DOBLE GRADO ADE – INGENIERÍA INFORMÁTICA

Facultad de Ciencias Económicas y Empresariales

Simulación de Monte Carlo para la evaluación del riesgo de estrategias de trading en R

Pablo Delgado Gómez

Trabajo Fin de Grado ADE – Ingeniería Informática

TUTORA: *María Pilar García Pineda*

Madrid, junio de 2024

RESUMEN

En la actualidad, la inversión en los mercados financieros se ha convertido en una opción cada vez más atractiva para diversificar el patrimonio y generar ingresos adicionales. Sin embargo, no podemos olvidar que esta actividad conlleva un riesgo intrínseco debido a la incertidumbre de estos mercados.

Para minimizar este riesgo y aumentar las posibilidades de éxito, la elección de una estrategia de inversión adecuada es fundamental. Existen diferentes tipos de estrategias, cada una con sus propias características y ventajas. La clave está en encontrar la estrategia que mejor se adapte a nuestro perfil de riesgo y objetivos de inversión.

El riesgo se puede definir como la probabilidad de que una inversión no cumpla con las expectativas del inversor, y por ello, es muy importante medirlo antes de implementar alguna estrategia de inversión.

En este TFG se desarrollará un sistema en R para la evaluación del riesgo de estrategias de *trading* basadas en indicadores técnicos. Para ello, se usa la técnica de Simulación de Monte Carlo (SMC), un método computacional que permite generar miles de escenarios posibles del mercado a partir de un modelo probabilístico.

ÍNDICE

Capítulo 1 – Introducción

1.1 Motivación

En la actualidad, la inversión en los mercados financieros es una actividad cada vez más popular. Sin embargo, invertir conlleva riesgo, y es importante que los inversores sean conscientes de los riesgos que están asumiendo.

Este TFG tiene como objetivo evaluar el riesgo de diferentes estrategias de *trading* basadas en indicadores técnicos en el contexto del mercado estadounidense (S&P500). De esta forma se evalúa la rentabilidad de la estrategia asociada al riesgo y hace más sencillo optar por una u otra estrategia según su perfil de aversión al riesgo.

Me apasiona el tema porque mi interés en el mundo de las inversiones está en constante crecimiento. Quiero profundizar para tomar mejores decisiones en el futuro, ya que considero que es importante que los inversores comprendan los riesgos que implica invertir.

Este TFG me permitirá aplicar los conocimientos que he adquirido durante mi carrera y me ayudará a ampliar mi formación al adentrarme un poco más en el análisis cuantitativo financiero.

1.2 Objetivos

- Implementar algoritmos en R para la simulación de escenarios de mercado.

- Implementar estrategias de *trading* basadas en la compraventa de indicadores técnicos

- Cálculo y comparación del rendimiento de las estrategias de *trading* bajo múltiples escenarios.

- Evaluar el riesgo de las diferentes estrategias de *trading* implementadas.

- Desarrollar la visualización de los indicadores técnicos, así como los puntos de compra y venta de las estrategias.

1.3 Metodología

Entorno de desarrollo:

Se utilizará el lenguaje de programación R para desarrollar la herramienta. R (R Core Team, 2023) es un lenguaje gratuito y de código abierto que cuenta con una amplia comunidad de usuarios y desarrolladores.

Recopilación de datos:

Se recopilarán datos históricos de los precios del S&P500 desde el año 2000 hasta la fecha actual. Los datos se obtendrán de una fuente pública llamada Yahoo Finance.

Implementación de la simulación y estrategias:

Se implementará la simulación de Monte Carlo para generar miles de escenarios posibles del mercado. En cada escenario se calculará el rendimiento y riesgo de las estrategias de *trading*.

Análisis de resultados:

Se analizarán los resultados de la simulación para evaluar el rendimiento y riesgo de las estrategias de *trading*. Se calcularán medidas de rendimiento como la rentabilidad y el porcentaje de éxito, y de riesgo como la volatilidad y el ratio de Sharpe.

Capítulo 2 – Marco Teórico

2.1 Estrategias de *Trading*

El *trading* es una actividad que consiste en la compra y venta de activos en un corto o medio plazo con el objetivo de obtener beneficios. A las personas que operan en el mercado de esta forma se les llama *traders*. Las estrategias de *trading* son conjuntos de reglas y técnicas que los *traders* utilizan para tomar decisiones como comprar y vender. Dentro de las estrategias de *trading* existen muchas con una enorme complejidad técnica, que incluso se programan para que las acciones se ejecuten de forma automática por computadoras. Sin embargo, en este caso se van a analizar estrategias muy simplistas y básicas basadas exclusivamente en indicadores técnicos.

2.1.1 *Relative Strength Index* (RSI)

El índice de fuerza relativo (RSI) es un oscilador de impulso que mide la velocidad y el cambio del movimiento de los precios (Liu, 2023).

Su cálculo (R Documentation) se realiza de la siguiente forma:

$$RSI = 100 - \frac{100}{1 + \frac{Avg\ gain}{Avg\ loss}}$$

Avg gain: Es la media de los n periodos que sube el precio
Avg loss: Es la media de los n periodos que baja el precio

El RSI oscila entre los valores 0 y 100, y coger los últimos 14 días para el cálculo es algo muy común. Los valores clave típicos de referencia que utilizan los traders son, cuando el valor está por debajo de 30 (hay una sobreventa y se puede esperar un repunte, por tanto, momento de comprar), y cuando se encuentra por encima de 70 (existe una sobrecompra y es probable que haya una corrección a la baja, señal de venta) (Liu, 2023).

2.1.2 *Moving Average Convergence Divergence* (MACD)

La media móvil de convergencia-divergencia es un indicador de impulso que muestra la relación entre dos medias móviles exponenciales (EMA) del precio de un activo.

Consiste en dos líneas. Por un lado, la línea de MACD refleja la diferencia entre las dos medias móviles exponenciales, siendo una a corto plazo y otra a largo. Por otro lado, la línea de señal, que es una EMA de la línea de MACD.

Su cálculo se realiza de la siguiente forma:

$$MACD = EMA\ (short) - EMA\ (long)$$

$$Signal = EMA\ (MACD)$$

Los periodos más utilizados para las EMA suelen ser, 12 días para el periodo de corto plazo, 26 días para el periodo de largo plazo y 9 días para el periodo de la señal.

Cuando la línea de MACD cruza por encima de la línea de señal puede indicar una señal alcista (señal de compra). Si pasa lo contrario, MACD cruza por debajo de la línea de señal, puede indicar una señal bajista (señal de venta) (Liu, 2023).

2.1.3 Bandas de Bollinger

Las bandas de Bollinger son un indicador de volatilidad que mide la desviación estándar de los cambios de precios.

El indicador consta de tres líneas: la línea media (una media móvil simple), y dos líneas exteriores (bandas superior e inferior) trazadas desde la media móvil en función de un número específico de desviaciones estándar.

Su cálculo se realiza de la siguiente forma:

$$Middle\ Band = SMA$$

$$Upper\ Band = SMA + Sd * k$$

$$Lower\ Band = SMA - Sd * k$$

SMA: Media móvil simple
Sd: Desviación estándar
k: Número de desviaciones estándar

Los precios suelen estar entre las bandas superior e inferior, y cuando la diferencia entre estas se ensancha, indica que ha crecido la volatilidad. La SMA que se suele tomar es de 20 días y el valor asociado de la k a este periodo es 2, es decir, para las bandas superior e inferior se le suma o resta 2 veces la desviación estándar del precio (Liu, 2023).

Lo más común es utilizar las bandas de Bollinger como complemento adicional para identificar tendencias de mercado. Sin embargo, como los precios normalmente se mueven entre las bandas, se quiere investigar cómo funciona que se compre y venda cuando se toquen sus límites. Este método se ejecuta de forma que, en cuanto se toca la banda inferior se compra, y cuando se toca la banda superior se vende.

2.2 Simulación de Monte Carlo

Los métodos de Monte Carlo se fundamentan en la relación entre probabilidad y tamaño. Los principios de la teoría de la medida nos permiten comprender la probabilidad como algo vinculado al tamaño o magnitud de un conjunto de resultados posibles. En este contexto, un evento se asocia a su tamaño o medida en relación con el total de resultados posibles en un universo dado. Monte Carlo emplea esta relación de forma inversa, al calcular el tamaño de un conjunto e interpretarlo como una probabilidad. En términos más simples, esto implica tomar muestras aleatorias de un conjunto de resultados posibles y considerar la proporción de muestras que caen dentro de un conjunto determinado como una estimación de su tamaño. La ley de los grandes números garantiza que esta estimación se aproxime al valor correcto a medida que aumentamos el número de muestras (Glasserman, 2004). En este escenario particular, Monte Carlo se utiliza para simular el precio de mercado del S&P500 en un horizonte temporal de un año. El procedimiento implica generar una gran cantidad de escenarios que se basan tanto en los precios históricos que reflejan el comportamiento pasado, como en números aleatorios que introducen incertidumbre al modelo.

2.2.1 Teoría del Movimiento Browniano Geométrico

“El movimiento browniano geométrico es el modelo más fundamental del valor de un activo financiero. En su tesis pionera de 1900, Louis Bachelier desarrolló un modelo de precios de acciones que en retrospectiva describimos como movimiento browniano ordinario, aunque las matemáticas del movimiento browniano aún no se habían desarrollado. El uso del movimiento browniano geométrico como modelo en finanzas se debe principalmente al trabajo de Paul Samuelson en la década de 1960. Mientras que el movimiento browniano ordinario puede tomar valores negativos (una característica indeseable en un modelo de precio de una acción o cualquier otro activo de responsabilidad limitada), el movimiento browniano geométrico siempre es positivo porque la función exponencial sólo toma valores positivos” (Glasserman, 2004).

2.2.2 Fundamentos matemáticos

Cálculo de los Retornos

Primero se calculan los incrementos a partir de los valores del precio de la acción al cierre (S_i es el precio de cierre del día i).

$$\Delta S_i = S_{i+1} - S_i$$

Después, al dividir esos incrementos entre S_i, se obtienen los retornos de la muestra. Si posteriormente se hace un bucle recorriendo todos los precios históricos se tienen los retornos totales(Shonkwiler, 2013).

$$\frac{\Delta S}{S}$$

Sin embargo, se puede facilitar el cálculo en el código de R en una única línea, haciendo el cálculo de los retornos con la función “diff”. Esto es posible debido a que la suma de los retornos diarios se toma normalmente como hacer las diferencias de los precios con logaritmo (Fan & Yao).

$$\Delta log(S) = \log(S_{i+1}) - \log(S_i) = \frac{\Delta S}{S} = ret$$

Por tanto, se opta por realizarlo de esta forma para la posterior optimización y limpieza del código.

Cálculo de los Estimadores μ y σ

Para calcular el estimador de la media de los retornos (μ):

$$\mu \approx \frac{1}{n\Delta t}\sum_{0}^{n-1}\frac{\Delta S}{S} \approx \frac{mean(ret)}{\Delta t}$$

$$donde\ \Delta t = \frac{1}{n\ dias}$$

N días se refiere al número de días que la bolsa está operativa en el año, que suelen ser valores cercanos a los 252 días.

La varianza σ^2 y la desviación estándar σ se calculan (Shonkwiler, 2013) de esta forma:

$$\sigma 2 \approx \frac{1}{(n-1)\Delta t}\sum_{0}^{n-1}\left(\frac{\Delta S}{S} - \mu\Delta t\right)^2$$

$$\sigma \approx \sqrt{\frac{var(ret)}{\Delta t}}$$

Simulación con Movimiento Browniano Geométrico (GBM)

Partiendo de un precio inicial conocido, que es el último precio de cierre de mercado, se generan unos precios para cada día de mercado abierto dentro de una simulación. Esto se logra aplicando la siguiente fórmula (Glasserman, 2004) para calcular el precio de los días consecutivos:

$$S_{t+1} = S_t * \ e^{\left[\left(\mu - \frac{\sigma^2}{2}\right)\Delta t + \sigma\sqrt{\Delta t} Z_t\right]}$$

donde S_t es el precio del día t, y Z_t es un número aleatorio generado para cada día de la simulación, necesario para introducir incertidumbre al modelo. Este número aleatorio sigue una distribución normal estándar N (0, 1).

Con esto, se genera una serie temporal de precios simulados, que luego se utiliza para evaluar las estrategias de *trading*.

2.2.3 Aplicaciones en finanzas

Los métodos de Monte Carlo son herramientas fundamentales en las finanzas modernas debido a su capacidad para modelar y analizar sistemas complejos y aleatorios. Un ejemplo de ello es la aplicación de los métodos de Monte Carlo en la medición y gestión de riesgos.

El concepto de riesgo es multidimensional, muchos factores influyen en él. Algunos de los riesgos más importantes son los relativos a los mercados financieros, como la volatilidad, los tipos de interés y los tipos de cambio. Otros, como el riesgo de crédito, han adquirido más importancia tras sucesos como la crisis de 2008 o los cambios en los mercados energéticos.

Entidades que lo utilizan

- Instituciones financieras como bancos, fondos de inversión, etc.
- Empresas para valorar proyectos y gestionar riesgos.
- Reguladores para evaluar la estabilidad del sistema financiero.

En resumen, los métodos de Monte Carlo son una herramienta versátil y poderosa para las finanzas modernas.

2.2.4 Ventajas e inconvenientes

Ventajas

Una de las principales ventajas de los métodos de Monte Carlo es su capacidad para tener en cuenta la incertidumbre y la variabilidad de las variables en un modelo financiero. A diferencia de los modelos deterministas que suponen valores fijos, los métodos de Monte Carlo permiten simular una amplia gama de posibles valores futuros basados en distribuciones de probabilidad.

Otra gran ventaja es la capacidad de generar miles o millones de simulaciones, que permite a los analistas explorar una gran variedad de escenarios posibles, identificando tanto los resultados más probables, como aquellos que lo son menos pero potencialmente pueden tener un alto impacto.

Inconvenientes

Una de las mayores desventajas de los métodos de Monte Carlo es su elevado coste computacional. La necesidad de ejecutar un gran número de simulaciones para obtener resultados precisos puede requerir un alto consumo de tiempo y recursos computacionales.

Otro inconveniente es que, para ciertos tipos de problemas, las simulaciones de Monte Carlo pueden necesitar un gran número de simulaciones para obtener una estimación en la que poder confiar.

2.3 Análisis de las estrategias

2.3.1 Análisis de la rentabilidad

En este apartado se van a analizar los resultados de la estrategia aplicada en la simulación de Monte Carlo. Para ello se han utilizado una serie de medidas:

Número total de operaciones

Es importante conocer el número total de operaciones, debido a que muchos brokers imponen una comisión por transacción realizada. Si el número de operaciones es elevado, la suma de estas comisiones es una variable más que habría que tener en cuenta para la rentabilidad final de la estrategia, pudiendo mermarla hasta el punto de que acabe dando pérdidas. Por tanto, hay que tener precaución con algo que aparentemente puede no parecer un problema.

Rentabilidad media por operación

Implica cuánto rendimiento se obtiene cada vez que finaliza una operación. Para calcular la rentabilidad media por operación lo primero que se hace es el cálculo de la rentabilidad de todas las operaciones de forma individual. Este cálculo se ejecuta haciendo la división entre el precio de venta y el de compra, y a ese resultado se le resta 1. Después de tener todas las rentabilidades de las operaciones se saca la media.

Profit factor

El profit factor (factor de ganancias) es la relación entre las pérdidas y las ganancias totales. Se podría expresar de forma que por 1 unidad de pérdidas se tiene el profit factor de ganancias. Se calcula haciendo la división del total de las ganancias entre el total de las pérdidas. Si da un resultado mayor que 1 significa que en términos de precios la estrategia ha sido rentable.

Rentabilidad media por simulación

Es una medida de rentabilidad que representa aproximadamente la rentabilidad por simulación de la estrategia. Es la que más va a marcar la diferencia para comparar las rentabilidades de las diferentes estrategias.

Rentabilidad de comprar y mantener

Hay veces que en los mercados se puede obtener rentabilidad, aunque la estrategia sea mala. Esto puede suceder, por ejemplo, si el activo ha aumentado de valor. Por ello, es muy útil comparar la rentabilidad del activo (desde el inicio de la SMC hasta el final) con la de la estrategia, para ver si la estrategia ha superado al mercado de forma natural. Se calcula como la división entre la media de precios del último día de todas las simulaciones entre el precio actual, y a ese resultado se le resta 1.

Winning percentage

Si el porcentaje de acierto de la estrategia es mayor que el 50% significa que acierta más que falla. Sin embargo, este porcentaje no implica que haya ganancias, ya que las pérdidas pueden ser mayores que las ganancias haciendo que la rentabilidad quede negativa. Se calcula como operaciones ganadoras entre total de operaciones.

2.3.2 Análisis del riesgo

En el *trading* y en general en el mundo de las inversiones, no sólo es importante la rentabilidad, sino que es muy importante tener en cuenta el riesgo que se asume. Lo más normal es que exista una correlación positiva entre rentabilidad y riesgo, pero esa correlación implica tanto rentabilidad positiva, como negativa. En este apartado se van a analizar algunas medidas para ver cómo se comportan las estrategias en cuanto al riesgo.

Volatilidad media de las rentabilidades

La rentabilidad total y la media de las rentabilidades de cada operación pueden ser positivas, pero si existe una elevada variación eso implica un mayor riesgo. La media no implica que no existan probabilidades en los mercados financieros, sino que hay que tener en cuenta esa posibilidad. Por ello, es una medida muy importante. Se calcula como la desviación típica del conjunto de todas las rentabilidades.

Sharpe ratio

El Sharpe ratio (Liu 2023) sirve para medir la rentabilidad asociada al riesgo. La diferencia fundamental con respecto al retorno sobre riesgo (volatilidad), es que a la rentabilidad se le resta la tasa libre de riesgo. Es importante porque si la rentabilidad va a ser la misma o menor no tiene ningún sentido invertir en un activo con mayor riesgo.

Por regla general, se calcula restando la rentabilidad del portfolio menos la tasa libre de riesgo, y se divide el resultado entre la volatilidad del portfolio. Sin embargo, en Monte Carlo como el precio no es único, sino que son muchas simulaciones, hay dos posibilidades de cálculo.

La primera de ellas es calculando la rentabilidad de cada simulación menos la tasa libre de riesgo, entre la volatilidad de la simulación, dando un Sharpe ratio por simulación. Posteriormente habría que realizar la media de todos los Sharpe ratios para obtener un único resultado final. De esta forma se puede ver la distribución de la rentabilidad asociada al riesgo por simulaciones.

La segunda sería calculando la media de las rentabilidades de todas las operaciones que se realizan en Monte Carlo, menos la tasa libre de riesgo, entre la volatilidad media de las rentabilidades de las operaciones.

Como se cree que ambas opciones son válidas y los resultados pueden diferir, se van a calcular ambas para analizar los resultados.

La tasa libre de riesgo adquiere el valor relativo a la rentabilidad (Tesoro Público de España) de las Letras del Tesoro de España a un año, que es 3,405%.

Capítulo 3 – Metodología

3.1 Entorno de desarrollo

3.1.1 Lenguaje R

Los lenguajes de programación más recurridos para este tipo de proyectos de análisis de datos son R y Python. En este caso, se puede destacar que R es superior a Python a nivel de estadística y visualización de datos. Se utiliza la versión 4.3.2 de R debido a que es reciente y compatible con los paquetes utilizados.

3.1.2 Paquetes R

Los paquetes adicionales utilizados son:

- **Quantmod** (Ryan, Ulrich, 2024): Framework de modelado cuantitativo financiero.
- **Tidyverse** (Wickham et al., 2019): Este paquete incluye a su vez a varios paquetes, el fundamental en este caso es el ggplot2 que sirve para crear visualizaciones elegantes de datos.
- **Plotly** (C.Sievert, 2020): Crea visualizaciones de datos interactivos basados en web, se utiliza si se quiere interactuar con el gráfico.

3.1.3 Entorno de trabajo

Se utiliza RStudio (Posit Team, 2024) (IDE especializado en R de código abierto) para el desarrollo del código y la visualización de resultados. Se ha decidido utilizar este entorno debido a su facilidad de uso, su interfaz intuitiva y su organización. Además, la sintaxis del código está resaltada y cuenta con una gran comunidad que ofrece información para la resolución de problemas que puedan aparecer.

3.2 Recopilación de datos

3.2.1 Fuente y periodo de datos

Se ha utilizado Yahoo Finance como fuente de datos. Esta elección fundamentalmente es porque contiene toda la información requerida relativa a los valores de bolsa del S&P500 (Yahoo Finance España), los datos están actualizados y son de alta fiabilidad. Estos datos se descargan e importan automáticamente en RStudio a través del paquete “quantmod”. El periodo de los datos abarca desde el 1 de enero del año 2000 hasta el último precio de cierre del S&P500 (ejecutado el día 02-06-2024).

En primer lugar, se descargan los datos de Yahoo con “getSymbols()”. Después con “Cl (GSPC)” asignamos a una variable (sp500_cierre) únicamente los datos del precio de cierre. Por último, se crea un *dataframe* (sp500_df) que contiene el campo fecha (date) y campo precio de cierre (close).

3.2.2 Preprocesamiento de datos

Una vez que se tienen los datos históricos que se van a utilizar para la simulación de Monte Carlo, se debe preparar el periodo que se va a simular. En este caso, el periodo abarca 1 año natural que son 365 días. Sin embargo, en bolsa el número de días difiere bastante de los que conforman un año, ya que esta se encuentra cerrada los fines de semana y determinados festivos (Intercontinental Exchange, 2022). Por ende, se tiene que hacer una limpieza de datos, quedando finalmente el año natural en 255 días en los que la bolsa está abierta.

Cabe mencionar que, los días de bolsa abiertos al año, pueden cambiar en función de ciertos parámetros, como pueden ser el propio año, las festividades, el país, etc. La fecha de ejecución del código es a día 2 de junio de 2024.

3.3 Implementación de la simulación de Monte Carlo

3.3.1 Configuración de la simulación

En esta sección, describimos cómo se configuró la simulación de Monte Carlo, detallando las variables y su propósito en el modelo:

- **Números aleatorios reproducibles** (set.seed(n)): Establece una semilla aleatoria para asegurar la reproducibilidad en la generación de números aleatorios. Esto significa que, si se ejecuta con el mismo número “n”, la generación de números aleatorios producirá los mismos resultados cada vez que se ejecute el código. De esta forma se pueden analizar los datos con certeza de cómo afectan los posibles cambios o distintas estrategias de *trading*.

- **Número de simulaciones** (num_simulations): Representa la cantidad de simulaciones generadas para cada estrategia. Se ha elegido 1.000 simulaciones para mejorar la precisión sin tener por ello un costo computacional excesivo.

- **Horizonte temporal** (t): Número de días en un año de mercado. Este parámetro define la longitud de cada simulación.

- **Precio actual** (p_actual): El último precio de cierre disponible, que sirve como punto de partida para todas las simulaciones.

- **Retornos diarios logarítmicos** (ret): Se calculan para obtener la media de los retornos (μ).

- **Delta de t** (dt): El incremento del tiempo, necesario para fórmulas posteriores.

- **Media de los retornos** (mu): Calculada a partir de los datos históricos, la media de los retornos diarios logarítmicos.

- **Desviación típica** (desv): Volatilidad de los rendimientos diarios logarítmicos, calculada a parir de los datos históricos

3.3.2 Parámetros del modelo

En esta sección, se explican los parámetros específicos del Movimiento Browniano Geométrico y cómo se utilizan para la simulación.

Se crea una **matriz de datos** (gbm) para almacenar los precios simulados de cada día y cada simulación. La matriz se compone de filas, el número de días de bolsa abierta en 1 año, y de columnas, el número estipulado de simulaciones. La primera columna de cada simulación se inicializa con el precio actual.

Los parámetros del Movimiento Browniano Geométrico se definen de la siguiente forma:

- **Bucle simulaciones** (for 1): El primer bucle for es para ir recorriendo las simulaciones, desde la 1 hasta las que marque la variable num_simulations.

- **Bucle días** (for 2): El segundo bucle for se encuentra dentro del primero y va a ir recorriendo los días de cada simulación. Va desde el 2 (recordar que la primera columna siempre es el último precio de cierre) hasta lo que marque la variable t. Dentro de este bucle, y por tanto en cada iteración, se van a ejecutar la generación de un número aleatorio y la aplicación de la fórmula para simular precios. Además, en esta misma iteración se van a aplicar las estrategias de *trading* para esos precios simulados (se explican en el punto 3.3.3).

- **Número aleatorio entre 0 y 1** (zt): Generado cada día para incluir incertidumbre en el modelo. Esto permite simular la variabilidad de los mercados financieros.

- **Asignación de precio al día** (gbm[day, sim]): Fórmula del Movimiento Browniano Geométrico que simula el precio de día basado en el precio del día anterior, la media de retornos, la volatilidad y el número aleatorio generado.

3.3.3 Implementación de las estrategias

Aunque las estrategias sean diferentes, se han seguido una serie de similitudes para cumplir los objetivos del trabajo. Todas las estrategias incluyen lo siguiente:

- **Matriz de operaciones** (operaciones): Es una matriz de las mismas dimensiones que la matriz de gbm (días x simulaciones), que se utiliza para escribir “Compra” o “Venta” cuando se efectúen alguna de estas dos acciones. Así, se logra tener todos los registros de las operaciones ejecutadas en relación con sus correspondientes días. Posteriormente con estos datos se pueden realizar representaciones visuales con esos puntos en los que se efectúa una operación.

- **Lista combinada** (lista_combinada): Es un vector, cuyo tamaño depende de la estrategia, que se inicializa con un número de los datos históricos del sp500. Este vector se inicializa al inicio de cada iteración del primer bucle for, antes de entrar en el segundo. En este segundo bucle for, la lista se va actualizando hasta que se necesite con el fin de calcular los indicadores técnicos en cada paso de la simulación

- **Variables de operaciones**: Existen las variables de ganancia, perdida, op_ganadora y op_perdedora. Representan respectivamente la acumulación de ganancias, la acumulación de pérdidas, el número de operaciones en las que hay ganancias y el número de operaciones con pérdidas. Todas se inicializan a 0 antes de los bucles.

- **Compra y venta**: La compra se efectúa en el día actual si le corresponde la señal. Otro aspecto importante a tener en cuenta es que, para la compra y venta, se ha establecido que cuando se compra no se puede volver a comprar sin haber vendido, y que no se puede vender sin antes haber comprado. Para ello, en la señal de compra se comprueba que la variable compra sea igual a 0 para no incurrir en una doble compra sin venta de por medio. En caso de señal de venta pasa lo contrario, se comprueba que compra es distinto de 0 para saber que ha habido una compra antes y existe la posibilidad de compra.

Todas las matrices tienen dimensiones (n días x n simulaciones).

La creación de la lista combinada es fundamental para el correcto desarrollo de las estrategias. Tenemos, por un lado, el *dataframe* con todos los precios históricos, y, por otro lado, la matriz de precios simulados que se va completando en la misma iteración en la que se aplican las estrategias. La lista entra en juego para coger tanto los precios históricos, como los precios simulados necesarios, ya que se necesita de ambos para el cálculo. Se ve mejor con un ejemplo:

Ej. Con el RSI de tamaño 14, se necesitan 15 días para su cálculo. Si la simulación se encuentra en la posición (3, 1) de gbm, se necesitan los últimos 13 días de precios históricos y 2 de los precios simulados (posiciones (2,1) y (3,1)).

Por tanto, esa combinación es cambiante, actualizando los valores antiguos por nuevos en cada día simulado hasta que se dejan de necesitar los precios históricos para el cálculo del indicador técnico.

Es importante destacar que la compra se puede realizar a partir del precio actual y no se puede comprar dos veces sin haber vendido. En cuanto a la venta igual, no se puede vender si previamente no se ha comprado.

En términos generales, es como si se fuera a todo o nada con la compra y venta. Así se simplifican las infinitas posibilidades que hay de operar en el mercado y se centra todo en la decisión basada en el indicador técnico.

3.3.3.1 Relative Strength Index (RSI)

En este apartado se va a explicar cómo se ha implementado la estrategia de *trading* basada en la compra o venta según señales del RSI. Se define lo siguiente:

- **Matriz de RSI** (rsi): Es una matriz que sirve para almacenar todos los valores del RSI de cada día simulado y poder hacer una representación visual del indicador.

- **Lista combinada**: Profundizando dentro de la estrategia basada en el RSI, esta lista debe tener un tamaño de 15, un día más que los n días utilizados para el cálculo del RSI, que se han decidido en 14 días.
- **Cálculo de RSI** (rsi_aux): Para el cálculo del RSI he usado el paquete "quantmod". Los parámetros de esta función RSI corresponden a los valores de la lista combinada o los de la matriz gbm, el periodo n de 14, y "type=SMA" especifica el tipo de media móvil que es simple.

- **Asignación a matriz**: El último valor devuelto por el RSI se inserta en la posición de la matriz.

- **Señales y acciones** (if, else if): Se compara si en el último valor del RSI hay señal de compra por ser menor a 30, y si lo es, se asigna a una variable (compra) el precio generado y se actualiza la matriz de operaciones. Si no lo es, se comprueba si hay señal de venta por ser mayor que 70, y en caso positivo se comprueba si es ganancia o pérdida, se pone a 0 la variable compra y se actualiza la matriz de operaciones.

3.3.3.2 Moving Average Convergence Divergence (MACD)

En este apartado se va a explicar la implementación de la estrategia de *trading* basada en la compraventa según señales del MACD. Se define lo siguiente:

- **Matrices de señales y macd** (signal, macd): En ambas matrices se guardan los valores de señal y de macd respectivamente para poder efectuar la representación visual.

- **Lista combinada**: Observando la estrategia basada en el MACD, esta lista debe tener un tamaño de 35, ya que el límite lo marca la media a largo plazo (26 días) más la señal (9 días).

- **Cálculo de MACD** (macd_aux): Se usa el paquete "quantmod". Los parámetros corresponden a los datos necesarios, el periodo a corto plazo (12), el de largo (26) y el de señal (9) y maType=EMA que indica que todas las medias móviles son de tipo exponencial. Devuelve dos valores, *macd* en la posición 1 y *signal* en la 2.

- **Asignación a matrices**: Se recogen los últimos valores que devuelve los cálculos del MACD y se insertan en las matrices correspondientes.

- **Señales y acciones** (if, else if): Se comparan los últimos valores de *macd* y de señal. Si el primero es mayor que el del segundo se compra, y si es menor se vende. Se ejecutan las mismas acciones dentro de compra y venta que las mencionadas en el RSI.

3.3.3.3 Bandas de Bollinger

En este apartado se va a explicar cómo se ha implementado la estrategia de *trading* basada en la compra o venta según señales de las bandas de Bollinger. Se define lo siguiente:

- **Matrices de banda superior, media e inferior** (upperBollinger, middleBollinger y lowerBollinger): Se guardan los valores de las tres bandas en sus matrices para poder hacer una representación visual.

- **Lista combinada**: Al efectuar el cálculo de las bandas se necesita un tamaño de lista de 20, ya que es el tamaño del cálculo de la media móvil.

- **Cálculo de las bandas**: Se utiliza el paquete “quantmod”. Los parámetros de datos necesarios son el periodo de datos que se utilizan para el cálculo, el tipo de media móvil (SMA) y la desviación típica escogida, que es 2 porque es lo que le corresponde al periodo de la media. Devuelve 4 valores de los cuales nos interesan los 3 primeros, banda inferior, media y superior en ese orden.

- **Asignación de últimos valores**: Se insertan en la posición de las matrices los últimos valores relativos a las bandas superior, media e inferior.

- **Señales y acciones**: (if, else if): Se comparan los últimos valores del precio y de la banda inferior y superior. Si el precio es menor que la banda inferior se compra y si el precio es mayor que la banda superior se vende. Se actualiza la matriz de operaciones y la variable compra en función de la acción.

3.4 Análisis de las estrategias

3.4.1 Análisis de la rentabilidad

Número total de operaciones

Para contar el número total de operaciones se hace un conteo cuando finaliza una operación, tras venta. Si hay ganancia se le suma uno a un contador llamado operaciones ganadoras, y si hay pérdidas se le suma a otro de operaciones perdedoras. El número total es la suma de ambos.

Rentabilidad media por operación

Cada vez que se vende se calcula la rentabilidad de esa operación y se mete en un vector que recopila todas las rentabilidades de Monte Carlo. Posteriormente se calcula la media de este vector que refleja la rentabilidad media por operación.

Profit factor

Se suman todas las ganancias en una variable (ganancias), esa se divide entre otra variable (perdidas), que tiene la suma de todas las pérdidas.

Rentabilidad media por simulación

Se utiliza la rentabilidad media por operación. Se multiplica por el número total de operaciones, dividido entre el número total de simulaciones.

Rentabilidad de comprar y mantener

Se cogen todos los precios correspondientes al último día de todas las simulaciones (gbm[, sim]) y se hace la media. Esa media entre el precio actual y restando -1, da la rentabilidad obtenida sin hacer nada más que comprar al principio de la simulación.

Winning percentage

El contador de todas las operaciones ganadoras dividido entre el contador de todas las perdedoras.

3.4.2 Análisis del riesgo

Volatilidad media de las rentabilidades

Para el cálculo se utiliza un vector que contiene todas las rentabilidades de todas las operaciones ejecutadas en las simulaciones, y se calcula sobre ese vector la desviación típica.

Sharpe ratio

Dos enfoques.

- En el primero se calcula obteniendo la media de las rentabilidades de la simulación, de forma que, si hay 3 operaciones, se acumulan las rentabilidades en una variable y se divide entre un contador del número de operaciones en esa simulación. Después se le resta la tasa libre de riesgo y se divide el resultado entre la volatilidad de esa simulación. Esta volatilidad se calcula haciendo la desviación típica de los retornos del precio de cada simulación, y se multiplica por la raíz del número de días.

- En el segundo enfoque, se hace la media del vector de todas las rentabilidades, quedando la rentabilidad media por operación, se le resta la tasa libre de riesgo, y se divide el resultado entre la volatilidad media de las rentabilidades. Esta se calcula haciendo la desviación típica del vector de todas las rentabilidades.

3.4.3 Visualización de los resultados

Para la visualización de los resultados se crean *dataframes* para las operaciones y cada uno de los indicadores correspondientes. Después se hace "cbind()" para vincular cada posición con la de las fechas. Además, a todas las columnas se les pone como nombre de cabecera el número de la simulación correspondiente. Luego se pasan a formato largo con los correspondientes datos y se hacen *left joins* para que se junten en un mismo formato largo por fecha y que se pueda representar lo que se requiere. Se apreciará mejor en los gráficos de cada estrategia. Por otro lado, se han dibujado dos histogramas para que se aprecie la variabilidad, uno de rentabilidades y otro de los Sharpe ratios.

Capítulo 4 – Resultados

4.1 Simulación de Monte Carlo

Se van a analizar los precios basados en la simulación de Monte Carlo mediante el Movimiento Browniano Geométrico. El Gráfico 1 muestra la evolución de los precios. Las áreas con tonos más rojos indican las zonas de precios con mayor probabilidad de ocurrencia, debido a una mayor concentración de simulaciones en esos rangos. En el Gráfico 2 se presenta la distribución de las simulaciones, observándose que la mayoría se encuentra entre los precios de 4500 y 6500. Al calcular la media, se obtiene un valor de 5556.172, lo que confirma la coherencia de los datos.

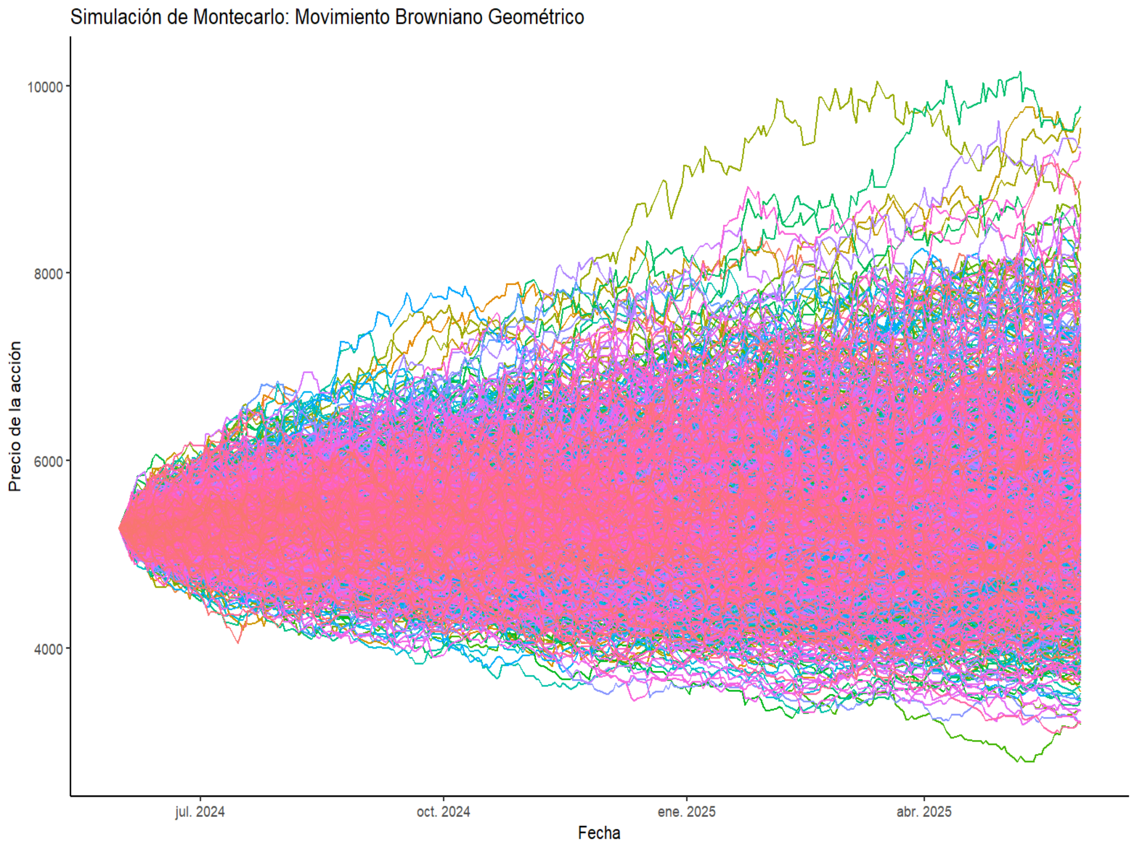

Gráfico 1. Simulaciones de Monte Carlo

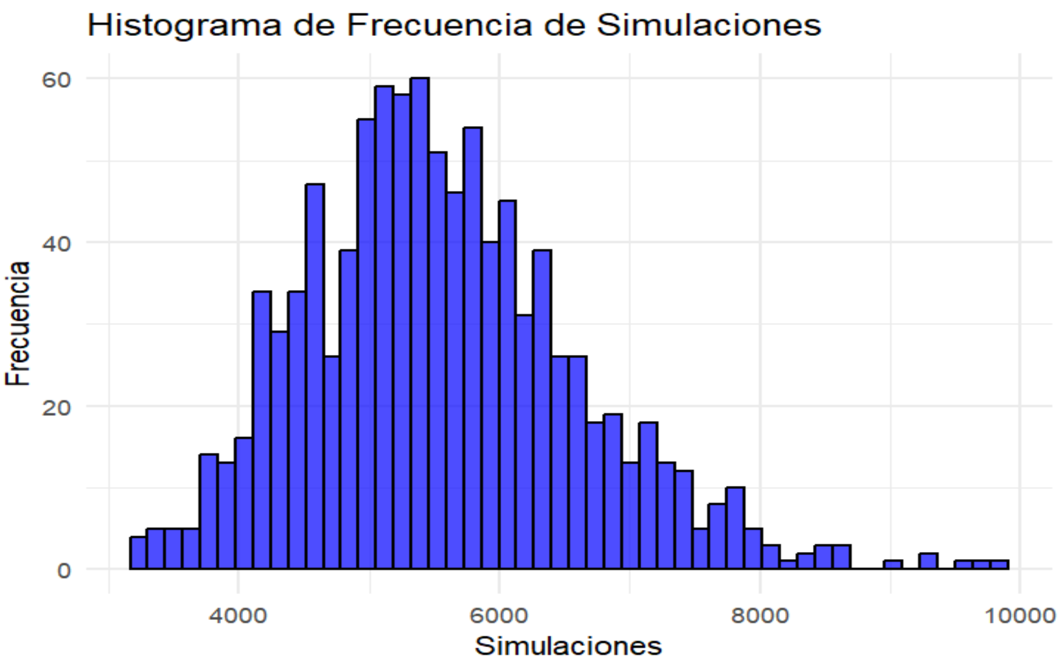

Gráfico 2. Histograma de simulaciones

A continuación, se van a analizar los resultados de las diferentes estrategias después de ejecutar mil simulaciones de Monte Carlo.

4.2 Análisis de la estrategia RSI

Para entender mejor los puntos de compraventa según el RSI, se pueden percibir en los gráficos 3 y 4, referentes a la primera simulación del modelo, siendo los triángulos verdes las compras y los rojos las ventas.

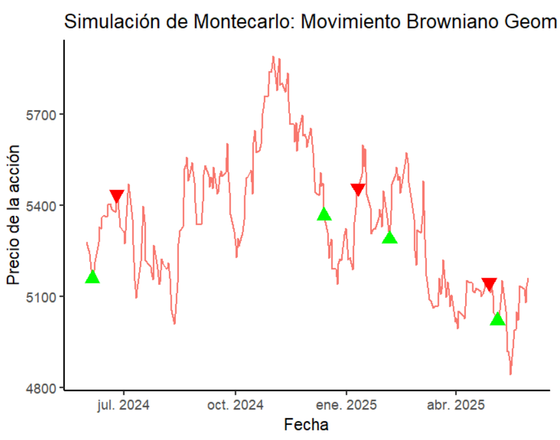

Gráfico 3. Señales RSI. Vista simulaciones

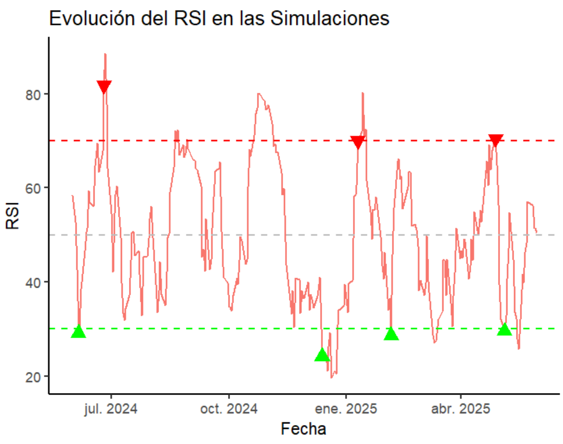

Gráfico 4. Señales RSI. Vista RSI

4.2.1 Análisis del rendimiento

Los resultados de las medidas de rentabilidad son los siguientes:

```
> print(paste("El total de operaciones es: ", total_operaciones))
[1] "El total de operaciones es:  2464"
> print(paste("La rentabilidad media por operación es: ", rentabilidad_media))
[1] "La rentabilidad media por operación es:  0.0170796353320467"
> print(paste("El profit factor es: ", profit_factor))
[1] "El profit factor es:  1.8864662187819"
> print(paste("La rentabilidad media por simulación es: ", rentabilidad_media_sim))
[1] "La rentabilidad media por simulación es:  0.0420842214581631"
> print(paste("La rentabilidad de comprar y esperar es: ", buy_and_wait))
[1] "La rentabilidad de comprar y esperar es:  0.0528018167276405"
> print(paste("El porcentaje de éxito de las operaciones es: ", winning_percentage))
[1] "El porcentaje de éxito de las operaciones es:  0.695616883116883"
```

Como se puede comprobar, el rendimiento de la estrategia de *trading* basándose en las señales del RSI parece bueno. Las probabilidades de éxito de las operaciones realizadas son elevadas (69,56%), y la rentabilidad media de cada una de las operaciones es del 1,7%, un valor muy adecuado. Por otro lado, no hay que olvidar los costes de operar, puesto que, en 1000 simulaciones, se han ejecutado 2464 operaciones que equivalen a una media de 2,464 operaciones en cada simulación y el doble de transacciones de compraventa. La rentabilidad media por simulación es de (4,20%), por lo que no supera la rentabilidad media del activo (5,28%), haciendo que sea más rentable comprar al inicio y no vender en todo el periodo que esta estrategia para hacer *trading*.

Analizando el gráfico 5, se puede ver cómo la distribución de las rentabilidades tiende a concentrarse más en la parte positiva, aunque se aprecia que hay una dispersión algo elevada.

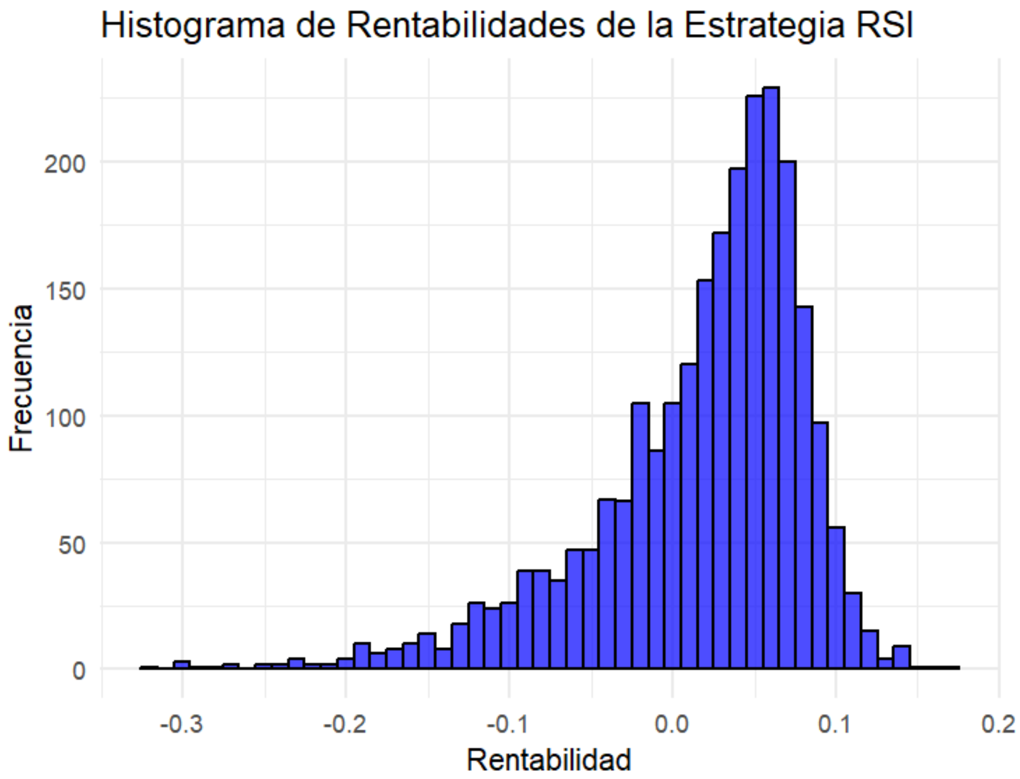

Gráfico 5. Histograma RSI. Rentabilidad

En general, se puede concluir que utilizando Monte Carlo la estrategia tiene buena predisposición en términos de rendimiento, pero seguramente la razón detrás de ello sea la rentabilidad del activo.

4.2.2 Análisis del riesgo

```
> print(paste("La volatilidad de todas las rentabilidades es: ", volatilidad_rentabilidades))
[1] "La volatilidad de todas las rentabilidades es:  0.0664138248903203"
> print(paste("El Sharpe ratio con la media de las rentabilidades es: ", sharpe_rentabilidades))
[1] "El Sharpe ratio con la media de las rentabilidades es:  -0.0439722403095461"
> print(paste("La media de las volatilidades es: ", volatilidad))
[1] "La media de las volatilidades es:  0.196468035334898"
> print(paste("La media de Sharpe ratio de las simulaciones  es: ", sharpe))
[1] "La media de Sharpe ratio de las simulaciones  es:  -0.127000214074131"
```

Con respecto al riesgo tenemos que la volatilidad de las rentabilidades elevada (6,64%), ya que como se pudo apreciar en el gráfico 5, existe cierta dispersión en cuanto a la rentabilidad de cada simulación.

Por otro lado, analizando los ratios de Sharpe, son negativos (-0,04 y -0,12), lo que implica que la rentabilidad asociada al riesgo no es positiva, y es mejor buscar otra estrategia o invertir en un activo libre de riesgo. Se puede apreciar cómo la distribución de los ratios de cada simulación en el gráfico 6 se concentra más en la parte negativa.

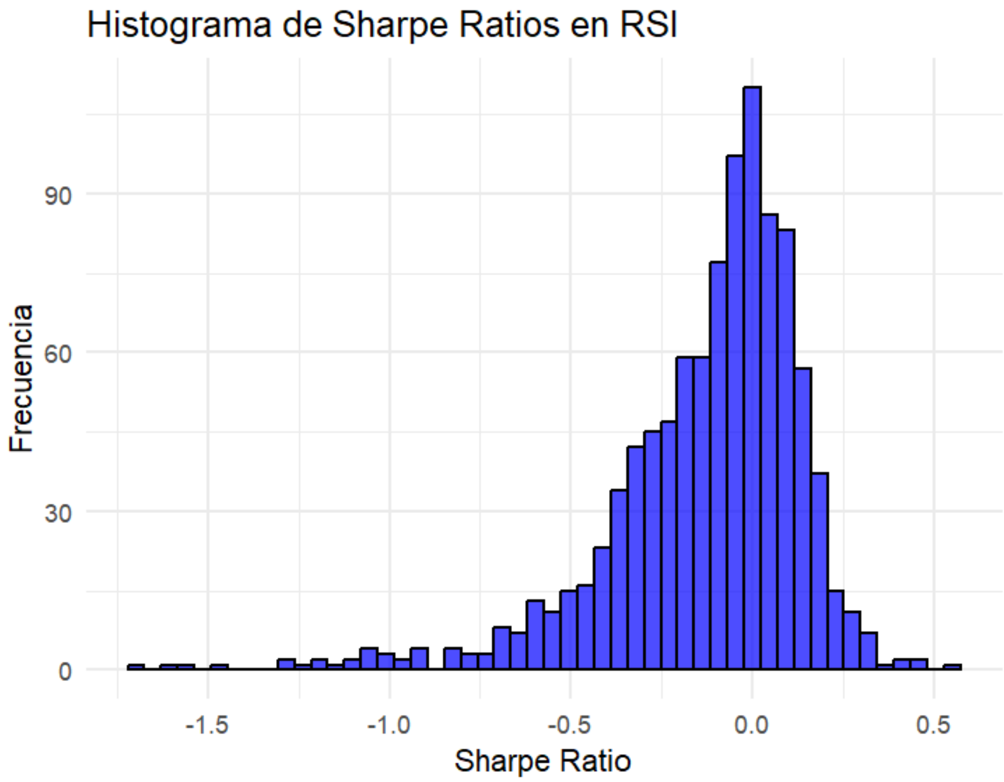

Gráfico 6. Histograma RSI. Sharpe ratios

4.3 Análisis de la estrategia MACD

Para entender mejor los puntos de compraventa según el MACD, se pueden percibir en los gráficos 7 y 8, referentes a la primera simulación del modelo, siendo en el segundo gráfico la línea azul el MACD y la roja la señal.

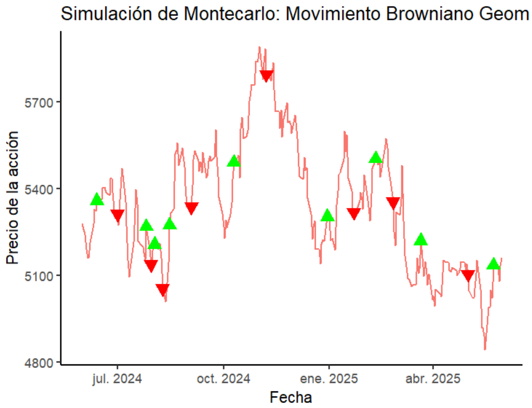

Gráfico 7. Señales MACD. Vista simulaciones

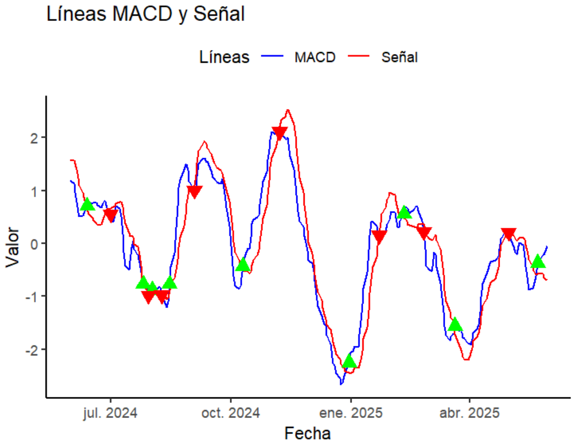

Gráfico 8. Señales MACD. Vista líneas

4.3.1 Análisis del rendimiento

En este apartado se van a presentar los resultados de la estrategia relacionados con la rentabilidad. Para ello se han utilizado una serie de medidas:

```
> print(paste("El total de operaciones es: ", total_operaciones))
[1] "El total de operaciones es:  9643"
> print(paste("La rentabilidad media por operación es: ", rentabilidad_media))
[1] "La rentabilidad media por operación es:  0.00179382771074572"
> print(paste("El profit factor es: ", profit_factor))
[1] "El profit factor es:  1.10599710938063"
> print(paste("La rentabilidad media por simulación es: ", rentabilidad_media_sim))
[1] "La rentabilidad media por simulación es:  0.017297880614721"
> print(paste("La rentabilidad de comprar y esperar es: ", buy_and_wait))
[1] "La rentabilidad de comprar y esperar es:  0.0528018167276405"
> print(paste("El porcentaje de éxito de las operaciones es: ", winning_percentage))
[1] "El porcentaje de éxito de las operaciones es:  0.389816447163746"
```

El rendimiento de la estrategia de *trading* basándose en las señales del MACD parece mala. Las probabilidades de éxito de las operaciones realizadas (38,98%) son menores que las de fracaso y la rentabilidad media de cada una de las operaciones es del 0,17%, un valor muy pobre. Además, no hay que olvidar los costes de operar, puesto que, en 1000 simulaciones, se han ejecutado 9643 operaciones que equivalen a una media de 9,643 operaciones en cada simulación y el doble de transacciones de compraventa. La rentabilidad media por simulación es de (1,72%), que se encuentra muy lejos de la rentabilidad media del activo (5,28%), haciendo que sea más rentable comprar al inicio y no vender en todo el periodo que esta estrategia para hacer *trading.*

Analizando el gráfico 9, se puede ver cómo la distribución de las rentabilidades tiende a concentrarse más en la parte negativa, aunque se aprecia que hay una dispersión hacia la parte positiva mayor que a la negativa.

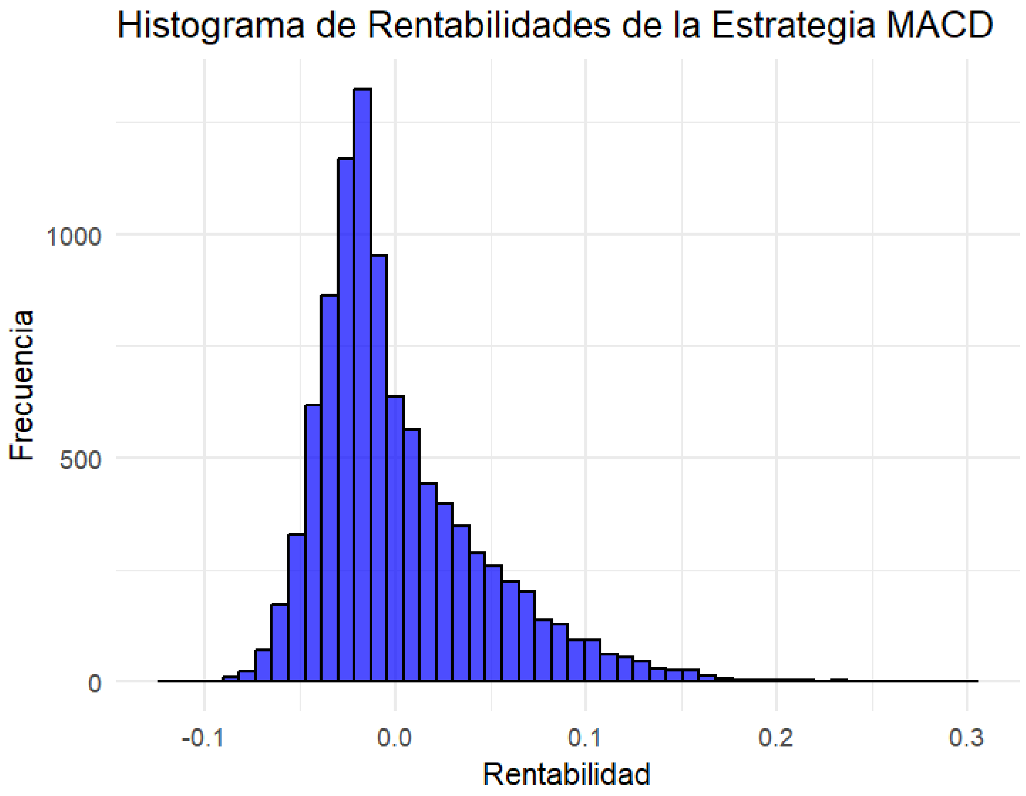

Gráfico 9. Histograma MACD. Rentabilidad.

En general, se puede concluir que utilizando Monte Carlo la estrategia tiene mala predisposición en términos de rendimiento y no merece la pena operar siguiendo las señales de MACD.

4.3.2 Análisis del riesgo

```
> print(paste("La volatilidad de todas las rentabilidades es: ", volatilidad_rentabilidades))
[1] "La volatilidad de todas las rentabilidades es:  0.0445272862045142"
> print(paste("El Sharpe ratio con la media de las rentabilidades es: ", sharpe_rentabilidades))
[1] "El Sharpe ratio con la media de las rentabilidades es:  -0.408876754932541"
> print(paste("La media de las volatilidades es: ", volatilidad))
[1] "La media de las volatilidades es:  0.196468035334898"
> print(paste("La media de Sharpe ratio de las simulaciones  es: ", sharpe))
[1] "La media de Sharpe ratio de las simulaciones  es:  -0.15756579533483"
```

Con respecto al riesgo tenemos que la volatilidad de las rentabilidades es moderada (4,45%), ya que como se pudo apreciar en el gráfico 9, existe algo de dispersión en cuanto a la rentabilidad de cada simulación.

Por otro lado, analizando los ratios de Sharpe, son negativos, lo que implica que la rentabilidad asociada al riesgo no es positiva, y es mejor buscar otra estrategia o invertir en un activo libre de riesgo. Se puede apreciar cómo la distribución de los ratios de cada simulación en el gráfico 10 es casi siempre negativa.

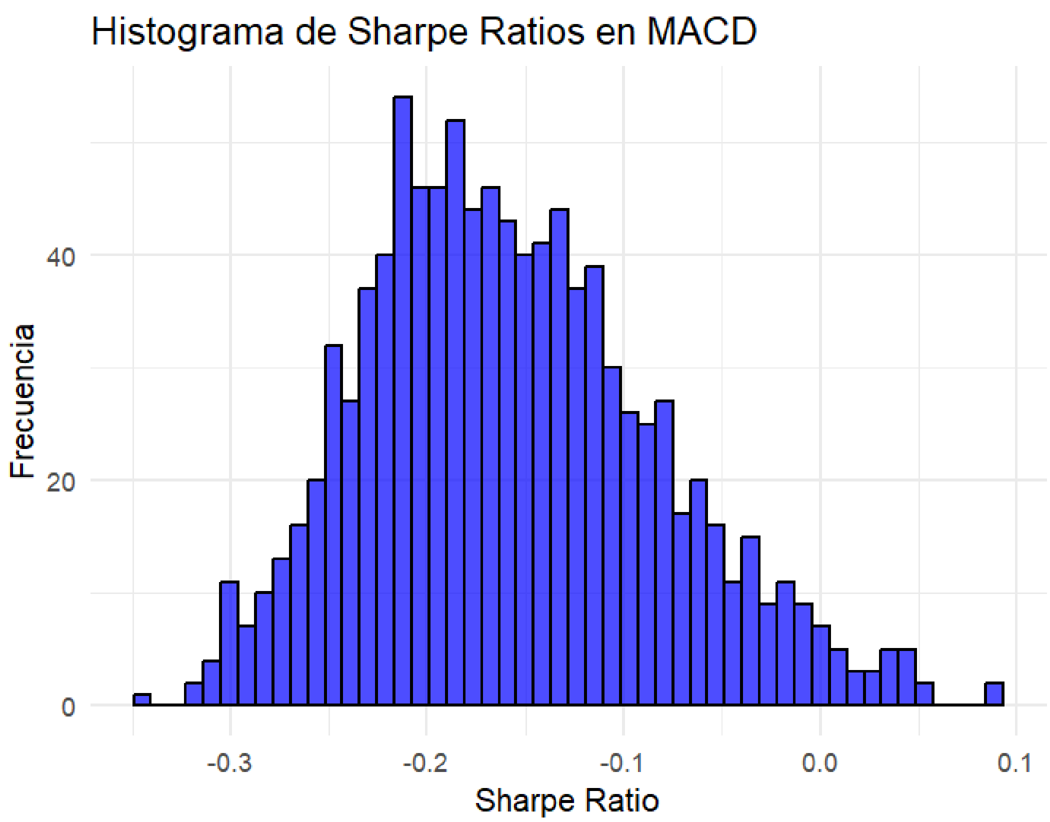

Gráfico 10. Histograma MACD. Sharpe ratios

4.4 Análisis de la estrategia bandas de Bollinger

Para entender mejor los puntos de compraventa según el MACD, se pueden percibir en el gráfico 11, referente a la primera simulación del modelo, siendo la línea negra el precio, verde la banda inferior, la gris la banda media y la roja la banda superior.

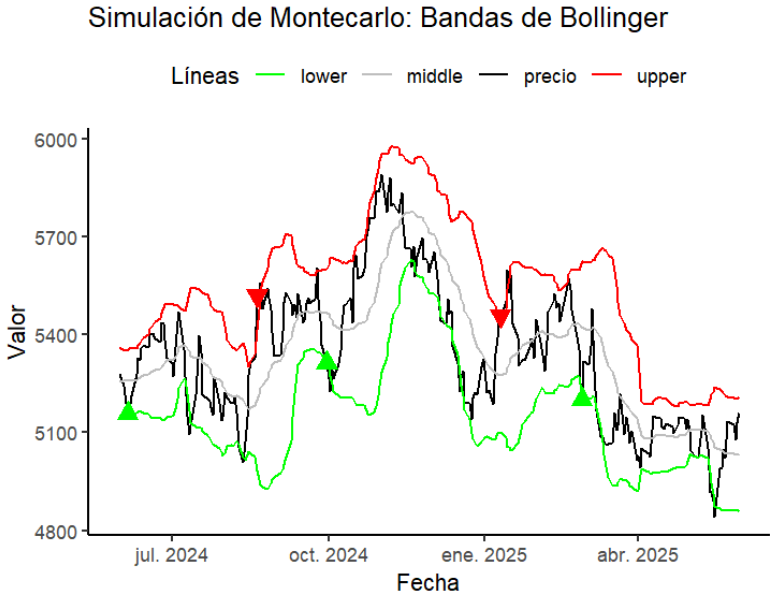

Gráfico 11. Señales bandas de Bollinger

4.4.1 Análisis de la rentabilidad

En este apartado se van a presentar los resultados de la estrategia relacionados con la rentabilidad. Para ello se han utilizado una serie de medidas:

```
> print(paste("El total de operaciones es: ", total_operaciones))
[1] "El total de operaciones es:  2800"
> print(paste("La rentabilidad media por operación es: ", rentabilidad_media))
[1] "La rentabilidad media por operación es:  0.0158550891412894"
> print(paste("El profit factor es: ", profit_factor))
[1] "El profit factor es:  1.84364022830628"
> print(paste("La rentabilidad media por simulación es: ", rentabilidad_media_sim))
[1] "La rentabilidad media por simulación es:  0.0443942495956103"
> print(paste("La rentabilidad de comprar y esperar es: ", buy_and_wait))
[1] "La rentabilidad de comprar y esperar es:  0.0528018167276405"
> print(paste("El porcentaje de éxito de las operaciones es: ", winning_percentage))
[1] "El porcentaje de éxito de las operaciones es:  0.693928571428571"
```

El rendimiento de la estrategia de *trading* basándose en tocar las bandas de Bollinger parece bueno. Las probabilidades de éxito de las operaciones realizadas son elevadas (69,39%), y la rentabilidad media de cada una de las operaciones es del 1,58%, un valor muy adecuado. Por otro lado, no hay que olvidar los costes de operar, puesto que, en 1000 simulaciones, se han ejecutado 2800 operaciones que equivalen a una media de 2,8 operaciones en cada simulación y el doble de transacciones de compraventa. La rentabilidad media por simulación es de (4,43%), por lo que no supera la rentabilidad del activo (5,28%), haciendo que sea más rentable comprar al inicio y no vender en todo el periodo que esta estrategia para hacer *trading*.

Analizando el gráfico 12, se puede ver cómo la distribución de las rentabilidades tiende a concentrarse más en la parte positiva, aunque tiene una dispersión apreciable.

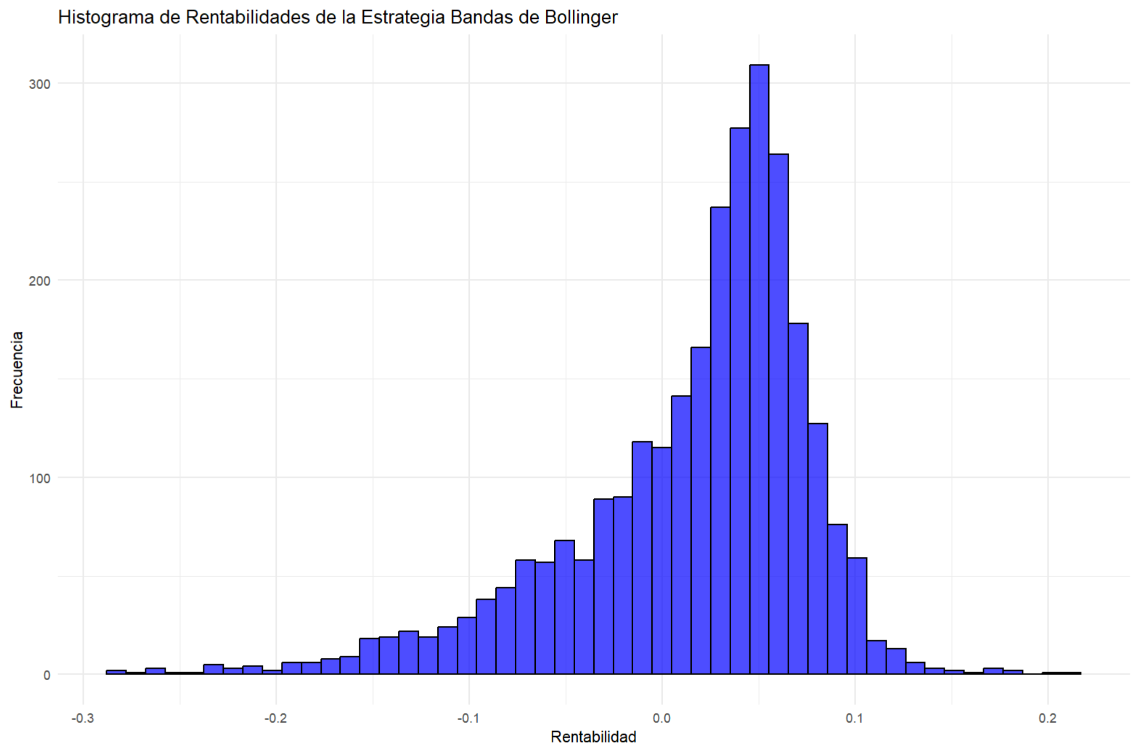

Gráfico 12. Histograma bandas Bollinger. Rentabilidad

4.4.2 Análisis del riesgo

```
> print(paste("La volatilidad de todas las rentabilidades es: ", volatilidad_rentabilidades))
[1] "La volatilidad de todas las rentabilidades es:  0.06350536542575"
> print(paste("El Sharpe ratio con la media de las rentabilidades es: ", sharpe_rentabilidades))
[1] "El Sharpe ratio con la media de las rentabilidades es:  -0.0652686718818553"
> print(paste("La media de las volatilidades es: ", volatilidad))
[1] "La media de las volatilidades es:  0.196468035334898"
> print(paste("La media de Sharpe ratio de las simulaciones  es: ", sharpe))
[1] "La media de Sharpe ratio de las simulaciones  es:  -0.123319641038832"
```

Con respecto al riesgo tenemos que la volatilidad de las rentabilidades es alta (6,35%), ya que como se pudo apreciar en el gráfico 12, existe algo de dispersión en cuanto a la rentabilidad de cada simulación.

Por otro lado, analizando los ratios de Sharpe, son negativos (-0,06 y -0,12), lo que implica que la rentabilidad asociada al riesgo no es positiva, y es mejor buscar otra estrategia o invertir en un activo libre de riesgo. Se puede apreciar cómo la distribución de los ratios de cada simulación en el gráfico está algo más concentrada en la parte negativa, al igual que la dispersión de los ratios.

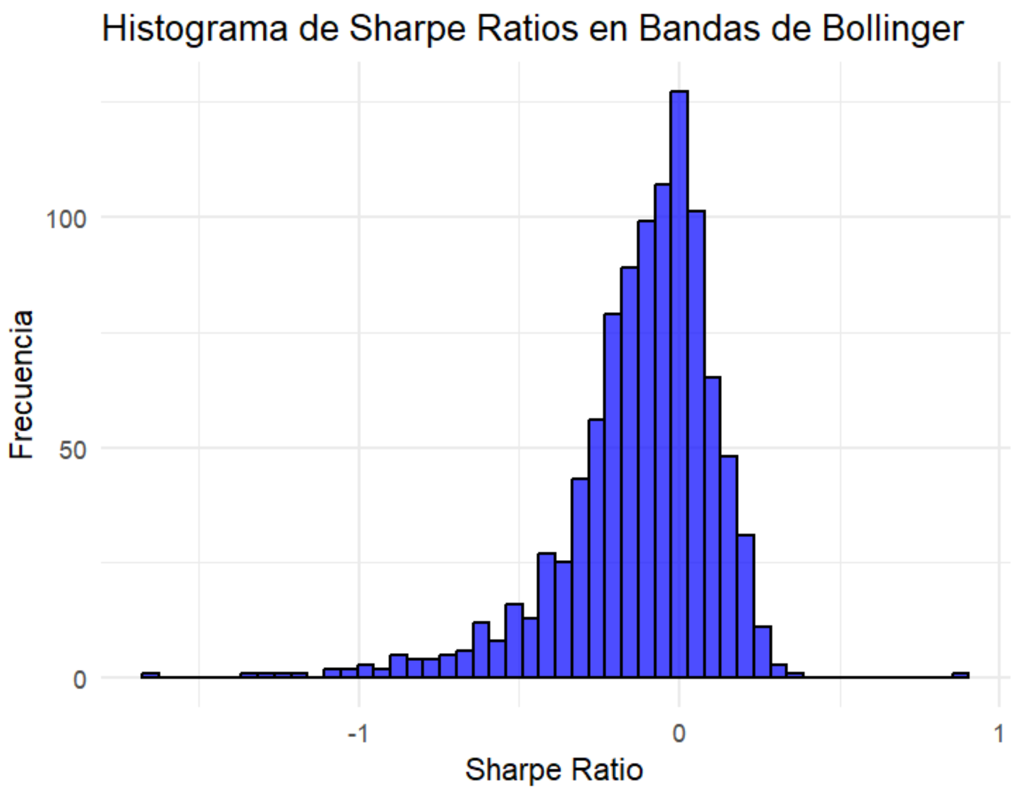

Gráfico 13. Histograma bandas Bollinger. Sharpe ratios.

Capítulo 5 – Conclusiones

5.1 Resumen de resultados

Después de analizar los resultados podemos comparar las estrategias de *trading* de compraventa basadas en indicadores técnicos.

La estrategia de MACD es la peor de las tres. No ofrece prácticamente rentabilidad y tiene un gran número de operaciones sin porcentaje de éxito. Además, ajustada al riesgo también es la que peor resultados ofrece.

Las estrategias de RSI y bandas de Bollinger son algo similares en cuanto a rentabilidad, teniendo la primera menos operaciones de media que la segunda. Los resultados son bastante parejos y no destaca demasiado una sobre otra. Por otro lado, en cuanto al riesgo ambas dan resultados negativos que sugieren la decisión de optar por otro tipo de inversión.

En términos generales, la conclusión es que no merece la pena seguir ninguno de estos indicadores de forma aislada por la rentabilidad y el riesgo asociado que tiene operar de esta manera. Es más rentable comprar el activo sin riesgo o comprar y esperar a los resultados que te proporcione de manera natural.

5.2 Limitaciones

Existen una serie de limitaciones que no se han tenido en cuenta a la hora de realizar los análisis y que son importantes mencionar de cara a posibles sesgos.

No se tienen en cuenta en el modelo las simulaciones que tienen una señal de compra, pero no de venta. De esta forma, en esa simulación no hay operaciones de compraventa y por ello no tiene la misma repercusión en los resultados que las demás simulaciones.

Por otro lado, en muchas simulaciones con operaciones se efectúa una última compra que no tiene venta. En esa situación no se computa como operación ni se tiene en cuenta para el análisis, aunque seguramente a posteriori del periodo simulado habría una señal de venta y se podría evaluar.

No se han hecho optimizaciones o combinaciones de indicadores para mejorar las estrategias. Es algo muy interesante que se podría realizar en un futuro para seguir explorando cuáles son los indicadores que mejor funcionan, en línea con el objetivo de obtener rentabilidades superiores en un corto plazo.

Capítulo 6 – Bibliografía

Fan, J., & Yao, Q. (n.d.). The Elements of Financial Econometrics (Excerpt). Cambridge University Press. https://assets.cambridge.org/97811071/91174/excerpt/9781107191174_excerpt.pdf (Recuperado el 2 de junio de 2024).

Glasserman, P. (2004). Monte Carlo Methods in Financial Engineering. Springer Science & Business Media.

Intercontinental Exchange. (2022, 21 de diciembre). NYSE Group Announces 2023, 2024 and 2025 Holiday and Early Closings Calendar. [Comunicado de prensa]. https://ir.theice.com/press-release/nye-group-announces-2023-2024-and-2025-holiday-and-early-closings-calendar (Recuperado el 2 de junio de 2024).

Liu, P. (2023). Quantitative Trading Strategies Using Python. Apress.

Posit team (2024). RStudio: Integrated Development Environment for R. Posit Software, PBC, Boston, MA. URL http://www.posit.co/.

R Core Team (2023). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

RDocumentation. (n.d.). RSI: Relative Strength Index. https://www.rdocumentation.org/packages/TTR/versions/0.24.4/topics/RSI (Recuperado el 2 de junio de 2024)

Ryan JA, Ulrich JM (2024). quantmod: Quantitative Financial Modelling Framework. R package version 0.4.26, https://CRAN.R-project.org/package=quantmod.

Shonkwiler, R. W. (2013). Finance with Monte Carlo. Springer.

Sievert, C. (2020). Interactive Web-Based Data Visualization with R, plotly, and shiny. Chapman and Hall/CRC Florida.

Tesoro Público de España. (n.d.). Página principal. https://www.tesoro.es/index.aspx (Recuperado el 2 de junio de 2024).

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). “Welcome to the tidyverse.” Journal of Open Source Software, 4(43), 1686. doi:10.21105/joss.01686 https://doi.org/10.21105/joss.01686.

Yahoo Finance España. (s.f.). ^GSPC - S&P 500. https://es.finance.yahoo.com/quote/%5EGSPC/ (Recuperado el 2 de junio de 2024).

Capítulo 7 – Anexos

7.1 Código R

7.1.1 Código común

```
# Instalar y cargar paquetes necesarios
library(quantmod)
library(tidyverse)
library(plotly)

# Descargar datos históricos del precio de cierre del S&P 500
getSymbols("^GSPC", from = "2000-01-01", to = Sys.Date(), warnings = FALSE, src = "yahoo")

# Extraer solo el precio de cierre
sp500_cierre <- Cl(GSPC)
# Convertir el objeto a un dataframe y agregar la fecha como una columna
sp500_df <- data.frame(date = index(sp500_cierre), close = as.numeric(sp500_cierre))
#Festivos del día 10 junio a un año vista de la bolsa de valores de EE.UU
dias_festivos <- c("2024-06-19", "2024-09-02", "2025-01-01", "2025-20-01",
                   "2025-02-17", "2025-04-18", "2025-05-26")

fechas <- seq(from = max(sp500_df$date), by = "day", length.out = 365)

# Filtrar las fechas para excluir los fines de semana (sábados y domingos) y festivos
fechas_abierto <- fechas[as.POSIXlt(fechas)$wday %in% 1:5 & !fechas %in% dias_festivos]

# Convertir las fechas en un dataframe
fechas_df <- data.frame(date = fechas_abierto)

#Los dias que va a tener la serie (quito los fines de semana porque no hay bolsa)
num_dias <- nrow(fechas_df)

# Configurar parámetros para simulación
set.seed(127)
num_simulations <- 1000
p_actual <- tail(sp500_df$close, 1)
t = num_dias #simulo un año de bolsa
ret <- diff(log(sp500_df$close)) # Calcular retornos diarios logarítmicos
dt <- 1/t #los dias que tiene la serie)
mu<- (mean(ret)/dt) #Calculo de media rendimientos (mu)
desv<- sqrt(var(ret)/dt) #Calculo de desviacion típica (alfa)
```

```
# Inicializar vectores para almacenar los resultados de cada simulación (RIESGO)
sharpe_ratios <- numeric(num_simulations)
rentabilidades_medias <- numeric(num_simulations)
volatilidades <- numeric(num_simulations)
tasa_libre_riesgo <- 0.03405

ganancias <- 0
perdidas <- 0
rentabilidad <- c()
op_ganadora <- 0
op_perdedora <- 0
```

```
# Calcular métricas de rendimiento
total_operaciones <- op_ganadora + op_perdedora
rentabilidad_media <- mean (rentabilidad)
profit_factor <- ganancias / abs(perdidas)
rentabilidad_media_sim <- rentabilidad_media * total_operaciones / num_simulations
buy_and_wait <- (mean(gbm[nrow(gbm), ]) / p_actual) - 1
winning_percentage <- op_ganadora / total_operaciones

#Calcular métricas de riesgo

#Calcula la volatilidad de todas las rentabilidades y el sharpe ratio
volatilidad_rentabilidades <- sd(rentabilidad)
sharpe_rentabilidades <- (rentabilidad_media - 0.02) / volatilidad_rentabilidades
#Calcula la volatilidad de todos los precios y el sharpe ratio como media de los de cada sim
sharpe <- mean(sharpe_ratios)
volatilidad <- mean(volatilidades)

print(paste("El total de operaciones es: ", total_operaciones))
print(paste("La rentabilidad media por operación es: ", rentabilidad_media))
print(paste("El profit factor es: ", profit_factor))
print(paste("La rentabilidad media por simulación es: ", rentabilidad_media_sim))
print(paste("La rentabilidad de comprar y esperar es: ", buy_and_wait))
print(paste("El porcentaje de éxito de las operaciones es: ", winning_percentage))
print(paste("La volatilidad de todas las rentabilidades es: ", volatilidad_rentabilidades))
print(paste("El Sharpe ratio con la media de las rentabilidades es: ", sharpe_rentabilidades))
print(paste("La media de las volatilidades es: ", volatilidad))
print(paste("La media de Sharpe ratio de las simulaciones  es: ", sharpe))
```

7.1.2 Código de la estrategia RSI

```
gbm <- matrix(nrow = t, ncol = num_simulations)
operaciones <- matrix(nrow = t, ncol = num_simulations)
rsi <- matrix(nrow = t, ncol = num_simulations)

for (sim in 1:num_simulations) {
  gbm[1, sim] <- p_actual
  lista_combinada <- tail(sp500_df$close, 15)
  # Calcular el RSI
  rsi[1, sim] <- tail(RSI(lista_combinada, n = 14, type = "SMA"), 1)

  compra <- 0
  num_op_sim <- 0

  for (day in 2:t) {
    zt <- rnorm(1)
    gbm[day, sim] <- exp((mu - desv^2 / 2) * dt + desv * zt * sqrt(dt)) * gbm[day - 1, sim]

    if (day <= 14) {
      lista_combinada <- c(lista_combinada[-1], gbm[day, sim])
      rsi[day, sim] <- tail(RSI(lista_combinada, n = 14, type = "SMA"), 1)
    } else {
      rsi[day, sim] <- tail(RSI(gbm[(day-14):day, sim], n = 14, type = "SMA"), 1)
    }
```

```r
    if (rsi[day, sim] < 30 && compra == 0) {
      compra <- gbm[day, sim]
      operaciones[day, sim] <- "Compra"
    }
    else if (rsi[day, sim] > 70 && compra != 0) {
      if (compra < gbm[day, sim]) {
        ganancias <- ganancias + gbm[day, sim] - compra
        op_ganadora <- op_ganadora + 1

      } else {
        perdidas <- perdidas + compra - gbm[day, sim]
        op_perdedora <- op_perdedora + 1
      }
      porcentaje <- (gbm[day, sim] / compra) - 1
      rentabilidad <- c(rentabilidad, porcentaje)
      rentabilidades_medias[sim] <- rentabilidades_medias[sim] + porcentaje
      num_op_sim <- num_op_sim + 1
      compra <- 0
      operaciones[day, sim] <- "Venta"
    }
  }
  # Calcular retornos diarios
  retornos_diarios <- diff(log(gbm[, sim]))
  # Calcular rentabilidad media por simulación
  rentabilidades_medias[sim] <-  rentabilidades_medias[sim] / num_op_sim
  # Reemplazar los valores NaN en sharpe_ratios por 0
  rentabilidades_medias[sim] <- replace(rentabilidades_medias[sim], is.nan(rentabilidades_medias[sim]), 0)
  # Calcular volatilidad diaria
  volatilidad_diaria <- sd(retornos_diarios)
  volatilidades[sim] <- volatilidad_diaria * sqrt(num_dias)
  # Calcular Sharpe Ratio para la simulación
  sharpe_ratios[sim] <- (rentabilidades_medias[sim] - tasa_libre_riesgo) / (volatilidad_diaria * sqrt(num_dias))

}
```

```r
# Histograma de los Sharpe Ratios
histograma_sharpe <- ggplot(data.frame(sharpe_ratios), aes(x = sharpe_ratios)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Sharpe Ratio") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Sharpe Ratios en RSI") +
  theme_minimal()



# Crear el histograma de rentabilidades
histograma_rentabilidad <- ggplot(data.frame(rentabilidad), aes(x = rentabilidad)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Rentabilidad") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Rentabilidades de la Estrategia RSI") +
  theme_minimal()



# Combinar la matriz gbm y el dataframe de fechas
gbm_df <- cbind(fechas_df, gbm)
operaciones_df <- cbind(fechas_df, operaciones)
rsi_df <- cbind(fechas_df, rsi)


# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(operaciones_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en gbm_df para que coincidan con las de gbm_long
colnames(gbm_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en gbm_df para que coincidan con las de gbm_long
colnames(rsi_df)[-1] <- as.character(1:num_simulations)
```

```
# Convertir gbm_df a formato largo para graficar
gbm_long <- gbm_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "precio")%>%
  filter(!is.na(precio))

# Convertir operaciones_df a formato largo y filtrar operaciones
operaciones_long <- operaciones_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "operacion") %>%
  filter(!is.na(operacion))


# Añadir precios correspondientes a operaciones_long
operaciones_long <- operaciones_long %>%
  left_join(gbm_long, by = c("date", "simulacion"))

# Dibujar el gráfico de todas las simulaciones
p <- ggplot(gbm_long, aes(date, precio, color = simulacion)) +
  geom_line() +
  xlab("Fecha") + ylab("Precio de la acción") +
  labs(title = "Simulación de Montecarlo: Movimiento Browniano Geométrico") +
  theme_classic() +
  theme(legend.position = "none")

# Añadir puntos de compra y venta
p <- p +
  geom_point(data = filter(operaciones_long, operacion == "Compra"),
             aes(date, precio),
             shape = 24, color = "green", fill = "green", size = 3) +
  geom_point(data = filter(operaciones_long, operacion == "Venta"),
             aes(date, precio),
             shape = 25, color = "red", fill = "red", size = 3)
```

```
# Convertir rsi_df a formato largo para graficar
rsi_long <- rsi_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "rsi")

# Añadir precios correspondientes a operaciones_long
operaciones_long <- operaciones_long %>%
  left_join(rsi_long, by = c("date", "simulacion"))

# Dibujar el gráfico de la evolución del RSI
p_rsi <- ggplot(rsi_long, aes(date, rsi, color = simulacion)) +
  geom_line() +
  geom_hline(yintercept = 30, linetype = "dashed", color = "green") +
  geom_hline(yintercept = 50, linetype = "dashed", color = "grey") +
  geom_hline(yintercept = 70, linetype = "dashed", color = "red") +
  geom_point(data = filter(operaciones_long, operacion == "Compra"),
             aes(date, rsi),
             shape = 24, color = "green", fill = "green", size = 3) +
  geom_point(data = filter(operaciones_long, operacion == "Venta"),
             aes(date, rsi),
             shape = 25, color = "red", fill = "red", size = 3) +
  xlab("Fecha") + ylab("RSI") +
  labs(title = "Evolución del RSI en las Simulaciones") +
  theme_classic() +
  theme(legend.position = "none")

print(p)
print(p_rsi)
print(histograma_rentabilidad)
print(histograma_sharpe)
```

7.1.3 Código de la estrategia MACD

```r
gbm <- matrix(nrow = t, ncol = num_simulations)
operaciones <- matrix(nrow = t, ncol = num_simulations)
signal <- matrix(nrow = t, ncol = num_simulations)
macd <- matrix(nrow = t, ncol = num_simulations)

for (sim in 1:num_simulations) {
  gbm[1, sim] <- p_actual

  lista_combinada <- tail(sp500_df$close, 35) #MIRAR TAMAÑO LISTA
  # Calcular el MACD
  macd_aux <- MACD(lista_combinada, nFast=12, nSlow=26, nSig=9, maType="EMA")
  # Obtener el último valor del MACD
  macd[1, sim] <- tail(macd_aux, 1)[1]  # El primer valor es el MACD

  # Obtener el último valor de la señal
  signal[1, sim] <- tail(macd_aux, 1)[2]  # El segundo valor es la señal


  # Comparar el MACD y la línea de señal para determinar si comprar o vender
  if (macd[1, sim] > signal[1, sim]) {
    compra <- p_actual
    operaciones[day, sim] <- "Compra"
  } else {
    compra <- 0
  }
  num_op_sim <- 0
```

```r
  for (day in 2:t) {
    zt <- rnorm(1)
    dt = 1 / t
    gbm[day, sim] <- exp((mu - desv^2 / 2) * dt + desv * zt * sqrt(dt)) * gbm[day - 1, sim]
    if (day <= 35){
      # Añadir el nuevo valor al final de la lista
      lista_combinada <- c(lista_combinada[-1], gbm[day, sim])
      macd_aux <- MACD(lista_combinada, nFast=12, nSlow=26, nSig=9, maType="EMA")
    }
    else {
      macd_aux <- MACD(gbm[(day-35):day, sim], nFast=12, nSlow=26, nSig=9, maType="EMA")
    }
    # Obtener el último valor del MACD
    macd[day, sim] <- tail(macd_aux, 1)[1]  # El primer valor es el MACD
    # Obtener el último valor de la señal
    signal[day, sim] <- tail(macd_aux, 1)[2]  # El segundo valor es la señal
    # Comparar el MACD y la línea de señal para determinar si comprar o vender
    if (macd[day, sim] > signal[day, sim] && compra == 0) {
      compra <- gbm[day, sim]
      operaciones[day, sim] <- "Compra"
    } else if (macd[day, sim] < signal[day, sim]  && compra != 0) {
      if (compra < gbm[day, sim]){
        ganancias <- ganancias + gbm[day, sim] - compra
        op_ganadora <- op_ganadora + 1
      }
      else {
        perdidas <- perdidas + compra - gbm[day, sim]
        op_perdedora <- op_perdedora + 1
      }
      porcentaje <- (gbm[day, sim] / compra) - 1
      rentabilidad <- c(rentabilidad, porcentaje)
      rentabilidades_medias[sim] <- rentabilidades_medias[sim] + porcentaje
      num_op_sim <- num_op_sim + 1
      compra <- 0
      operaciones[day, sim] <- "Venta"
    }
```

```r
  }
  # Calcular retornos diarios
  retornos_diarios <- diff(log(gbm[, sim]))
  # Calcular rentabilidad media por simulación
  rentabilidades_medias[sim] <-  rentabilidades_medias[sim] / num_op_sim
  # Reemplazar los valores NaN en sharpe_ratios por 0
  rentabilidades_medias[sim] <- replace(rentabilidades_medias[sim], is.nan(rentabilidades_medias[sim]), 0)
  # Calcular volatilidad diaria
  volatilidad_diaria <- sd(retornos_diarios)
  volatilidades[sim] <- volatilidad_diaria * sqrt(num_dias)
  # Calcular Sharpe Ratio para la simulación
  sharpe_ratios[sim] <- (rentabilidades_medias[sim] - tasa_libre_riesgo) / (volatilidad_diaria * sqrt(num_dias))

}
```

```r
# Histograma de los Sharpe Ratios
histograma_sharpe <- ggplot(data.frame(sharpe_ratios), aes(x = sharpe_ratios)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Sharpe Ratio") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Sharpe Ratios en MACD") +
  theme_minimal()



# Crear el histograma de rentabilidades
histograma_rentabilidad <- ggplot(data.frame(rentabilidad), aes(x = rentabilidad)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Rentabilidad") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Rentabilidades de la Estrategia MACD") +
  theme_minimal()

# Combinar la matriz gbm y el dataframe de fechas
gbm_df <- cbind(fechas_df, gbm)
operaciones_df <- cbind(fechas_df, operaciones)
macd_df <- cbind(fechas_df, macd)
signal_df <- cbind(fechas_df, signal)


# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(operaciones_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(gbm_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(macd_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(signal_df)[-1] <- as.character(1:num_simulations)
```

```r
# Convertir gbm_df a formato largo para graficar
gbm_long <- gbm_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "precio")

# Convertir operaciones_df a formato largo y filtrar operaciones
operaciones_long <- operaciones_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "operacion") %>%
  filter(!is.na(operacion))

# Añadir precios correspondientes a operaciones_long
operaciones_long <- operaciones_long %>%
  left_join(gbm_long, by = c("date", "simulacion"))

# Convertir macd_df y signal_df a formato largo para graficar
macd_long <- macd_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "macd")

signal_long <- signal_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "signal")

# Dibujar el gráfico de todas las simulaciones
p <- ggplot(gbm_long, aes(date, precio, color = simulacion)) +
  geom_line() +
  xlab("Fecha") + ylab("Precio de la acción") +
  labs(title = "Simulación de Montecarlo: Movimiento Browniano Geométrico") +
  theme_classic() +
  theme(legend.position = "none")
```

```r
# Convertir macd_df y signal_df a formato largo para graficar
macd_long <- macd_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "macd")

signal_long <- signal_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "signal")

# Graficar las líneas de MACD y señal
macd_signal_long <- macd_long %>%
  left_join(signal_long, by = c("date", "simulacion"))

# Añadir precios correspondientes a operaciones_long
operaciones_long <- operaciones_long %>%
  left_join(macd_long, by = c("date", "simulacion"))

p_macd <- ggplot(macd_signal_long, aes(x = date)) +
  geom_line(aes(y = macd, color = "MACD")) +
  geom_line(aes(y = signal, color = "Señal")) +
  xlab("Fecha") + ylab("Valor") +
  labs(title = "Líneas MACD y Señal") +
  theme_classic() +
  scale_color_manual(name = "Líneas", values = c("MACD" = "blue", "Señal" = "red")) +
  theme(legend.position = "top")
```

```
p_macd <- p_macd +
  geom_point(data = filter(operaciones_long, operacion == "Compra"),
             aes(date, macd),
             shape = 24, color = "green", fill = "green", size = 3) +
  geom_point(data = filter(operaciones_long, operacion == "Venta"),
            aes(date, macd),
            shape = 25, color = "red", fill = "red", size = 3)

print(p)
print(p_macd)
print(histograma_rentabilidad)
print(histograma_sharpe)
```

7.1.4 Código de la estrategia Bandas de Bollinger

```
gbm <- matrix(nrow = t, ncol = num_simulations)
upperBollinger <- matrix(nrow = t, ncol = num_simulations)
lowerBollinger <- matrix(nrow = t, ncol = num_simulations)
middleBollinger <- matrix(nrow = t, ncol = num_simulations)
operaciones <- matrix(nrow = t, ncol = num_simulations)

for (sim in 1:num_simulations) {
  gbm[1, sim] <- p_actual
  lista_combinada <- tail(sp500_df$close, 20)

  # Calcular las Bandas de Bollinger
  bollinger_bands <- BBands(lista_combinada, n = 20, maType="SMA", sd = 2)
  lowerBollinger[1, sim] <- tail(bollinger_bands, 1)[1]
  middleBollinger[1, sim] <- tail(bollinger_bands, 1)[2]
  upperBollinger[1, sim] <- tail(bollinger_bands, 1)[3]


  if (p_actual < lowerBollinger[1, sim]){
    compra <- p_actual
    operaciones[day, sim] <- "Compra"
  }
  else {
    compra <- 0
  }
  num_op_sim <- 0
```

```
for (day in 2:t) {
  zt <- rnorm(1)
  dt = 1 / t
  gbm[day, sim] <- exp((mu - desv^2 / 2) * dt + desv * zt * sqrt(dt)) * gbm[day - 1, sim]

  if (day <= 19){
    # Añadir el nuevo valor al final de la lista
    lista_combinada <- c(lista_combinada[-1], gbm[day, sim])
    # Calcular las Bandas de Bollinger
    bollinger_bands <- BBands(lista_combinada, n = 20, maType="SMA", sd = 2)
  }
  else {
    bollinger_bands <- BBands(gbm[(day-19):day, sim], n = 20, maType="SMA", sd = 2)
  }
  lowerBollinger[day, sim] <- tail(bollinger_bands, 1)[1]
  middleBollinger[day, sim] <- tail(bollinger_bands, 1)[2]
  upperBollinger[day, sim] <- tail(bollinger_bands, 1)[3]

  if (gbm[day, sim] < lowerBollinger[day, sim] && compra == 0){
    compra <- gbm[day, sim]
    operaciones[day, sim] <- "Compra"
  }
  else if (gbm[day, sim] > upperBollinger[day, sim] && compra != 0){
    if (compra < gbm[day, sim]){
      ganancias <- ganancias + gbm[day, sim] - compra
      op_ganadora <- op_ganadora + 1
    }
    else {
      perdidas <- perdidas + compra - gbm[day, sim]
      op_perdedora <- op_perdedora + 1
    }
    porcentaje <- (gbm[day, sim] / compra) - 1
    rentabilidad <- c(rentabilidad, porcentaje)
```

```
      rentabilidades_medias[sim] <- rentabilidades_medias[sim] + porcentaje
      num_op_sim <- num_op_sim + 1
      compra <- 0
      operaciones[day, sim] <- "Venta"
    }
  }
  # Calcular retornos diarios
  retornos_diarios <- diff(log(gbm[, sim]))
  # Calcular rentabilidad media por simulación
  rentabilidades_medias[sim] <-  rentabilidades_medias[sim] / num_op_sim
  # Reemplazar los valores NaN en sharpe_ratios por 0
  rentabilidades_medias[sim] <- replace(rentabilidades_medias[sim], is.nan(rentabilidades_medias[sim]), 0)
  # Calcular volatilidad diaria
  volatilidad_diaria <- sd(retornos_diarios)
  volatilidades[sim] <- volatilidad_diaria * sqrt(num_dias)
  # Calcular Sharpe Ratio para la simulación
  sharpe_ratios[sim] <- (rentabilidades_medias[sim] - tasa_libre_riesgo) / (volatilidad_diaria * sqrt(num_dias))

}
```

```r
# Histograma de los Sharpe Ratios
histograma_sharpe <- ggplot(data.frame(sharpe_ratios), aes(x = sharpe_ratios)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Sharpe Ratio") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Sharpe Ratios en Bandas de Bollinger") +
  theme_minimal()

# Crear el histograma de rentabilidades
histograma_rentabilidad <- ggplot(data.frame(rentabilidad), aes(x = rentabilidad)) +
  geom_histogram(bins = 50, fill = "blue", color = "black", alpha = 0.7) +
  xlab("Rentabilidad") +
  ylab("Frecuencia") +
  ggtitle("Histograma de Rentabilidades de la Estrategia Bandas de Bollinger") +
  theme_minimal()


# Combinar la matriz gbm y el dataframe de fechas
gbm_df <- cbind(fechas_df, gbm)
upperBollinger_df <- cbind(fechas_df, upperBollinger)
lowerBollinger_df <- cbind(fechas_df, lowerBollinger)
middleBollinger_df <- cbind(fechas_df, middleBollinger)
operaciones_df <- cbind(fechas_df, operaciones)


# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(operaciones_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(gbm_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(upperBollinger_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(lowerBollinger_df)[-1] <- as.character(1:num_simulations)
# Cambiar nombres de las columnas en operaciones_df para que coincidan con las de gbm_long
colnames(middleBollinger_df)[-1] <- as.character(1:num_simulations)
```

```r
# Convertir gbm_df a formato largo para graficar
gbm_long <- gbm_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "precio")

# Convertir operaciones_df a formato largo y filtrar operaciones
operaciones_long <- operaciones_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "operacion") %>%
  filter(!is.na(operacion))

# Añadir precios correspondientes a operaciones_long
operaciones_long <- operaciones_long %>%
  left_join(gbm_long, by = c("date", "simulacion"))

upper_long <- upperBollinger_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "upper")

lower_long <- lowerBollinger_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "lower")

middle_long <- middleBollinger_df %>%
  pivot_longer(-date, names_to = "simulacion", values_to = "middle")

bandas_long <- upper_long %>%
  left_join(middle_long, by = c("date", "simulacion"))

bandas_long <- bandas_long %>%
  left_join(lower_long, by = c("date", "simulacion"))

bandas_long <- bandas_long %>%
  left_join(gbm_long, by = c("date", "simulacion"))
```

```
p <- ggplot(bandas_long, aes(x = date)) +
  geom_line(aes(y = precio, color = "precio")) +
  geom_line(aes(y = upper, color = "upper")) +
  geom_line(aes(y = middle, color = "middle")) +
  geom_line(aes(y = lower, color = "lower")) +
  xlab("Fecha") + ylab("Valor") +
  labs(title = "Simulación de Montecarlo: Bandas de Bollinger") +
  theme_classic() +
  scale_color_manual(name = "Líneas", values = c("precio" = "black", "upper" = "red", "middle" = "gray", "lower" = "green"
  theme(legend.position = "top")

p <- p +
  geom_point(data = filter(operaciones_long, operacion == "Compra"),
             aes(date, precio),
             shape = 24, color = "green", fill = "green", size = 3) +
  geom_point(data = filter(operaciones_long, operacion == "Venta"),
             aes(date, precio),
             shape = 25, color = "red", fill = "red", size = 3)

print(p)
print(histograma_rentabilidad)
print(histograma_sharpe)
```