede obtener un subconjunto de los rendimientos originales, con la lista de mejores parámetros. Así, obtenemos el siguiente *Dataframe*:

Equity Ticker	AAL US Equity	AAPL US Equity	ABBV US Equity	ABC US Equity	ABT US Equity	ACN US Equity	ADI US Equity	ADSK US Equity	AEE US Equity	AES US Equity	...	WMT US Equity	WU US Equity	WY US Equity
Dates														
2018-01-01	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000
2018-01-02	0.018283	0.017734	0.017426	0.023890	0.029693	0.004887	0.013943	0.021610	-0.010223	0.004606	...	-0.001622	0.003676	0.001134
2018-01-03	-0.012342	-0.000163	0.015528	0.003715	0.002209	0.004605	0.012330	0.020878	-0.005151	-0.000920	...	0.008685	-0.000524	0.008744
2018-01-04	0.006285	0.004634	-0.005719	-0.002227	-0.001699	0.011771	-0.001095	0.024296	-0.011427	-0.003687	...	0.000905	0.019731	-0.008177
2018-01-05	-0.000380	0.011309	0.017258	0.012032	0.002886	0.008215	0.004044	-0.011036	-0.000697	0.003687	...	0.005910	0.057444	-0.001417

5 rows × 292 columns

Figura 9.1. *DataFrame de rendimientos del subconjunto óptimo de acciones obtenido con regresión lineal y backwards stepwise selection. Para 2018, se generó un portafolio con 292 emisoras, en lugar de las 476 originales.*

Se obtiene un subconjunto de emisoras y los pesos de los coeficientes para cada variable. Lo siguiente es normalizar los pesos para lograr que la suma de los pesos sea del 100%, permitiendo que se distribuya el total del portafolio en los pesos proporcionales. Con eso, es posible realizar un producto punto entre los coeficientes y sus rendimientos, para obtener un valor final de rendimiento de todo el portafolio.

Una vez que se tiene la serie de rendimientos del portafolio, se toma el precio del índice a replicar, del primer día del año de estudio y se aplican los rendimientos obtenidos. De esa forma, podemos construir una serie en precios del nuevo portafolio y compararla contra la serie de precios del índice a replicar.

9.3 Aplicación del modelo para 2018

Para poder aplicar el algoritmo de *trading* para 2019, es necesario construir previamente un modelo para 2018, con el fin de obtener la media de la razón entre el portafolio generado y el mercado, así como las bandas de desviación estándar. Este primer modelo redujo el conjunto de 476 emisoras originales a 292. Es decir, el portafolio tuvo una reducción ligeramente superior al 38%.

Al construir la serie de precios y graficar ambos portafolios, se obtuvo lo siguiente:

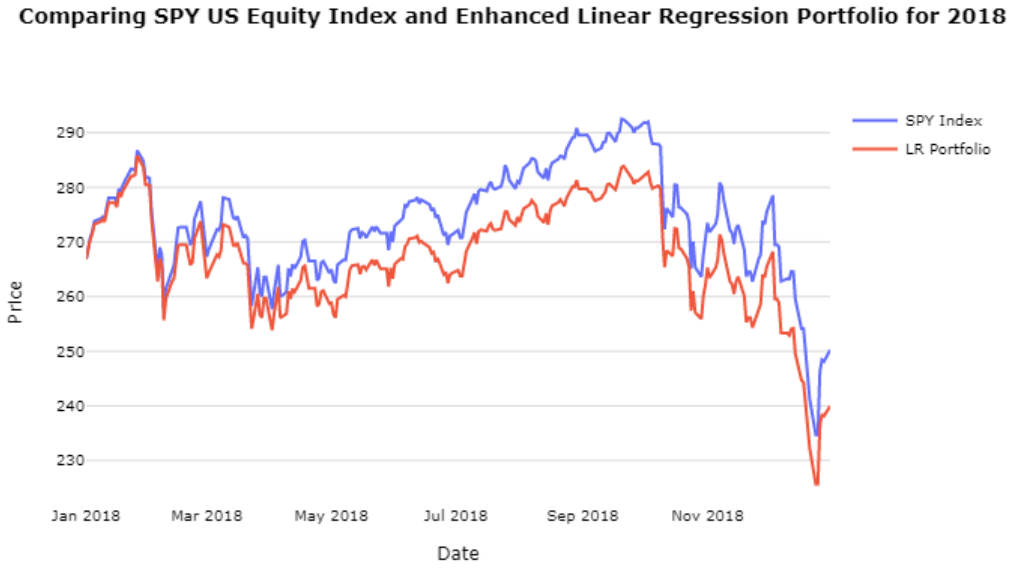


Figura 9.2. *Gráfica de precios para la comparación entre un portafolio de réplica del S&P 500 con un modelo de regresión lineal, utilizando backwards stepwise y el SPY Index, para el año 2018.*

9.3.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	LR Portfolio
Beta	1.0056	0.9630
Tracking Error	0.0008	0.0011
R^2 Ajustada	0.9971	0.9950
Volatilidad (S&P 1.05%)	1.06%	1.02%

Resultados riesgo/rendimiento	SPY Index	LR Portfolio
VaR 95% Porcentual	1.44%	1.72%
CVaR 95% Porcentual	1.81%	2.41%
Treynor's Ratio	-0.07	-0.11
Sharpe Ratio	-0.42	-0.67

Figura 9.3. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

Aunque en el capítulo de conclusiones se habla a detalle de estos resultados, se puede asumir que el portafolio generado con regresión lineal replica de manera adecuada al índice.[1]

Por último, para construir la razón entre el portafolio y el índice SPY, se dividen los precios del portafolio entre los precios del índice, se obtiene su media y se calculan dos desviaciones estándar para las bandas. Los resultados de este proceso para 2018 fueron: **Media**: 0.9762, **Banda superior:** 0.9961, y **Banda inferior:** 0.9563. Con estos datos, es posible construir la estrategia de *trading* para el siguiente año.

[1]Los criterios para determinar si replica o no al índice también se mencionan en el capítulo de conclusiones.

9.4 Aplicación del modelo para 2019

Para aplicar el modelo para el siguiente año, se repiten los mismos pasos con el objetivo de construir el portafolio, tomar métricas de ajuste con respecto al mercado y comparar su razón contra el índice SPY.

El modelo de regresión lineal redujo el número de emisoras de 468 a 282, luego de correr el análisis de colinealidad, es decir, una reducción de casi el 40%. Al correr la serie de precios y graficar contra el índice, se obtiene la siguiente gráfica:

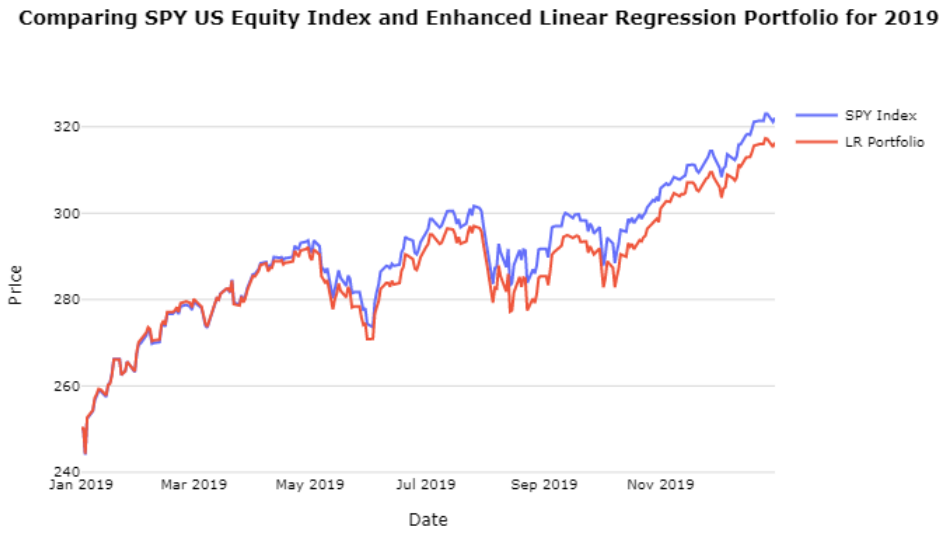


Figura 9.4. *Comparación de un portafolio de réplica del S&P 500 con un modelo de regresión lineal, utilizando backwards stepwise para la selección de variables para el año 2019, contra el SPY index.*

Se puede observar que las gráficas se separan con el paso del tiempo. Una explicación puede deberse a que el SPY Index tiene rebalanceos trimestrales, lo que significa que se hacen ajustes en los pesos de las emisoras durante el año, mientras que el portafolio creado para el trabajo asigna pesos al inicio del periodo y los mantiene durante todo el año.

9.4.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	LR Portfolio
Beta	1.0063	1.0047
Tracking Error	0.0007	0.0008
R^2 Ajustada	0.9955	0.9944
Volatilidad (S&P 0.77%)	0.78%	0.78%

Resultados riesgo/rendimiento	SPY Index	LR Portfolio
VaR 95% Porcentual	1.49%	1.36%
CVaR 95% Porcentual	1.86%	1.97%
Treynor's Ratio	0.27	0.25
Sharpe Ratio	2.22	2.04

Figura 9.5. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

En esta ocasión el portafolio compuesto por aproximadamente el 60% de emisoras, se acerca aún más a los valores de las métricas vistas para el SPY Index. Con esto es posible asumir que la hipótesis sobre si es posible crear un portafolio que replique al mercado utilizando un modelo de machine learning, es correcta.

Una posible extensión al presente trabajo sería analizar las métricas del portafolio de replica, si se realizaran rebalanceos periódicamente, para mantener el mayor nivel de precisión al replicar el índice. Lo siguiente es crear la razón del portafolio entre el SPY para crear la estrategia de inversión y, por último, correr la estrategia de *trading*.

9.4.2 Algoritmo de operación

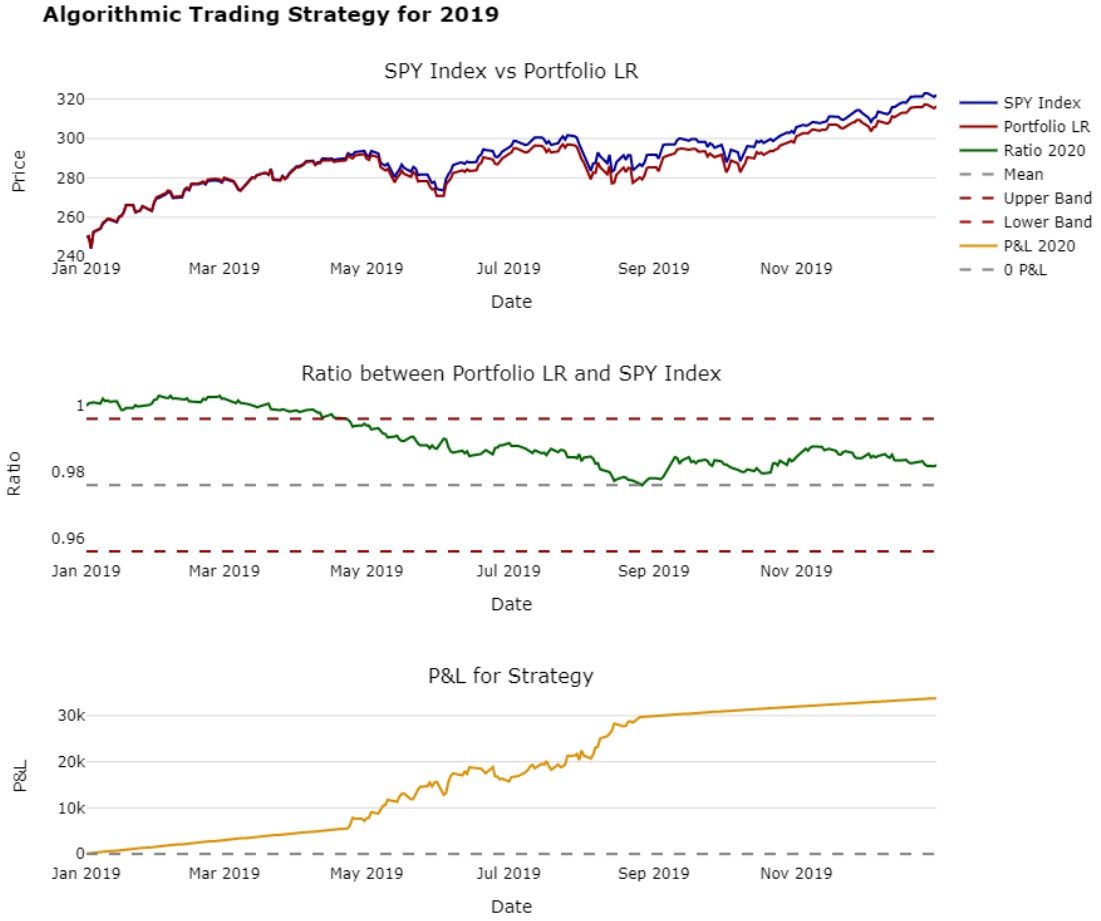


Figura 9.6. *Resultados finales para la estrategia del algoritmo de trading utilizando una estrategia de arbitraje estadístico entre un portafolio generado con regresión lineal que cuenta con aproximadamente el 60% de emisoras del índice, y el SPY Index, para el 2019.*

Se puede observar en la gráfica que el periodo de estudio comienza con la razón fuera de las dos desviaciones estándar, por encima de la media. El punto de entrada lo marca el 23 de abril. Antes de ese punto, se invertía un millón de dólares a la tasa de fondeo diario de Estados Unidos —un *US Treasury* de un año con capitalización diaria—. Por ese motivo es que se ve un *P&L*[2]

A partir de la fecha del punto de entrada, se venden títulos del portafolio generado y se compran títulos del *SPY*[3]. Vender en corto poco más de un millón de dólares[4] del portafolio generado, tomando el precio del día, equivale a $\mathbf{3,438}$ títulos. Por otro lado, con el dinero de la venta se pueden comprar $\mathbf{3,452}$ títulos del *SPY Index*, y aún se tiene el millón de dólares en efectivo para invertir a la tasa libre de riesgo.

Conforme avanza el tiempo, los títulos cambian de precio, lo que hace que tanto el valor del portafolio como el *P&L*, varíen.

[2] *P&L* son las siglas en inglés de *Profit and Loss*, que representa la utilidad o pérdida de un portafolio.

[3] Esto se determina así porque la razón se calcula como Portafolio LR/SPY, por lo que si se sale por encima de las dos desviaciones estándar, significa que el Portafolio LR está "caro" en valor relativo contra el *SPY*, por lo que se vende el Portafolio LR y se compra el *SPY*.

[4] El extra es por el rendimiento generado a la tasa libre de riesgo

La razón vuelve a la media el día 26 de agosto, por lo que se realiza el cierre de la operación. Se venden los títulos del *SPY* al precio de ese día y se compran los del portafolio generado, obteniendo una ganancia superior a los 19,000 dólares, mismos que se suman al rendimiento de la inversión libre de riesgo. El resto del periodo no tiene más señales, por lo que sólo se continúa con la inversión en fondeo diario.

Al final del año se tiene una utilidad superior a los 33,700 dólares en una estrategia de riesgo neutral. Con esto se confirma la segunda hipótesis, afirmando que se puede tener una estrategia de inversión redituable con un algoritmo de arbitraje estadístico, a partir del portafolio construido con el modelo paramétrico de *machine learning*.

Hasta este punto se sabe que la estrategia es redituable, pero ¿qué tan redituable es realmente? Para responder esta pregunta, podemos medir la volatilidad del portafolio, así como su *Sharpe Ratio* y su *Treynor Ratio*, y comparar los resultados con otras posibles inversiones en el mercado. Los resultados fueron los siguientes:

Algoritmo de trading RL vs Benchmarks	Vol	Treynor's Ratio	Sharpe Ratio
Algorithmo de Trading	**0.05%**	**4.05**	**2.34**
ACWI	0.75%	0.24	1.86
S&P 500 (SPY ETF)	0.79%	0.27	2.18
NASDAQ (QQQ ETF)	1.02%	0.29	2.25
Dow Jones (DIA ETF)	0.79%	0.22	1.67
Oro (GLD ETF)	0.73%	-0.65	1.41
Petróleo (Brent Comdty)	1.50%	-3.60	-0.98
Bitcoin (BTCUSD Curncy)	4.47%	-10.30	2.21

Figura 9.7. *Comparación de métricas de riesgo y rendimiento del algoritmo de trading para el 2019, contra otras opciones de inversión.*

Se puede apreciar que el portafolio construido por el algoritmo de *trading* tiene una volatilidad de 0.05%, lo cual hace sentido pues en la mayor parte del año, la única inversión del portafolio es en el UST de un año, el cual tiene volatilidad muy baja. Además, aun en el tiempo en que se tiene la inversión en los otros portafolios, la volatilidad de los rendimientos generados sigue siendo baja, por lo que en conjunto dan al portafolio una volatilidad igualmente baja.

Por otro lado, se tienen un *Treynor Ratio* y un *Sharpe Ratio* positivos y, de hecho, mayores a los de las otras inversiones. Particularmente, el índice de Treynor es significativamente elevado porque el denominador, la beta del portafolio contra el mercado, es muy baja.

Con estos resultados se puede concluir que el algoritmo propuesto en el presente trabajo hubiera sido la mejor inversión en el año, comparado con las inversiones alternas presentadas. Se llega a esta conclusión gracias a que el portafolio generado por el algoritmo de *trading* en el presente trabajo tiene la volatilidad más baja y, al mismo tiempo, el *Treynor Ratio*[5] y *Sharpe Ratio* más altos. Estos resultados indican que el portafolio que tuvo mejor rendimiento comparado con el riesgo asumido —para el año 2019— fue el portafolio del algoritmo de arbitraje estadístico.

A continuación se hace el mismo análisis pero para el año 2020, el cual presenta una volatilidad particularmente elevada debido a la pandemia de COVID-19.

[5]Es importante señalar que el *Treynor Ratio*, tanto del oro como del Bitcoin, no pueden tomarse en cuenta pues provienen de una beta negativa, y de acuerdo a la definición en Investopedia, un valor de índice de Treynor negativo proveniente de una beta negativa no es significativo.

9.5 Aplicación del modelo para 2020

Como se vio en el capítulo anterior, de las 505 empresas del índice, se redujo el espacio a 479 empresas, luego del análisis de colinealidad. Ahora, al aplicar el modelo de regresión lineal, ese espacio se reduce a 291, es decir, una reducción de nuevo de casi el 40%. Similar al año anterior, se analiza la serie de precios y se hace una gráfica contra el índice, obteniendo lo siguiente:

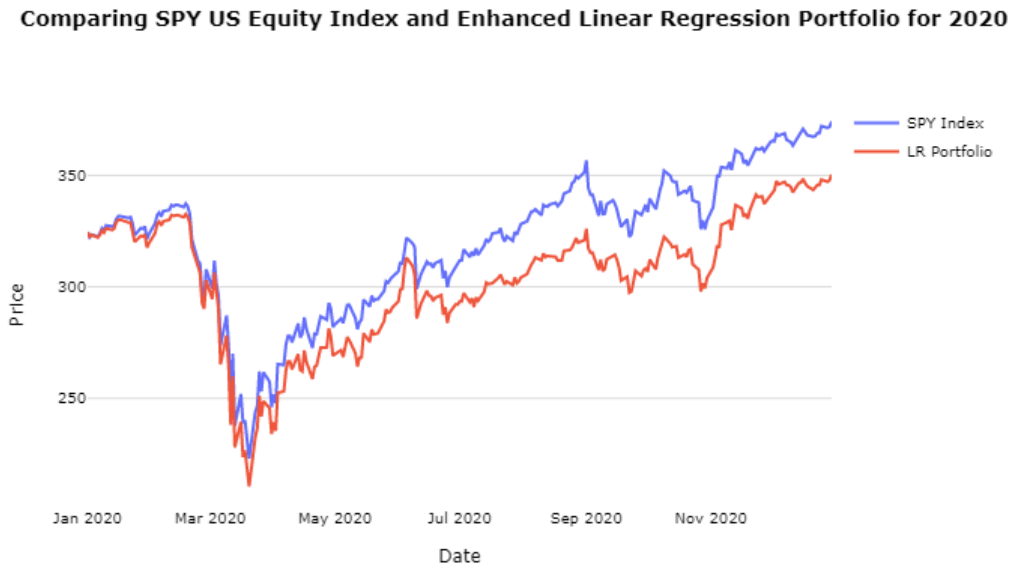


Figura 9.8. *Comparación de un portafolio de réplica del S&P 500 con un modelo de regresión lineal, utilizando backwards stepwise para la selección de variables para el año 2020, contra el SPY index.*

Se puede observar que las gráficas se separan con el paso del tiempo, igual que para el caso de 2019, e igual que en ese año, se puede asumir que tiene que ver con los rebalanceos que realiza el *SPY Index* y que no se realizan para el portafolio generado. Sin embargo, se puede apreciar que la diferencia en precios se vuelve mucho más significativa que el año anterior. Una teoría al respecto es que, debido a la elevada volatilidad que existió durante el 2020, una diferencia pequeña en la composición del portafolio ocasionaría una gran diferencia en los rendimientos finales del mismo.

9.5.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	LR Portfolio
Beta	0.9754	1.0348
Tracking Error	0.0014	0.00362
R^2 Ajustada	0.9982	0.9874
Volatilidad (S&P 2.15%)	2.09%	2.24%

Resultados riesgo/rendimiento	SPY Index	LR Portfolio
VaR 95% Porcentual	3.24%	3.66%
CVaR 95% Porcentual	4.06%	5.19%
Treynor's Ratio	0.15	0.07
Sharpe Ratio	0.45	0.21

Figura 9.9. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

Se puede observar que las métricas son positivas para el portafolio que replica el índice; sin embargo, están más alejadas de las métricas del portafolio existente en el mercado. Estos números quizás mejorarían con rebalanceos semestrales, o trimestrales, para ajustar los pesos y mantener vigente el portafolio.

Aun así, es posible asumir que, en periodos de elevada volatilidad, el modelo de regresión lineal múltiple podría no ser el mejor modelo a utilizar, pues aunque cumple con su objetivo dentro de lo que marcan las métricas, los resultados no son los mejores.

9.5.2 Algoritmo de operación

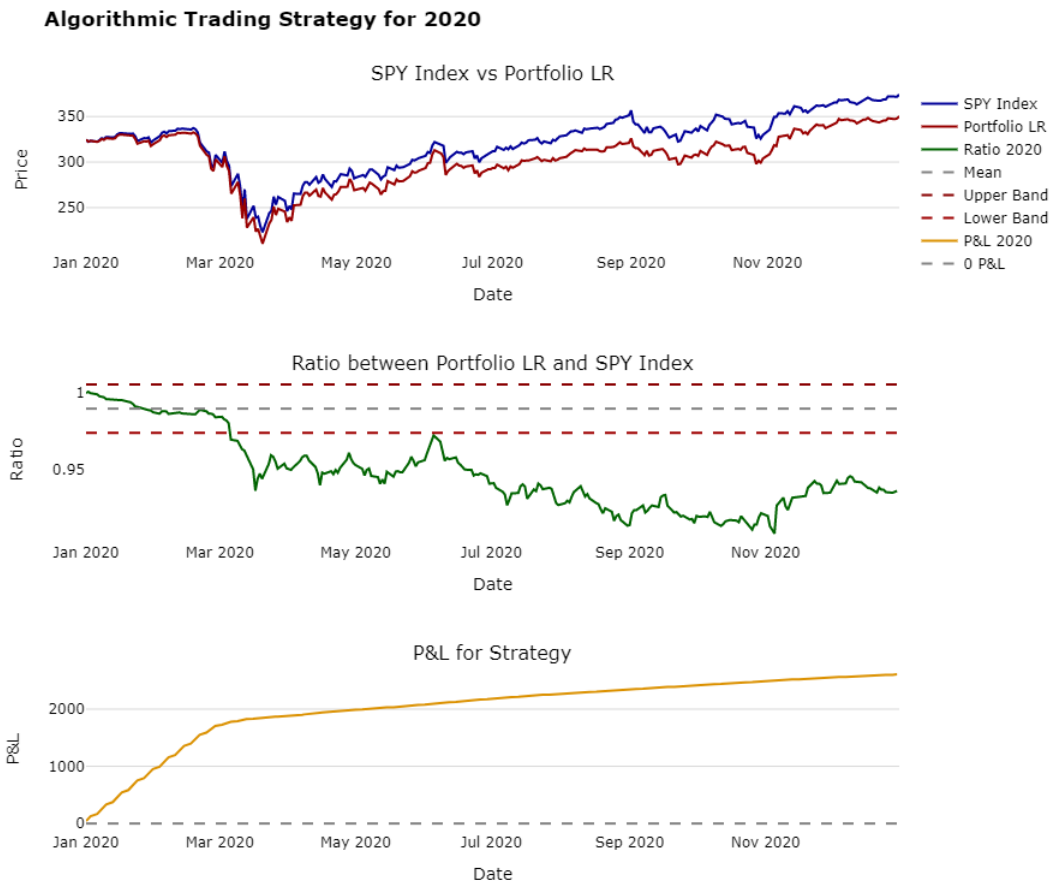


Figura 9.10. *Resultados finales para la estrategia del algoritmo de trading utilizando una estrategia de arbitraje estadístico entre un portafolio generado con regresión lineal que cuenta con aproximadamente el 60% de emisoras del índice, y el SPY Index, para el 2020.*

Es posible observar que la razón sale del rango en marzo, por la banda inferior; sin embargo, aunque se acerca al rango de nuevo en junio, no alcanza a entrar al canal por lo que no se realiza ninguna operación con el algoritmo. Por ese motivo, el portafolio se comporta exactamente igual que una inversión en el *US Treasury* de un año, con un P&L creciente y, relativamente constante[6].

Dado que el portafolio construido por el algoritmo de *trading* no es más que una inversión en la tasa libre de riesgo, se tiene un retorno en exceso[7] igual a cero, dando como resultado un *Treynor Ratio* y un *Sharpe Ratio* de cero igualmente. De la misma manera, al ser una inversión con rendimiento constante igual a la tasa libre de riesgo capitalizable diariamente, la volatilidad también resulta ser prácticamente cero.

El siguiente paso sería analizar los resultados de los otros tipos de inversiones y compararlos con los resultados del portafolio creado por el algoritmo.

[6]Se puede notar un cambio abrupto en la velocidad de crecimiento del P&L a partir de marzo 2020, y esto se debe a que fue a partir de ese momento en que la Reserva Federal de Estados Unidos comenzó a bajar las tasas de interés, para hacer frente a la crisis económica ocasionada por la pandemia del COVID 19.

[7]La diferencia entre el rendimiento del portafolio y la tasa libre de riesgo

Algoritmo de trading RL vs Benchmarks	Vol	Treynor's Ratio	Sharpe Ratio
Algorithmo de Trading	0.00%	0.00	0.00
ACWI	2.04%	0.15	0.44
S&P 500 (SPY ETF)	2.12%	0.16	0.47
NASDAQ (QQQ ETF)	2.26%	0.49	1.32
Dow Jones (DIA ETF)	2.32%	0.07	0.19
Oro (GLD ETF)	1.23%	2.69	1.26
Petróleo (Brent Comdty)	1.23%	-0.42	0.13
Bitcoin (BTCUSD Curncy)	4.23%	-6.80	3.74

Figura 9.11. *Comparación de métricas de riesgo y rendimiento del algoritmo de trading para el 2020, contra otras opciones de inversión.*

Claramente no es la mejor inversión que se pudo haber hecho en el año, pues el algoritmo no generó ninguna operación que pudiera ser redituable, generando que la única inversión fuera en la tasa libre de riesgo. Uno de los principales factores externos que afectaron el desempeño de invertir solo en la tasa libre de riesgo, fue la decisión de la Reserva Federal de Estados Unidos de utilizar algo que se conoce como *Quantitative Easing*[8], generando una baja en las tasas de interés, implicando menor rendimiento para los bonos del tesoro, o *US Treasuries*.

Aunado a eso, el aumento de liquidez en el mercado generado tanto por la Reserva Federal como por los incentivos entregados por el gobierno, provocaron una inflación de precios en el mercado accionario, principalmente en el sector de tecnología, lo cual se puede apreciar con los resultados del *NASDAQ*, el cual fue la inversión que tuvo el mejor resultado en el índice de *Sharpe*, luego del *Bitcoin*.

El siguiente mejor resultado fue el del oro, el cual históricamente se ha visto beneficiado en épocas de crisis debido a que es considerado un refugio seguro, o *safe haven* en inglés, por no perder valor ante decisiones de política monetaria que afecten la tasa de interés y el cual, históricamente, ha mantenido su valor en el tiempo.

El mejor resultado en índice de Sharpe fue el del Bitcoin, la criptomoneda[9] más popular y con mayor capitalización de mercado hasta el momento, con un índice de Sharpe de 3.74. Aunque es un resultado muy positivo, y más en comparación con las otras inversiones, algunos inversionistas podrían negarse a invertir en dicho activo debido a la elevada volatilidad que representa este tipo de inversión (particularmente ese año fue de 4.23%).

Pese a que se tiene un resultado positivo, que puede corroborar la hipótesis de poder crear un algoritmo de arbitraje estadístico que pueda brindar rendimientos positivos, la realidad es que no sería la mejor inversión en un entorno de crisis, pues la premisa del algoritmo de tener dos activos altamente correlacionados, no es enteramente clara al no estar seguros si el portafolio generado replica adecuadamente al índice. En los siguientes capítulos se analiza otro tipo de modelo que, se espera, tenga mejores resultados que el modelo de regresión lineal, y posteriormente, se analizarán los resultados generales.

[8]También conocido simplemente como *QE*, es una forma poco convencional de política monetaria, en el que un banco central compra bonos de largo plazo, para proveer al mercado de liquidez, incentivando los préstamos y la inversión. Asimismo, esta compra de activos también genera que se bajen las tasas de interés al agregar demanda por los bonos.

[9]Una criptomoneda es un tipo de moneda digital, o dinero virtual, protegido por criptografía, lo cual la vuelve casi imposible de falsificar. La mayoría son monedas descentralizadas.

Capítulo 10

Desarrollo del modelo no paramétrico

10.1 Introducción

En este capítulo se construye y aplica el modelo de aprendizaje supervisado no paramétrico, un *Random Forest*, con toda la teoría previamente vista. Para lograrlo, el primer paso es repetir los pasos del capítulo 8, la parte de análisis y exploración de datos, así como el análisis de colinealidad, comenzando por el año 2018, para poder construir las bases que servirán para el algoritmo en 2019.

Posteriormente, se crea una función para realizar cross validation en forma de bloques, de acuerdo a lo visto en capítulos anteriores, tomando como base la función de *TimeSeriesSplit*. Se pueden apreciar las diferencias entre los diferentes tipos de cross validation en la siguiente imagen:

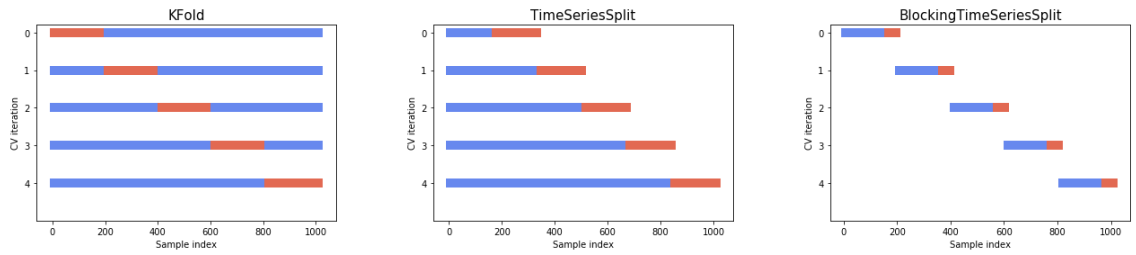


Figura 10.1. *Los datos azules son datos de entrenamiento, y los datos rojos son datos de prueba. En las imágenes se ve la diferencia del cómo cada método utiliza los datos para validar el modelo. Izquierda: K-Fold Cross Validation. Centro: Time Series Split. Derecha: Blocking Time Series Split. El método utilizado en el presente trabajo es el del lado derecho.*

Para construir el modelo, se utiliza la función **Random Forest Regressor**[1], que construye modelos de *Random Forest* con parámetros determinados y obtiene la proyección como el promedio de los resultados de los árboles, como se vio en el capítulo de modelos no paramétricos, y **Grid Search CV**. La primera es la base del modelo, y la segunda es para poder pasar un grupo de parámetros e ir probando con diferentes tamaños de árboles, hojas, profundidad, entre otros parámetros, para seleccionar el mejor modelo.

[1] *Random Forest* Regressor toma subconjuntos de prueba $m = \sqrt{p}$ variables por default.

En el presente trabajo, se prueba para diferentes tipos de árboles: 1, 5 y 7 árboles, con 100, 200, 300 y 500 hojas, con y sin *bootstrap*. La finalidad del código es ajustar los distintos tipos de *Random Forest* con esas características, y tomar la mejor combinación. Una vez que se encontró la mejor combinación de parámetros con *GridSearchCV*, se corre el modelo con dichos parámetros, siendo este el modelo final[2].

Por construcción, un modelo de *Random Forest* no elimina variables; sin embargo, cada variable contiene pesos o *importancias* que sirven para seleccionar variables. Utilizar todas las variables significaría utilizar el 100% de la suma de importancias. Para poder tener un modelo reducido en este ejercicio, se realizó una suma acumulativa y se mantuvieron las que acumularon un determinado porcentaje.

Para seleccionar el porcentaje a mantener, se utilizó una optimización del modelo, eliminando una por una las variables, de menor a mayor importancia, y guardando su *tracking error*. El modelo que tuviera menor *tracking error* contra el índice es el que determinaría el porcentaje de emisoras a mantener. El mejor resultado se obtuvo al mantener 19% del total de pesos, y el restante se invirtió a la tasa libre de riesgo.

Una vez que se determina el mejor subconjunto de emisoras, se vuelve a correr el modelo de *Random Forest* para ajustar con estos datos. Este es el portafolio generado sobre el cual se trabaja y se analizan métricas, estrategias y rendimientos. A continuación se puede observar los datos del modelo para 2018:

Equity Ticker	AAL US Equity	AAPL US Equity	ABBV US Equity	ABC US Equity	ABT US Equity	ACN US Equity	ADI US Equity	ADSK US Equity	AEE US Equity	AES US Equity	...	WMT US Equity	WU US Equity	WY US Equity
Dates														
2018-01-01	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000000
2018-01-02	0.018283	0.017734	0.017426	0.023890	0.029693	0.004887	0.013943	0.021610	-0.010223	0.004606	...	-0.001622	0.003676	0.001134
2018-01-03	-0.012342	-0.000163	0.015528	0.003715	0.002209	0.004605	0.012330	0.020878	-0.005151	-0.000920	...	0.008685	-0.000524	0.008744
2018-01-04	0.006285	0.004634	-0.005719	-0.002227	-0.001699	0.011771	-0.001095	0.024296	-0.011427	-0.003687	...	0.000905	0.019731	-0.008177
2018-01-05	-0.000380	0.011309	0.017258	0.012032	0.002886	0.008215	0.004044	-0.011036	-0.000697	0.003687	...	0.005910	0.057444	-0.001417

5 rows × 292 columns

Figura 10.2. *DataFrame de rendimientos del subconjunto óptimo de acciones obtenido con un Random Forest, para 2018.*

Se puede observar que para 2018, se conservaron 51 emisoras de un total de 476; una reducción de casi el 90% del total del universo de emisoras. Más adelante se analizan las métricas de riesgo y rendimiento, así como las métricas para verificar que el modelo replique adecuadamente al índice; sin embargo, si se quiere llevar a la práctica este modelo, sería prudente agregar un análisis de composición sectorial, pues al reducir tanto el campo muestral de las emisoras, es posible que la diversificación no sea la más adecuada, lo que podría añadir riesgo al portafolio.

Una vez que se tiene la composición del portafolio, se obtiene su serie de rendimientos para el portafolio completo. Posteriormente, se toma el primer valor del *SPY Index* y se aplican los rendimientos obtenidos para construir una serie de precios.

A continuación se hace el análisis de la aplicación del modelo para 2018.

[2]Para 2018, el mejor modelo fue un *Random Forest* sin *bootstrapping*, de 7 árboles y 100 hojas por árbol.

10.2 Aplicación del modelo para 2018

Como se mencionó previamente, este modelo redujo el conjunto de emisoras de 476 emisoras originales a 51; una reducción de casi el 90%. A primera instancia, se puede pensar que con una reducción tan elevada del número de emisoras, los resultados del modelo resultarían desfavorables, o por lo menos no mejores a los de regresión lineal; sin embargo, al construir la serie de precios y graficar ambos portafolios, se obtuvo:

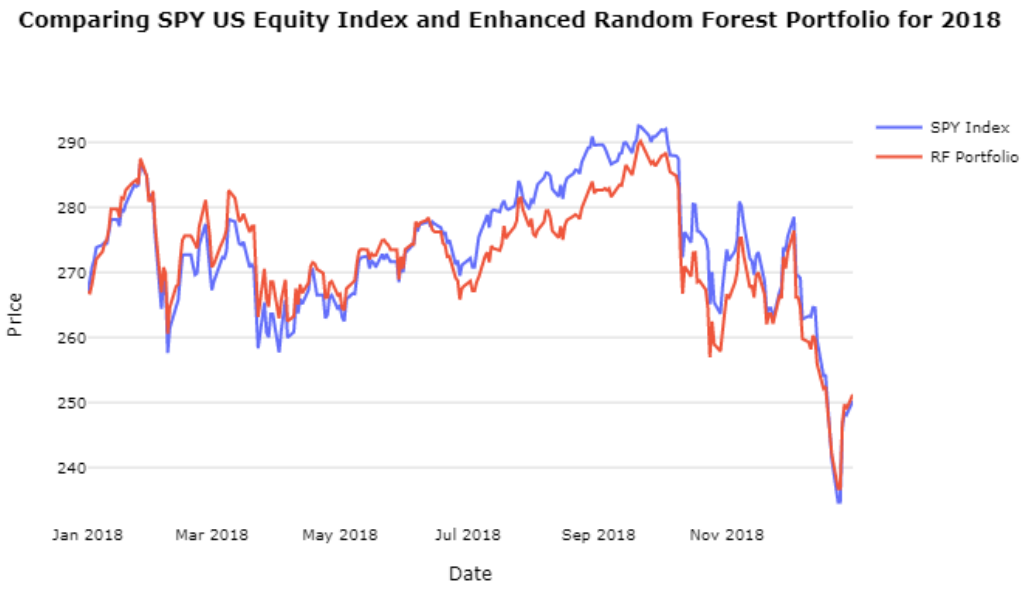


Figura 10.3. *Gráfica de precios para la comparación entre un portafolio de réplica del S&P 500 con un modelo de Random Forest y el SPY Index, para el año 2018.*

10.2.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	RF Portfolio
Beta	1.0056	0.9786
Tracking Error	0.0008	0.0026
R^2 Ajustada	0.9971	0.9699
Volatilidad (S&P 1.05%)	1.06%	1.06%

Resultados riesgo/rendimiento	SPY Index	RF Portfolio
VaR 95% Porcentual	1.44%	1.77%
CVaR 95% Porcentual	1.81%	2.49%
Treynor's Ratio	-0.07	-0.07
Sharpe Ratio	-0.42	-0.39

Figura 10.4. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

En términos del modelo, se puede asumir que cumple con replicar adecuadamente al índice, aunque el detalle se revisa en el capítulo de conclusiones. También se puede apreciar que términos de la β y la volatilidad, resulta mejor que el modelo de regresión lineal, aunque no tiene mejores resultados en lo referente a *tracking error* y R^2 ajustada.

Por otro lado, se puede apreciar que tanto el VaR como el CVaR son mayores en el portafolio nuevo, por lo que se asume que tiene colas más pesadas en cuanto a rendimientos (lo cual también era de esperarse debido a que presenta una volatilidad mayor). Finalmente, tuvo mejores resultados en términos de índices de Sharpe y de Treynor.

Los resultados generados al construir la razón del ETF de mercado entre el portafolio generado para 2018, fueron: **Media**: 1.0021, **Banda superior:** 1.0282, y **Banda inferior:** 0.9760.

10.3 Aplicación del modelo para 2019

Al igual que para el modelo en 2018, se deben repetir los mismos pasos con el objetivo de construir el portafolio, tomar métricas de ajuste con respecto al mercado y comparar su razón contra el índice SPY. Además, se comienza con las mismas 468 emisoras luego del análisis de correlación, igual que en el caso de regresión lineal, solo que esta vez en lugar de reducir a 282, el modelo de *Random Forest* redujo el espacio muestral a 45, es decir, una reducción de más del 90%. El mejor modelo obtenido fue un *Random Forest* sin *bootstrapping*, de siete árboles y 100 hojas.

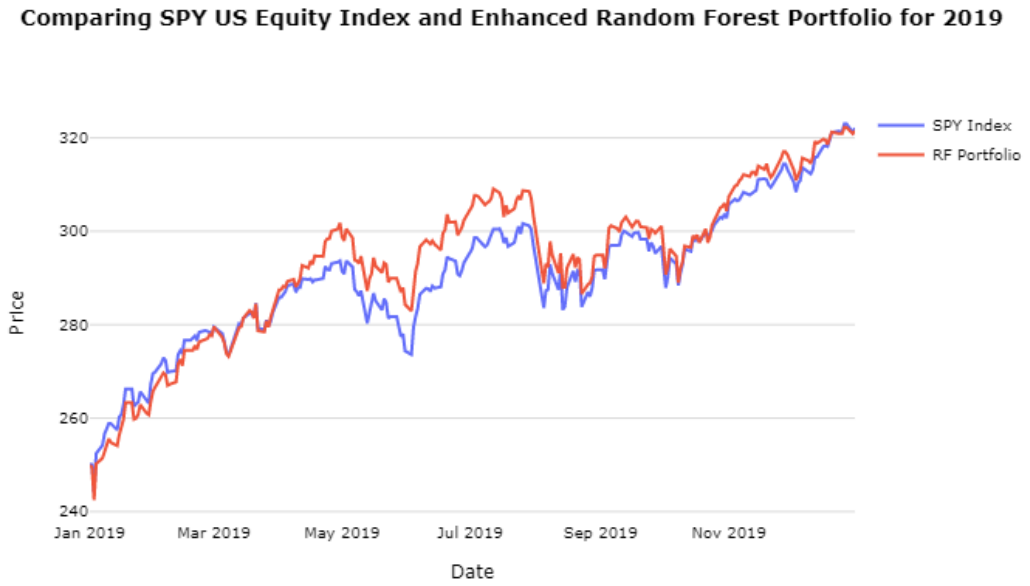


Figura 10.5. *Comparación de un portafolio de réplica del S&P 500 con un modelo de Random Forest para la selección de variables para el año 2019, contra el SPY index.*

10.3.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	RF Portfolio
Beta	1.0063	1.0093
Tracking Error	0.0007	0.0021
R^2 Ajustada	0.9955	0.9665
Volatilidad (S&P 0.77%)	0.78%	0.80%

Resultados riesgo/rendimiento	SPY Index	RF Portfolio
VaR 95% Porcentual	1.49%	1.23%
CVaR 95% Porcentual	1.86%	1.77%
Treynor's Ratio	0.27	0.27
Sharpe Ratio	2.22	2.10

Figura 10.6. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

Con estos resultados, es posible aceptar que el portafolio replica al índice de manera adecuada. De manera interesante, aunque las gráficas se acercan más al final del periodo, en los momentos de mayor volatilidad, la diferencia entre el portafolio modelo y el índice de mercado es mucho mayor que para el modelo de regresión lineal, lo que provoca que igualmente los resultados no sean tan favorables como el modelo anterior.

Esto puede ser atribuible a la gran diferencia en el número de emisoras que fueron eliminadas del portafolio, provocando que en periodos de elevada volatilidad, las diferencias fueran mucho mayores.[3]

[3]Un estudio posterior podría ser el limitar la reducción de emisoras, para corroborar la teoría de que las diferencias en la gráfica, como en los resultados, se deben al tamaño de la reducción del espacio muestral, y no a la selección de la muestra.

10.3.2 Algoritmo de trading

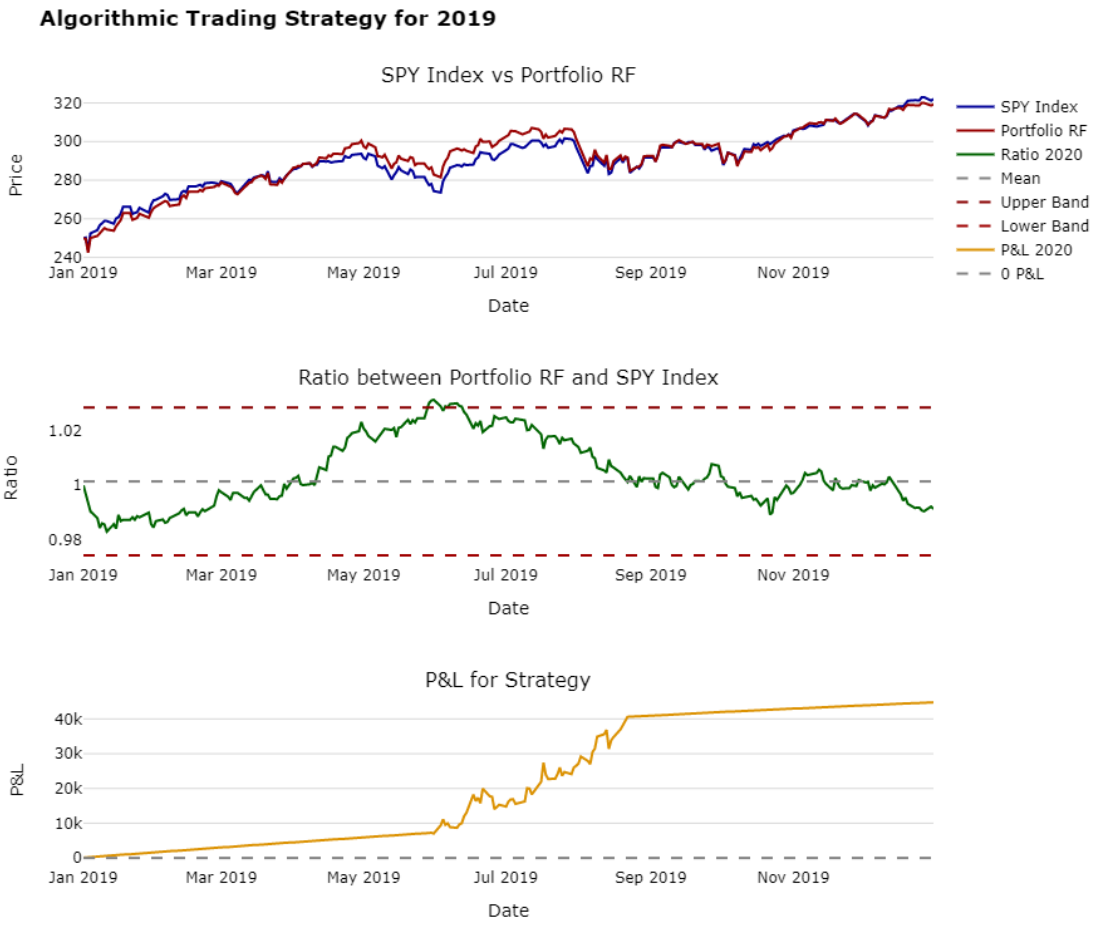


Figura 10.7. *Resultados finales para la estrategia del algoritmo de trading utilizando una estrategia de arbitraje estadístico entre un portafolio generado con un Random Forest que cuenta con aproximadamente el* 10% *de emisoras del índice, y el SPY Index, para el 2019.*

Gráficamente se puede observar a la razón en medio de las bandas de desviación estándar, hasta finales de mayo. En este punto, la razón sale por arriba de las dos desviaciones estándar, lo que significa que se debe comprar el índice SPY y vender el portafolio generado. Como en el modelo anterior, previo al punto de entrada a finales de mayo, se invertía un millón de dólares a la tasa de fondeo diario de Estados Unidos.

Vender en corto un millón de dólares del portafolio generado, tomando el precio del 29 de mayo, equivale a $\mathbf{3,496}$ títulos. Por otro lado, con el dinero de la venta se pueden comprar $\mathbf{3,604}$ títulos del *SPY Index*, y aún se tiene el millón de dólares en efectivo para invertir a la tasa libre de riesgo.

Como en el modelo anterior, el *P&L* varía conforme va cambiando el precio de los títulos, hasta que la razón cruza por debajo de la media el 22 de agosto. Hasta este punto, la estrategia generó poco más de $30,170$ dólares de rendimiento, que se suman al rendimiento que genera por la inversión a la tasa libre de riesgo.

La razón se mantiene dentro de las bandas de desviaciones estándar, por lo que no se realiza otra operación más que la inversión a la tasa libre de riesgo. Al concluir el periodo de un año, se obtiene un rendimiento final superior a los 44,780 dólares. Este rendimiento es superior, en términos absolutos, al generado por el algoritmo de *trading* que utilizó un modelo de regresión lineal. Pero, ¿cómo se compara en términos relativos con respecto a otras posibles inversiones? Al comparar la volatilidad, el *Sharpe ratio* y el *Treynor ratio*, se obtienen los siguientes resultados:

Algoritmo de trading RL vs Benchmarks	Vol	Treynor's Ratio	Sharpe Ratio
Algorithmo de Trading	**0.10%**	**2.46**	**1.92**
ACWI	0.75%	0.24	1.86
S&P 500 (SPY ETF)	0.79%	0.27	2.18
NASDAQ (QQQ ETF)	1.02%	0.29	2.25
Dow Jones (DIA ETF)	0.79%	0.22	1.67
Oro (GLD ETF)	0.73%	-0.65	1.41
Petróleo (Brent Comdty)	1.50%	-3.60	-0.98
Bitcoin (BTCUSD Curncy)	4.47%	-10.30	2.21

Figura 10.8. *Comparación de métricas de riesgo y rendimiento del algoritmo de trading para el 2019, contra otras opciones de inversión.*

En términos del índice de Sharpe, una inversión en el algoritmo del presente capítulo hubiera tenido menor resultado al presentado por una inversión 100% en el *S&P 500*, o en el *NASDAQ*, o incluso en Bitcoin, por lo que uno podría pensar que esas otras inversiones pudieron ser mejores.

Sin embargo, es importante considerar el universo completo de métricas. Si se toma también en consideración el índice de Treynor, se puede observar que la inversión en el algoritmo de *trading*, utilizando el portafolio generado con *Random Forest*, tiene el mayor índice de Treynor, dentro de los cuáles, su índice es significativo[4]. Es decir, que la inversión en el algoritmo de *trading* resulta más atractiva que las inversiones en el *S&P 500* y en el *NASDAQ*.

Otro punto a considerar es la volatilidad. La inversión en el algoritmo de *trading* presenta una volatilidad de 0.10%, la cual se encuentra muy por debajo de las otras inversiones, y es drásticamente menor a la volatilidad de una inversión en Bitcoin. Por este motivo, aunque una inversión en Bitcoin tiene un índice de Sharpe ligeramente superior (2.21 vs 1.92), la gran diferencia en volatilidad (4.47% vs 0.10%), permite asumir que una inversión en el algoritmo de trading sería más adecuada para el público inversionista en general.

Por último, es importante también comparar las métricas del presente algoritmo (Vol. 0.10%, T.R. 2.46 y S.R. 1.92) con las del algoritmo anterior (Vol. 0.05%, T.R. 4.05 y S.R. 2.34). En un primer análisis, se puede concluir que las métricas del algoritmo que utiliza un portafolio con regresión lineal son mejores; sin embargo, es importante considerar que se obtuvo mejor rendimiento absoluto con el algoritmo que utiliza el portafolio de *Random Forest*, por lo que se puede concluir que cualquiera de los dos algoritmos pudiera resultar atractivo según el perfil individual de riesgo y rendimiento del inversionista, sin haber uno mejor que otro.

A continuación, se analiza el periodo de crisis de 2020.

[4]Se excluyen las inversiones cuyo índice de Treynor es negativo pero su Sharpe es positivo, pues esto indica una beta negativa, la cual no necesariamente representa una mala inversión.

10.4 Aplicación del modelo para 2020

Del mismo modo en que se realizó el modelo para los años anteriores, en este modelo se deben repetir los mismos pasos con el objetivo de construir el portafolio, tomar métricas de ajuste con respecto al mercado y comparar su razón contra el índice SPY. Se comienza con las mismas 479 emisoras luego del análisis de correlación, igual que en el caso de regresión lineal. Igual que en 2018, el modelo de *Random Forest* redujo mucho más el espacio muestral que el modelo de regresión lineal, reduciendo el espacio a 35 emisoras, en lugar de 291, es decir, de nuevo una reducción de más del 90%. De manera interesante, el modelo obtenido fue igual que para 2018, un *Random Forest* sin *bootstrapping*, de siete árboles y 100 hojas.[5]

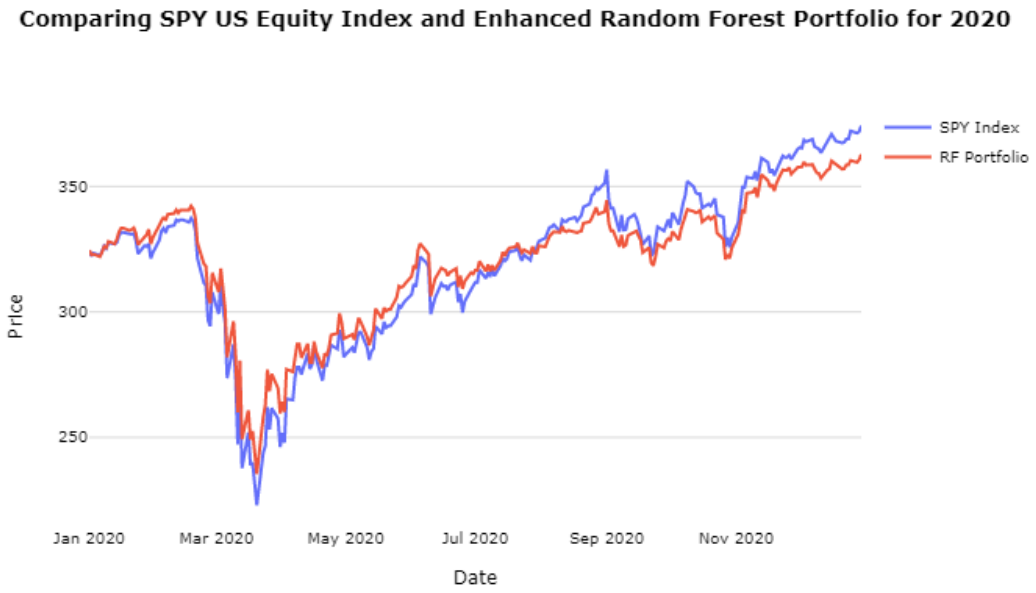


Figura 10.9. *Comparación de un portafolio de réplica del S&P 500 con un modelo de Random Forest para la selección de variables para el año 2020, contra el SPY index.*

10.4.1 Resultados de la construcción del portafolio

Resultados métricas vs S&P 500	SPY Index	RF Portfolio
Beta	0.9754	0.8829
Tracking Error	0.0014	0.00419
R^2 Ajustada	0.9982	0.9851
Volatilidad (S&P 2.15%)	2.09%	1.92%

Resultados riesgo/rendimiento	SPY Index	RF Portfolio
VaR 95% Porcentual	3.24%	3.11%
CVaR 95% Porcentual	4.06%	4.42%
Treynor's Ratio	0.15	0.13
Sharpe Ratio	0.45	0.37

Figura 10.10. ***Tabla 1****: Resultados de métricas para replicar un portafolio vs el S&P 500 del portafolio generado y el SPY Index.* ***Tabla 2****: Resultados de métricas de riesgo y rendimiento de ambos portafolios.*

Igual que para los años anteriores, se puede concluir que el portafolio generado por el modelo de *Random Forest*, replica de manera correcta al índice. También se observa que, al igual que el año anterior, a pesar de tener muy buenas métricas de réplica, no son mejores que las de regresión lineal, lo cual también puede explicarse por la reducción tan grande que se hizo del espacio muestral.

A pesar de que el año presentó una volatilidad muy elevada, el portafolio compuesto por un menor número de empresas presenta una volatilidad menor; un punto interesante a analizar en futuros estudios sobre los modelos.

[5]Se podría analizar por qué este modelo resultó ser el mejor en ambos casos, en un estudio posterior sobre los modelos.

10.4.2 Algoritmo de trading

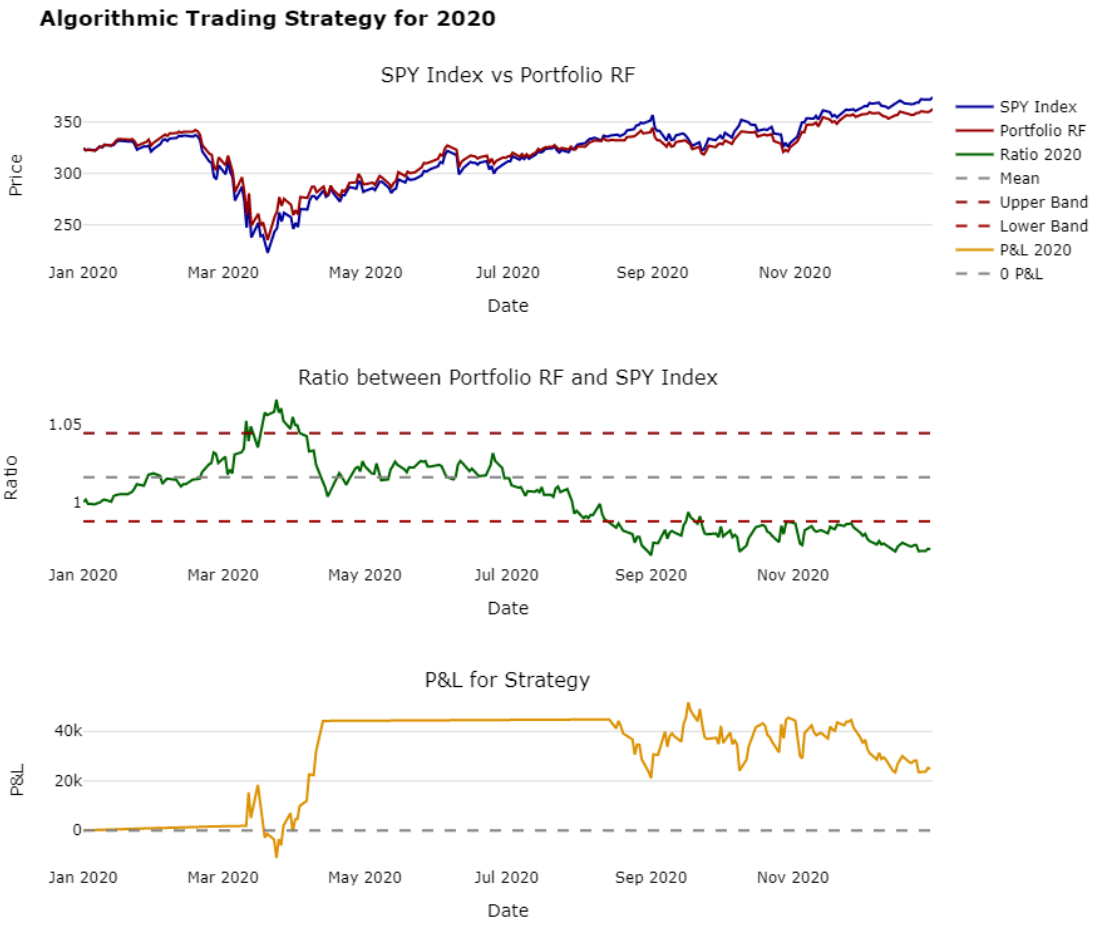


Figura 10.11. *Resultados finales para la estrategia del algoritmo de trading utilizando una estrategia de arbitraje estadístico entre un portafolio generado con un Random Forest que cuenta con aproximadamente el* 10% *de emisoras del índice, y el SPY Index, para el 2020.*

Para este caso, tenemos dos momentos de operación. El primero comienza cuando la razón sale por la parte superior de las bandas, a mediados de marzo. En ese momento, como en casos anteriores se divide poco más de un millón de dólares, por el rendimiento obtenido en el fondeo hasta esta fecha. Al precio de ese día, se compran **4,054** títulos del SPY, y se venden en corto **3,853** títulos del portafolio construido. Esta operación se cierra un mes después, a mediados de abril, con una utilidad ligeramente superior a los 42 mil dólares.

Un caso particular del algoritmo para este año es que marcó otra señal de entrada, pero esta vez por debajo de la banda inferior, lo que significa que se compran 3,142 títulos del portafolio generado y se venden 3,100 títulos del SPY, con precios de ese día, y tomando el valor total del portafolio, que para ese momento es poco más de un millón 44 mil dólares.

Dado que la razón no regresa a la media, esta operación continúa abierta para finales de año, y al irse alejando de las bandas, va perdiendo valor, lo que hace que el P&L disminuya y dando como resultado un portafolio a finales de año con un valor de casi un millón 25 mil dólares.

Pese a que el punto máximo de utilidad del algoritmo fue antes de la segunda operación, alrededor de los 44 mil dólares, y posteriormente bajó a poco más de 24,800 dólares, el rendimiento es superior al generado por el algoritmo de *trading* que utilizó un modelo de regresión lineal. Para conocer más sobre el resultado, comparamos la volatilidad, el *Sharpe ratio* y el *Treynor ratio*, similar a lo que se hizo en el ejercicio anterior. Los resultados fueron los siguientes:

Algoritmo de trading RF vs Benchmarks	Vol	Treynor's Ratio	Sharpe Ratio
Algorithmo de Trading	**0.33%**	**0.95**	**0.42**
ACWI	2.04%	0.15	0.44
S&P 500 (SPY ETF)	2.12%	0.16	0.47
NASDAQ (QQQ ETF)	2.26%	0.49	1.32
Dow Jones (DIA ETF)	2.32%	0.07	0.19
Oro (GLD ETF)	1.23%	2.69	1.26
Petróleo (Brent Comdty)	1.23%	-0.42	0.13
Bitcoin (BTCUSD Curncy)	4.23%	-6.80	3.74

Figura 10.12. *Comparación de métricas de riesgo y rendimiento del algoritmo de trading para el 2020, contra otras opciones de inversión.*

Al igual que el caso de 2019, la inversión en el algoritmo del presente capítulo, en términos de índice de Sharpe, hubiera tenido menor resultado al presentado por una inversión 100% en casi cualquiera de las otras opciones: *ACWI*, *S&P 500*, *NASDAQ*, *Oro* o incluso en Bitcoin.

Sin embargo, de la misma forma que sucedió en 2019, si se considera también el índice de Treynor, se puede observar que la inversión en el algoritmo de *trading* utilizando el portafolio generado con *Random Forest*, tiene el mayor índice de Treynor dentro de los cuáles su índice de es significativo, con excepción de una inversión en un ETF que siga el precio del oro. El tema particular del oro podría ser debido a que, una vez entrando de lleno en la pandemia, el oro resulta ser un *safe haven*[6], o refugio seguro en español.

Finalmente, observamos la volatilidad de las distintas opciones de inversión. La inversión en el algoritmo de *trading* presenta una volatilidad de 0.33% la cual, igual que el año anterior, se encuentra muy por debajo de las otras inversiones. Resulta una muy buena métrica, considerando el enorme salto en volatilidad que tuvieron las otras opciones entre 2019 y 2020. Por este motivo, podría ser una muy buena opción a considerar.

En este caso, no es posible comparar con una inversión con el algoritmo de regresión lineal, pues no presentó señales de entrada. Lo que sí es importante mencionar, y aplica para todos los casos vistos en el presente trabajo, es que una inversión con volatilidad similar podría ser la inversión en la tasa libre de riesgo. Sin embargo, tomando como ejemplo el 2020 con el algoritmo de *Random Forest*, si bien la volatilidad sería similar, el rendimiento de la tasa libre de riesgo de Estados Unidos fue de 0.26%, mientras que el algoritmo generó 2.48%, lo cual es una diferencia bastante amplia.

Con esto concluye el capítulo de aplicación del modelo de *Random Forest*, dejando únicamente el siguiente capítulo para conclusiones generales y análisis de algunos resultados, así como para posibles investigaciones futuras y bibliografía.

[6]Un safe haven es un tipo de inversión que se espera que mantenga, o incluso incremente su valor, durante tiempos de elevada volatilidad y turbulencia en los mercados.

Capítulo 11

Resultados y conclusiones

11.1 Comprobación de hipótesis

11.1.1 Modelos de machine learning para la creación de ETFs de réplica

Resultados

Resultados métricas vs S&P 500		SPY Index	LR Portfolio	RF Portfolio
2018	Beta	1.0056	0.9630	0.9551
	Tracking Error	0.0008	0.0011	0.0026
	R^2 Ajustada	0.9971	0.9950	0.9693
	Volatilidad (S&P 1.05%)	1.06%	1.02%	1.03%
2019	Beta	1.0063	1.0047	1.0214
	Tracking Error	0.0007	0.0008	0.0021
	R^2 Ajustada	0.9955	0.9944	0.9674
	Volatilidad (S&P 0.77%)	0.78%	0.78%	0.77%
2020	Beta	0.9754	1.0348	0.9167
	Tracking Error	0.0014	0.0036	0.0039
	R^2 Ajustada	0.9982	0.9874	0.9854
	Volatilidad (S&P 2.15%)	2.09%	2.24%	1.99%

Métricas de riesgo y rendimiento		SPY Index	LR Portfolio	RF Portfolio
2018	VaR 95% Porcentual	1.44%	1.72%	1.73%
	CVaR 95% Porcentual	1.81%	2.41%	2.43%
	Treynor's Ratio	-0.07	-0.11	-0.07
	Sharpe Ratio	-0.42	-0.67	-0.37
2019	VaR 95% Porcentual	1.49%	1.36%	1.24%
	CVaR 95% Porcentual	1.86%	1.97%	1.79%
	Treynor's Ratio	0.27	0.25	0.26
	Sharpe Ratio	2.22	2.04	2.06
2020	VaR 95% Porcentual	3.24%	4.19%	3.23%
	CVaR 95% Porcentual	4.06%	5.94%	4.59%
	Treynor's Ratio	0.15	0.07	0.13
	Sharpe Ratio	0.45	0.21	0.37

Figura 11.1. *Resultados de las métricas de los portafolios creados con modelos de machine learning para 2018, 2019 y 2020.*

El primer resultado del trabajo tiene que ver con la hipótesis de poder construir un portafolio que replique al mercado, el cual se mide con las métricas mencionadas en capítulos anteriores, y que se pueden analizar en la figura 11.1.

De acuerdo con la empresa Zephyr Associates[1], el tracking error *típico* para considerar que un fondo del tipo *enhanced index funds*[2] replique al mercado se encuentra entre 1% y 2%. Para el caso de administradores de portafolios activos un tracking error típico está entre 4% y 7%.

Luego de analizar los resultados, es posible concluir que se puede construir un portafolio que replique al mercado utilizando un modelo de aprendizaje automático y un subconjunto de variables que componen el índice, gracias a que las métricas obtenidas se encuentran dentro de los parámetros aceptables. Además, se comprueba también con los gráficos obtenidos en capítulos anteriores, donde se pueden apreciar los comportamientos similares entre los portafolios.

[1] Empresa líder en análisis de software para inversiones.

[2] Un Enhanced Index Fund es un fondo que intenta obtener mejores resultados que un índice tomando como base las mismas empresas pero con diferentes pesos.

Comentarios sobre los resultados

Algunas de las diferencias que se dan entre los portafolios, y el por qué el SPY parece tener mejores métricas que los generados en el trabajo, se deben a diversos factores, siendo el principal el número de emisoras. Era de esperar que si el SPY utiliza las 500 emisoras del S&P, tendría mejores métricas de tracking error, beta, R^2 ajustada y volatilidad, que las de los portafolios generados.

Además de eso, el tener menos emisoras también impactó en las colas de la distribución de rendimientos, como se puede observar en las métricas de VaR y CVaR: los portafolios generados presentan colas más pesadas para los años 2018 y 2020, que coinciden con ser los años con mayor volatilidad de la ventana de estudio del presente trabajo.

Sin embargo, también existen ventajas al tener portafolios más pequeños como lo son: menores costos de transacción, menores comisiones por rebalanceo, así como facilidad de manejo y operación. También, al tener un universo más pequeño de empresas, se puede hacer un mejor análisis para encontrar oportunidades de inversión con un equipo más pequeño de analistas.

Otro factor importante es que el S&P realiza rebalanceos trimestrales, cambiando pesos de las emisoras, o incluso a veces quitando alguna empresa e incluyendo a otra distinta. El índice de mercado, SPY Index, hace los mismos rebalanceos trimestrales.

En el presente trabajo no se realizaron rebalanceos de portafolios, lo cual provocaba que, conforme avanzara el tiempo, la correlación entre los portafolios generados y el índice de mercado se fuera perdiendo, y las métricas de comprobación resultaran mejor en el índice de mercado que en los portafolios generados.

El rebalanceo se vuelve aún más crítico en momentos de elevada volatilidad, como el caso del año 2020, pues una diferencia pequeña en el peso de un activo puede representar grandes cambios en el rendimiento total del portafolio. Aun así, a pesar de no realizar estos rebalanceos, la volatilidad de los portafolios generados se mantuvo similar a la del índice durante esos años, por lo que esto también confirma la hipótesis.

Debido a que el portafolio de regresión lineal resultó, en gran medida, mejor para algunas métricas (Beta, tracking error y R^2 ajustada y volatilidad), y el de *Random Forest* tuvo mejores resultados en otras (Treynor, Sharpe, VaR y CVaR), no es posible determinar con certeza si alguno de los dos modelos resultaría mejor que el otro para la construcción de portafolios que repliquen índices.

Lo que se podría hacer sería un análisis de cada portafolio, tomando en cuenta su diversificación, nivel de riesgo, composición sectorial, etc., a lo largo de un periodo económico completo de diez años, e incluso en ese punto, quizás ningún modelo es mejor que otro, sino que dependerá de las preferencias de cada inversionista.

A continuación se analizan los resultados relacionados con la segunda hipótesis del trabajo.

11.1.2 Rendimientos positivos utilizando arbitraje estadístico

Resultados

Algoritmo de trading RL vs Benchmarks		Vol	Treynor's Ratio	Sharpe Ratio
2019	Algorithmo de Trading	**0.05%**	**4.07**	**2.33**
	ACWI	0.75%	0.24	1.86
	S&P 500 (SPY ETF)	0.79%	0.27	2.18
	NASDAQ (QQQ ETF)	1.02%	0.29	2.25
	Dow Jones (DIA ETF)	0.79%	0.22	1.67
	Oro (GLD ETF)	0.73%	-0.65	1.41
	Petróleo (Brent Comdty)	1.50%	-3.60	-0.98
	Bitcoin (BTCUSD Curncy)	4.47%	-10.30	2.21
2020	Algorithmo de Trading	**0.00%**	**0.00**	**0.00**
	ACWI	2.04%	0.15	0.44
	S&P 500 (SPY ETF)	2.12%	0.16	0.47
	NASDAQ (QQQ ETF)	2.26%	0.49	1.32
	Dow Jones (DIA ETF)	2.32%	0.07	0.19
	Oro (GLD ETF)	1.23%	2.69	1.26
	Petróleo (Brent Comdty)	1.23%	-0.42	0.13
	Bitcoin (BTCUSD Curncy)	4.23%	-6.80	3.74

Algoritmo de trading RF vs Benchmarks		Vol	Treynor's Ratio	Sharpe Ratio
2019	Algorithmo de Trading	**0.10%**	**2.46**	**1.92**
	ACWI	0.75%	0.24	1.86
	S&P 500 (SPY ETF)	0.79%	0.27	2.18
	NASDAQ (QQQ ETF)	1.02%	0.29	2.25
	Dow Jones (DIA ETF)	0.79%	0.22	1.67
	Oro (GLD ETF)	0.73%	-0.65	1.41
	Petróleo (Brent Comdty)	1.50%	-3.60	-0.98
	Bitcoin (BTCUSD Curncy)	4.47%	-10.30	2.21
2020	Algorithmo de Trading	**0.28%**	**-0.96**	**0.21**
	ACWI	2.04%	0.15	0.44
	S&P 500 (SPY ETF)	2.12%	0.16	0.47
	NASDAQ (QQQ ETF)	2.26%	0.49	1.32
	Dow Jones (DIA ETF)	2.32%	0.07	0.19
	Oro (GLD ETF)	1.23%	2.69	1.26
	Petróleo (Brent Comdty)	1.23%	-0.42	0.13
	Bitcoin (BTCUSD Curncy)	4.23%	-6.80	3.74

Figura 11.2. *Resultados de la estrategia de inversión de arbitraje estadístico, con los diferentes portafolios generados, y comparada con otras inversiones alternativas.*

Respecto a la segunda hipótesis, se concluye que, efectivamente es posible generar α positivo utilizando estrategias de arbitraje estadístico. Esto se confirma en ambos portafolios generados con ambos modelos de machine learning para el año 2019 y de nuevo para el portafolio generado con random forest para 2020, pues tuvieron rendimientos positivos.

Algo importante a destacar es que, a pesar de que se lograron rendimientos positivos, esto no significa que sean la mejor inversión disponible, o si quiera que sean portafolios ideales. Para realmente aceptar invertir en un portafolio se debería hacer un análisis mucho más riguroso en el cual se incluya la composición por sectores, se analicen los pesos de cada emisora, análisis de riesgo, concentración por emisora y sector, entre otras cosas que quedan fuera del enfoque del presente trabajo, como el análisis de factores y la diversificación del portafolio.

Hay que recordar que el número de emisoras que utilizan los portafolios generados es muchísimo menor a las que utiliza el ETF de mercado. Esto podría provocar un mayor riesgo por falta de diversificación, provocando que el portafolio esté muy cargado a un sector y que, si ese particular sector se viera afectado de manera positiva o negativa más que el resto de sectores, el portafolio podría desviarse del índice al que intenta replicar resultando en pérdidas para la estrategia.

Con relación al algoritmo de trading, podemos darnos cuenta de que genera buenos rendimientos siempre que se cumpla la premisa de reversión a la media. Si la razón entre un instrumento y otro comienza a expandirse fuera de los rangos promedio, se tienen pérdidas. Además, es un algoritmo que tiene ganancias acotadas por la distancia entre la media de la razón y las desviaciones estándar, y pérdidas no acotadas —potencialmente infinitas— si se pierde la correlación, provocando que la razón entre portafolios se pierda en el horizonte. Por ello, es recomendable el uso de una orden de *stop-loss* que permita tomar las pérdidas antes de que estas se vuelvan insostenibles.

Respecto a la comparación con otras inversiones, parece ser muy buena inversión **para los periodos de 2019 y 2020**, tomando en cuenta la baja volatilidad —más parecida a invertir en UST que a invertir en acciones—, y los rendimientos generados.

11.2 Investigaciones posteriores

Existen diversos elementos que no se tomaron en cuenta para el presente trabajo y que pueden servir para ampliar y enriquecer los resultados. Lo siguiente son algunas recomendaciones de investigaciones posteriores que pueden ser de interés y gran valor para los desarrollos en materia de trading con algoritmos basados en arbitraje estadístico.

Área: Programación/Machine Learning

Hay una gran cantidad de modelos que no se cubren en este trabajo. Entre ellos están los métodos por splines, regresión lasso, ridge, elastic net, métodos de suavizamiento de kernel, deep learning, natural language processing, máquinas de soporte vectorial, ensemble learning, entre muchos otros. De igual manera, se puede probar con un Random Forest pero esta vez de clasificación, o en teoría con cualquier otro modelo de clasificación, buscando encontrar categorías para las empresas de acuerdo a sus rendimientos.

Probar con cualquiera de estos métodos resultaría interesante, aunque, como se explicó en los primeros capítulos, el implementar el grado de precisión de los modelos, puede disminuir su grado de interpretabilidad. A grandes rasgos, estos modelos sugeridos, y que no se utilizaron en el presente trabajo, implican mayor flexibilidad en los supuestos, o un incremento en la dependencia de los paquetes estadísticos para programación (como el caso de redes neuronales).

Finalmente, también se pueden buscar otro tipo de inputs, diferentes a los rendimientos, que tengan que ver con las empresas. Puede ser desde volumen operado, tamaño de la empresa, sector, entre muchas otras opciones que podrían hacer interesante el análisis y la creación de un portafolio distinto, no solo basado en rendimientos.

Área: Estadística

Entre otras cosas, se pueden incluir pruebas para medir la correlación lineal entre los predictores y así encontrar causas a la correlación, y no únicamente el valor de la correlación. Esto ayudaría a buscar un factor único que fuera la variable más importante a estimar.

Por ejemplo, si varias empresas de tecnología están correlacionadas, se puede pensar en el motivo detrás de eso, que podrían ser diversos factores macroeconómicos o de tecnología. Con eso, cambiaríamos los predictores y en lugar de utilizar las empresas, utilizaríamos la causante principal del cambio en el precio de dichas empresas, y en lugar de eliminar alguna empresa aleatoria con el análisis de colinealidad, nos quedamos con las causas principales de la misma.

También es posible utilizar el factor de inflación de la varianza (VIF por sus siglas en inglés) para detectar correlaciones. [3] De esta forma, en lugar de utilizar el código para probar correlación por parejas y eliminar las que estén por debajo de un valor arbitrario, se puede utilizar para comparar una única variable independiente, con un grupo de variables, y así, conocer la contribución de los predictores al modelo.

[3] El VIF cuantifica la intensidad de la multicolinealidad en un análisis de regresión normal de mínimos cuadrados. Proporciona un índice que mide hasta qué punto la varianza de un coeficiente de regresión estimado se incrementa a causa de la colinealidad.

Área: Finanzas, economía y algoritmos

Lo primero sería hacer un análisis de la composición del portafolio. Esto permitiría conocer realmente la viabilidad de que el portafolio exista en el mercado, analizar su diversificación y obtener algunas medidas de riesgo, entre otros valores.

También, se puede ampliar el estudio a otros mercados globales, o incluso probar el algoritmo con acciones individuales que puedan ser explicadas por otros factores financieros, por otras empresas, o por factores macroeconómicos.

Es muy importante destacar que, si bien las hipótesis se cumplen en el presente trabajo, este únicamente aborda un ejemplo en tres años consecutivos: 2018, 2019 y 2020. Si realmente se quiere llegar a la conclusión de que se pueden construir portafolios de inversión que repliquen al índice con un subconjunto de empresas, y que se puede crear una estrategia de arbitraje estadístico a partir de este portafolio, es fundamental que se realice el estudio en por lo menos diez años de estudio, para poder incluir todo un ciclo macroeconómico.

Algo que no se tomó en cuenta en el presente trabajo, son costos de transacción, comisiones o brokerage fees, volúmenes operados, liquidez, o el bid/ask spread, entre otros costos que podrían afectar el desempeño de la estrategia.

Otro costo importante es el costo de ejecución, pues se asume que se puede ejecutar al precio de cierre visto en pantalla, lo cual puede no necesariamente ser cierto, existiendo una diferencia entre lo que se ve en pantalla y el precio al cual se ejecuta (a menos que se estén utilizando órdenes límite). Para esto se puede ocupar un análisis más profundo de micorestructura de mercado.

11.3 Conclusión y comentarios finales

En conclusión, se cumplieron ambas hipótesis: fue posible construir portafolios que repliquen un índice, utilizando modelos de machine learning y un subconjunto de acciones, y también se logró crear un algoritmo de trading basado en una estrategia de reversión a la media, el cual resultó en rendimientos positivos, para los años de 2018 a 2020 en particular. Si se quiere llegar a una conclusión más general, es necesario estudiar por lo menos diez años consecutivos, y pasar por diferentes periodos económicos, así como por crisis económicas distintas tales como la crisis financiera de 2008, o la de empresas de internet en el 2000.

Es un trabajo que, en opinión de quien lo escribe, abre la puerta a muchas investigaciones relacionadas pues se lograron resultados interesantes que, de expandir su investigación correctamente, pueden llevar a la construcción de modelos aplicables a la realidad y que generen rendimientos positivos consistentes, siempre que se haga un adecuado *backtesting* en diferentes tipos de escenarios.

Bibliografía

[1] Barry Johnson. *Algorithmic Trading & DMA - An introduction to direct access trading strategies*. London: 4 Myeloma Press, 2010.

[2] Gareth James et al. *An Introduction to Statistical Learning with applications in R*. New York: Springer, 2013.

[3] Trevor Hastie, Robert Tibhirani, and Jerome Friedman. *The Elements of Statistical Learning*. California: Springer, 2008.

[4] Marcos López de Prado. *Advances in Financial Machine Learning*. New Jersey: Wiley, 2018.

[5] Ernest P. Chan. *Algorithmic Trading: Winning strategies and their rationale*. New Jersey: Wiley, 2013.

[6] Tony Guida. *Big Data and Machine Learning in Quantitative Investment*. United States of America: Wiley, 2019.

[7] Yves Hilpisch. *Python for Finance: Analyze Big Financial Data*. California: O'Riley, 2015.

[8] Zura Kakushadze. "Quant Bust 2020". In: *World Economics* 21(2) (2020), pp. 183–217. DOI: https://dx.doi.org/10.2139/ssrn.3570280.

[9] Nagesh Singh Chauhan. *A beginner's guide to Linear Regression in Python with Scikit-Learn*. 2019. URL: https://towardsdatascience.com/a-beginners-guide-to-linear-regression-in-python-with-scikit-learn-83a8f7ae2b4f.

[10] Will Koehrsen. *An Implementation and Explanation of the Random Forest in Python*. 2018. URL: https://towardsdatascience.com//an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76.

[11] James Chen. *Equity Market*. 2020. URL: https://www.investopedia.com/terms/e/equitymarket.asp.

[12] James Chen. *Index*. 2020. URL: https://www.investopedia.com/terms/i/index.asp.

[13] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[14] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation". In: *Computing in Science & Engineering* 13.2 (2011), p. 22.

[15] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[16] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 56–61. DOI: 10.25080/Majora-92bf1922-00a.

[17] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[18] Skipper Seabold and Josef Perktold. "statsmodels: Econometric and statistical modeling with python". In: *9th Python in Science Conference*. 2010.

[19] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[20] Plotly Technologies Inc. *Collaborative data science*. 2015. URL: https://plot.ly.