Universidad de San Andrés

Escuela de Administración y Negocios

Licenciatura en Finanzas

Hipótesis de mercados eficientes: Test sobre la volatilidad de precios al alcanzarse soportes y resistencias

Autores: Alejo Genoud y Erika Yael Kleiman

DNI: 42.646.287 y 42.724.237

Director de Tesis: Gabriel Basaluzzo

Agosto de 2022

Tabla de contenidos

Resumen

El presente trabajo explora la validez de la hipótesis de mercado eficiente mediante el estudio de la volatilidad en el retorno de acciones al aproximarse a zonas de soportes/resistencias sugeridas por análisis técnico. Se utilizaron series históricas de quince instrumentos representativos que cotizan en el NYSE y el NASDAQ. El análisis fue llevado a cabo para un horizonte de cuatro años, del 1 de enero de 2018 al 1 de enero de 2022. La frecuencia de datos es diaria. Los resultados sugieren que existe una diferencia en las funciones de distribución de volatilidad y de retornos (para un gran número de instrumentos) para las observaciones que se encuentran en zonas de soporte/resistencia. Estas diferencias pueden presentarse tanto en ambas zonas de interés (soportes y resistencias) como en un solo grupo de interés (ej. Si en resistencias, no en soportes). La significatividad estadística encontrada en los *tests* prueba la existencia de diferencias en el comportamiento de precios, sugiriendo que los mercados no son eficientes.

1. Introducción

La validez de la hipótesis de mercado eficiente ha sido objeto de estudio durante décadas, donde la literatura académica financiera aún no ha logrado consensuar en un resultado definitivo. Debido a esto, se considera la descripción y el modelado del comportamiento de precios como uno de los principales paradigmas de las finanzas modernas. Esta teoría, inicialmente propuesta por Eugene Fama (1970), establece que el precio actual de un activo en el mercado contiene toda la información disponible, y no es posible predecir movimientos futuros mediante el análisis de estados financieros o comportamiento previo del instrumento. Como señala Malkiel (2003), la hipótesis de mercado eficiente plantea que los precios siguen un paseo aleatorio – o *random walk* – y su comportamiento se modela en base a desviaciones aleatorias e impredecibles. Como resultado, en un mercado eficiente, no existe forma de anticipar movimientos de precios y no es posible generar de manera consistente rendimientos superiores a los alcanzados por medio de una estrategia de inversión pasiva.

En contra de la hipótesis anterior, se encuentra la escuela del análisis técnico. Lejos de creer en que los precios siguen un *random walk*, los analistas se basan en una premisa principal: que la historia se repite. El analista técnico busca identificar patrones en series históricas, así intentando predecir movimientos futuros de precio. Dentro de estas figuras, las más elementales son los soportes y las resistencias. Murphy (1986) explica que un soporte es un nivel de precios donde el interés por comprar es lo suficientemente fuerte como para vencer la presión de vender, generando como resultado la detención de la caída de precios y una reversión de tendencia. Análogamente, una resistencia ocurre en un nivel de precios donde la presión por vender vence a la fuerza compradora, deteniendo la suba y dando lugar a una potencial caída.

El modelado de volatilidad busca desarrollar un proceso donde la volatilidad no es constante, sino dependiente del tiempo. A lo largo de los años los modelos fueron evolucionando, con la gran mayoría tomando como base el modelo de heterocedasticidad condicional autorregresiva (ARCH) de Engle (1982). En el presente trabajo utilizaremos el modelo de media móvil exponencialmente ponderada (EWMA) desarrollado por JPMorgan para modelar la volatilidad de nuestras series de precios. El estudio de evento es una metodología utilizada para cuantificar el impacto económico de un cierto acontecimiento en el valor de la firma. La utilidad de dicha prueba nace de la premisa

que, asumiendo racionalidad en el mercado, el efecto de un evento se verá reflejado inmediatamente en el precio del instrumento[1].

En caso de que la hipótesis de mercado eficiente fuera cierta, el comportamiento de precios no debería cambiar en los niveles de soporte/resistencia definidos por los analistas técnicos. Al plantear que los precios siguen un paseo al azar y su movimiento depende exclusivamente de la aparición de nueva información, no habría razón para un cambio en comportamiento por aproximarse a un determinado precio. El objetivo de este trabajo es llevar a cabo un estudio de evento para verificar si existe evidencia estadística de una disminución en la volatilidad de los retornos al llegar a niveles de soporte/resistencia. De ser el caso, quedaría refutada la hipótesis de mercado eficiente al probar que el comportamiento de precios no es aleatorio.

[1] Campbell, John Y, et al. *The Econometrics of Financial Markets*. New Age International Pvt Ltd Publishers, 2007.

2. Revisión de literatura

2.1. Análisis Técnico

Diversos estudios académicos han intentado validar la efectividad del análisis técnico, utilizando distintos periodos de tiempo e instrumentos financieros. Sin embargo, las investigaciones sobre los niveles de soporte y resistencia resultan ser escasas y distantes entre sí. A pesar no haber logrado alcanzar un consenso a favor o en contra de esta práctica, destacamos los siguientes estudios los cuales consideramos relevantes para nuestro trabajo:

Brock et al. (1992) investigaron dos de las más conocidas herramientas del análisis técnico: las medias móviles y los niveles de soporte y resistencia. Su estudio se concentró en la aplicación de herramientas de modelado de volatilidad como *random-walk with drift*, AR(l), GARCH-M y EGARCH, sobre el índice Dow Jones para el periodo que abarca los años de 1897 hasta 1986. En sus resultados encontraron evidencia a favor del uso del análisis técnico invalidando el concepto de la existencia de precios eficientes, pero con el supuesto de costos de transacción nulos. Por otro lado, Osler (2000) verifica la capacidad predictiva de los niveles de soporte y resistencia en los mercados de tipos de cambio utilizando como datos los niveles publicados por diversas empresas financieras entre 1996 y 1998. En su trabajo concluye que los niveles de soporte y resistencia tienen una capacidad estadísticamente significativa para predecir cambios o finales de tendencia y que dichas predicciones pueden durar hasta 5 días hábiles después de la fecha de publicación de los niveles. A su vez, corrige inconvenientes de kurtosis que no fueron tomados en cuenta en el trabajo académico anterior. Finalmente, en línea con los estudios anteriores, Garzarelli et al. (2014) investigó la probabilidad de reversión de tendencia en los niveles de soporte y resistencia para nueve acciones que cotizan en la Bolsa de Valores de Londres. Su análisis abarca el periodo de 2002 y concluyó que los precios tienden a rebotar en los niveles identificados y que, a mayor cantidad de toques, mayor será la probabilidad de que ocurra un nuevo rebote sobre dicho rango de valores.

2.2. Modelos de volatilidad

En la teoría clásica de series temporales (metodología de Box-Jenkins), el desarrollo estadístico de volatilidad se realiza a partir de un proceso estocástico estacionario; es decir (en sentido amplio o débil) de un proceso con media constante, varianza constante y correlación entre dos observaciones distintas igual a la de otras dos cualquiera separada por la misma distancia (mismo número de períodos)[2]. El modelado de volatilidad se originó bajo la suposición que la misma no es constante en el tiempo, sino dependiente de diferentes factores de innovación denominados ε_t (también referido como término de error). Engle (1982), mediante el estudio de la inflación del Reino Unido, fue el primero en introducir un modelo de volatilidad de heterocedasticidad condicional autorregresiva. Este modelo, comúnmente llamado ARCH por sus siglas en inglés,

[2] Borda, R. Arce. "20 Años De Modelos ARCH: Una Visión De Conjunto De Las Distintas Variantes De La Familia." *Redalyc.org*, Asociación Internacional De Economía Aplicada, 1 Jan. 1970, https://www.redalyc.org/articulo.oa?id=30122111.

fue estudiado durante muchos años, resultando en diferentes variantes del mismo. Dentro de las numerosas parametrizaciones publicadas, el modelo ARCH generalizado de Bollerslev (1986) (GARCH (p, q)) es considerada como una de las más exitosas. Un GARCH (p, q) modela la volatilidad como una función de rezagos de errores y volatilidades al cuadrado, y puede estimarse por máxima verosimilitud una vez que la distribución del error haya sido especificada (tiende a suponerse normalidad). Estudios empíricos sugieren que el modelo GARCH (1,1) es el más utilizado al trabajar con series financieras. Dentro de las extensiones de los modelos GARCH, se encuentra el modelo de media móvil exponencialmente ponderada (EWMA, por sus siglas en inglés) publicado por JPMorgan (1996). Este modelo parte de un GARCH (1,1) donde las ponderaciones de las volatilidades pasadas disminuyen exponencialmente a medida que se retrocede en el tiempo. Esta aproximación al modelado de la volatilidad es atractiva ya que se requiere almacenar pocos datos. Para calcular la volatilidad en un momento τ solo se necesita la varianza estimada en el momento $\tau - 1$ y la observación del día τ. En el presente trabajo se llevará a cabo el modelado de volatilidad mediante EWMA, ya que se obtiene una reacción inmediata de la volatilidad a movimientos de precios diarios y por ende proporciona una buena estimación para analizar la existencia de cambios en zonas de soporte/resistencia.

2.3. Estudio de evento

El primer trabajo que utilizó la metodología de *event study* fue publicado por Dolley (1933), y en él se analizan cambios de precio nominales de acciones al momento de un *split*[3]. Utilizando una muestra de 95 acciones con *splits* de 1921 a 1931, Dolley encontró que el precio incrementó en 57 casos y cayó en 26 casos, con 12 casos donde no hubo efecto observable[4]. Luego, con el pasar de las décadas, los estudios de evento se fueron volviendo más sofisticados, teniendo como mejora la eliminación de movimientos generales de mercado y la separación de eventos de confusión. Estudios de esa época incluyen Myers y Bakay (1948), Barker (1956, 1957, 1958) y Ashley (1962). En la década del 60, Ball y Brown (1968) y Fama et al. (1969) introdujeron la metodología que sigue en uso hoy en día. Estas dos publicaciones son consideradas las más influyentes en el tema de estudio de eventos. Ball y Brown concluyeron que los estados financieros publicados por las compañías contienen información que está relacionada con el precio de la acción. Encontraron que el error en ingresos pronosticados, medidos por la diferencia entre ingreso reportado e ingreso esperado, tiene un impacto positivo al momento de la publicación de los estados financieros y por ende existe la posibilidad de obtener rendimientos anormales. Por otro lado, Fama et al. encontraron que luego del anuncio de un *split*, los precios de las acciones reflejan rápidamente toda la información disponible y no generan retornos anormales, demostrando eficiencia en los mercados de capitales.

Dentro de las publicaciones más recientes y vinculadas a nuestro tema de estudio, Mazza (2012) encontró evidencia de que ciertas configuraciones de precios *high-low-open-close* (HLOC) están asociadas a mayor liquidez. Si bien estos eventos son de corta duración, agentes de mercado podrían beneficiarse del incremento de liquidez ejecutando sus operaciones cuando estos patrones específicos HLOC ocurren. Esta publicación trata directamente el estudio de evento con el *price action* de un instrumento, asemejándose a lo tratado en nuestra investigación. Por otro lado, Tweneboah-Koduah et al. (2020) llevaron a cabo un estudio de evento analizando la reacción de la volatilidad de acciones frente al anuncio de *data breaches.* Con una muestra de 96 firmas

[3] Acción de multiplicar el número de acciones circulantes por un determinado factor

[4] Campbell, John Y, et al. *The Econometrics of Financial Markets*. New Age International Pvt Ltd Publishers, 2007.

listadas en el Standard & Poors 500 los autores encontraron que para acciones individuales existe evidencia de una diferencia de volatilidad antes y después del anuncio del *data breach,* y que la industria financiera es la que más volatilidad anormal presenta luego del evento.

3. Datos

Las series históricas de precios de los activos son de acceso libre y están disponibles en Yahoo Finance. Utilizamos datos diarios para el periodo del 1 de enero de 2018 al 1 de enero de 2022. Descargamos las series de los quince activos que seleccionamos y no hay datos faltantes. A su vez, ninguno de las compañías realizó un *split* de acciones, por lo cual no es necesario hacer ningún ajuste sobre los precios. Estas series históricas serán nuestros *inputs* para luego, con el uso de Python, identificar soportes/resistencias y realizar la inferencia estadística de la volatilidad.

A continuación, podemos observar las covarianzas entre los instrumentos y la correlación contra su índice representativo:

Figura 4.1.

	Varianza del activo	Indice	Correlación con Indice
AMZN	0,04%	NASDAQ	0,63
BP	0,06%	S&P 500	0,59
CVX	0,05%	S&P 500	0,70
DIS	0,04%	S&P 500	0,67
GOLD	0,05%	S&P 500	0,16
GOOGL	0,03%	NASDAQ	0,84
JNJ	0,02%	S&P 500	0,63
JPM	0,04%	S&P 500	0,77
NKE	0,04%	S&P 500	0,68
PEP	0,02%	S&P 500	0,70
PG	0,02%	S&P 500	0,60
PYPL	0,06%	NASDAQ	0,77
SBUX	0,03%	S&P 500	0,73
WMT	0,02%	S&P 500	0,48
XOM	0,04%	S&P 500	0,48

Podemos observar que las covarianzas entre instrumentos son muy bajas y las correlaciones con los índices son variadas, descartando la posibilidad de ocurrencia de soportes/resistencias en instrumentos individuales causados por el comportamiento de precios del índice.

4. Metodología

4.1. Análisis técnico

Antes de comenzar con nuestro análisis, resulta importante indagar en la definición de niveles de soporte y resistencia propuesta por algunos académicos del tema.

En un primer lugar, Murphy (1986) define el soporte como "*un nivel o área del gráfico por debajo del mercado donde el interés de compra es lo suficientemente fuerte como para vencer la presión por vender. Como resultado, hay una bajada que se detiene y los precios vuelven a subir. . . La resistencia es lo opuesto al soporte*. . .' (p.81). Por su parte, Bulkowski (2002) completa la descripción anterior proponiendo que *"una mejor definición incluiría varios mínimos o máximos menores deteniéndose cerca del mismo precio varias veces"*. Este autor sostiene que los niveles de soporte y de resistencia no son puntos de precios individuales sino áreas en forma de bandas gruesas que ralentizan, o incluso detienen, el movimiento de los precios.

A pesar de haber instituciones financieras que informan los niveles de soporte y resistencia como precios individuales, en la práctica no suele observarse la ocurrencia de mínimos y máximos locales en un nivel de precio preciso. Es por esto que combinaremos las afirmaciones de Murphy (1986) y Bulkowski (2002) antes mencionadas y a lo largo de este trabajo el nivel de soporte o de resistencia será considerado como un área de precios, en lugar de un precio individual específico, en el que residen máximos y mínimos locales los cuales actúan como barreras de reversión de tendencia.

El mecanismo para identificar las bandas de soporte y resistencia en un momento dado será buscar dentro de un período de tiempo precedente constante las mayores y menores precios. Para ello aplicaremos un enfoque de ventana móvil comúnmente utilizado en el análisis técnico. Un paso preliminar requerido implica la identificación de máximos y mínimos locales que se pueden representar mediante las siguientes ecuaciones:

$$Soporte_{\tau}\colon\ S_{\tau} = MIN[P_{\tau-w}, \dots, P_{\tau-1}, P_{\tau}]$$
$$Resistencia_{\tau}\colon\ R_{\tau} = MAX[P_{\tau-w}, \dots, P_{\tau-1}, P_{\tau}]$$

donde τ es el índice de tiempo y w la longitud de la ventana móvil. Aquí, el nivel de soporte (resistencia) en el momento τ es igual al precio de cierre mínimo (máximo) "P" de los períodos "w" anteriores. Utilizaremos una ventana móvil de 10 días y consideraremos como mínimo o máximo local al valor que sobreviva transcurrido 5 días (Osler, 2000).

Una vez obtenidos los máximos y mínimos locales, calculamos las bandas de soportes y resistencias. Para ello consideramos un nivel de soporte como $S_t \pm \gamma$ y un nivel de resistencia como $R_t \pm \gamma$, donde γ es el parámetro de ancho de nivel. Este parámetro representa la tolerancia de los inversores en un soporte o resistencia dado. Si el precio traspasa este umbral, los inversores consideran que el soporte o la resistencia se ha roto.

Seguimos el enfoque de Garzarelli et al. (2014), que acomoda diferentes resoluciones de series de precios, eligiendo γ como el incremento de precio promedio de toda la serie de precios definida como

$$\gamma = \frac{1}{T-1}\sum_{t=2}^{T}|P_t - P_{t-1}|$$

La identificación de un soporte/resistencia ocurre cuando dentro de la ventana móvil ocurren dos mínimos/máximos que se encuentran dentro del ancho de banda descripto.

4.2. Modelos de volatilidad

El modelo ARCH propuesto por Engle (1982) plantea que la volatilidad condicional de un determinado día *t* (y_t) depende del desvío estándar ($h_t^{\frac{1}{2}}$, raíz cuadrada de la varianza) y el factor de innovación, también llamado factor de error, (ε_t). Se supone que este factor de error sigue una distribución normal estándar

$$y_t = h_t^{\frac{1}{2}}\varepsilon_t\,, \quad donde \quad \varepsilon_t \sim N(0,1)$$

$$y$$

$$h_t = \alpha_0 + \sum_{i=1}^{p}\alpha_i y_{t-p}^2\,, \quad donde\ p = Número\ elegido\ de\ rezagos$$

Como podemos ver, la varianza h_t depende de parámetros α constantes y de la varianza condicional de *p* días anteriores. Si bien Engle introduce la dependencia de volatilidades anteriores, el ARCH planteado tiene diversas limitaciones. En primer lugar, el modelo asume el mismo impacto en *shocks* positivos y negativos, cuando en la práctica se sabe que la respuesta depende de la dirección del *shock*. Este supuesto puede conducir a sobreestimaciones de impacto para ciertos *shocks*. En segundo lugar, si los rezagos *p* no son numerosos, la volatilidad persiste poco en el tiempo.

Por otro lado, el GARCH (p,q) propuesto por Bollerslev (1986) modela la volatilidad en función de *p* rezagos de varianza (σ^2) y *q* rezagos del error cuadrado (ε^2):

$$\sigma_t^2 = \alpha_0 + \alpha_1\varepsilon_{t-1}^2 + \cdots + \alpha_q\varepsilon_{t-q}^2 + \beta_1\sigma_{t-1}^2 + \cdots + \beta_p\sigma_{t-p}^2$$

$$= \alpha_0 + \sum_{i=1}^{q}\alpha_i\varepsilon_{t-i}^2 + \sum_{i=1}^{p}\beta_i\sigma_{t-i}^2$$

Para el cálculo del modelo EWMA utilizado en este trabajo, comenzamos calculando los retornos diarios logarítmicos de cada instrumento

$$u_i = \ln\left(\frac{S_i}{S_{i-1}}\right)\,, donde\ \mathrm{S}_i\ es\ el\ precio\ spot\ de\ la\ accion\ a\ tiempo\ i$$

$$y\ u_i\ es\ el\ retorno\ logaritmico\ en\ el\ intervalo\ (i-1, i)$$

Sabemos que la varianza σ^2 de un tiempo *t* está definida por:

$$\sigma_t^2 = \frac{1}{N-1}\sum_{i=1}^{N}(u_{t-i} - \bar{u})^2$$

$$donde$$

$$N = N\acute{u}mero\ de\ observaciones$$

$$\bar{u} = \frac{1}{N}\sum_{i=1}^{N} u_{t-i}$$

Como estamos trabajando con precios de acciones, podemos asumir que $\bar{u}$ tiende a cero (con N tendiendo a infinito) y podemos reemplazar $\frac{1}{N-1}$ por $\frac{1}{N}$, llegando a la siguiente ecuación de varianza σ_t^2:

$$\sigma_t^2 = \frac{\sum_{i=1}^{N}(u_{t-i})^2}{N}$$

Como el objetivo es estimar la volatilidad actual, queremos darles más peso a las observaciones más recientes, por lo que la volatilidad podría definirse como:

$$\sigma_t^2 = \sum_{i=1}^{N} \alpha_i\,(u_{t-i})^2$$

$$donde\ \alpha_i = \ peso\ de\ la\ observación\ del\ dia\ i$$

$$s.a \qquad \alpha_i \geq 0$$

$$s.a \sum_{i=1}^{N} \alpha_i = 1$$

El modelo EWMA es un caso particular de la ecuación anterior donde los ponderadores α_i decrecen exponencialmente a medida que se retrocede en el tiempo:

$$\alpha_{i+1} = \lambda * \alpha_i\ , 0 \leq \lambda \leq 1\ (constante)$$

Finalmente, la fórmula de la varianza σ_t^2 utilizando el modelo EWMA es:

$$\sigma_t^2 = \lambda\sigma_{t-1}^2 + (1-\lambda)u_{t-1}^2$$

Como JPMorgan encontró que un $\lambda = 0.94$ funciona para datos diarios, utilizaremos este valor para nuestros cálculos.

4.3. Estudio de evento

Las observaciones de retornos se dividieron en dos grupos: las que no se encuentran dentro de bandas de soporte/resistencia (grupo control) y las que si se encuentran dentro de estas bandas (grupo experimental). Para eliminar el sesgo de selección dentro del grupo experimental, se

eliminan de la muestra las observaciones en las cuales ocurre la identificación inicial de soporte o resistencia (por defecto, observaciones con baja volatilidad condicional asociada). Una vez hecho esto, se tomaron aleatoriamente 100 observaciones de cada grupo, obteniendo así la muestra control y experimental.

Ya que cada observación contiene su propia volatilidad EWMA (no constante), llevamos a cabo una simulación para corroborar que el método de muestreo es válido. Se estudió la probabilidad de error tipo 1 para cada instrumento y se confirmó que el método de muestreo elegido es consistente (referirse al Anexo 1 para resultados).

Luego, se construyeron las funciones de densidad de cada muestra control y experimental, tanto para volatilidades como para retornos. Para estudiar la potencial asimetría en soportes y resistencias, el análisis fue dividido en tres pruebas con tres muestras experimentales diferentes: la primera conteniendo observaciones dentro de zonas de soportes o resistencias, la segunda conteniendo observaciones solo en zonas de resistencias, y la tercera conteniendo observaciones solo en zonas de soportes.

Finalmente, mediante una prueba de Kolmogorov Smirnov, se estudia si las distribuciones de probabilidad de ambas muestras presentan evidencia significativa de ser diferentes. A su vez, mediante una prueba de diferencia de medias, buscamos evidencia de que los retornos y/o la volatilidad de ambas muestras presentan diferencias significativas.

A su vez, se llevará a cabo una prueba de sensibilidad modificando el ancho de las zonas de soporte/resistencia y se analizará, de existir, la superioridad estadística en cada caso (mayores niveles de significatividad en un caso que en otro).

5. Resultados

Como ejemplo se muestran los resultados para British Petroleum. Para resultados completos del resto de los instrumentos (grafico de precios y funciones de densidad para los tres escenarios) referirse al Anexo II.

Figura 5.1 Grafico de precios para BP con soportes y resistencias identificados

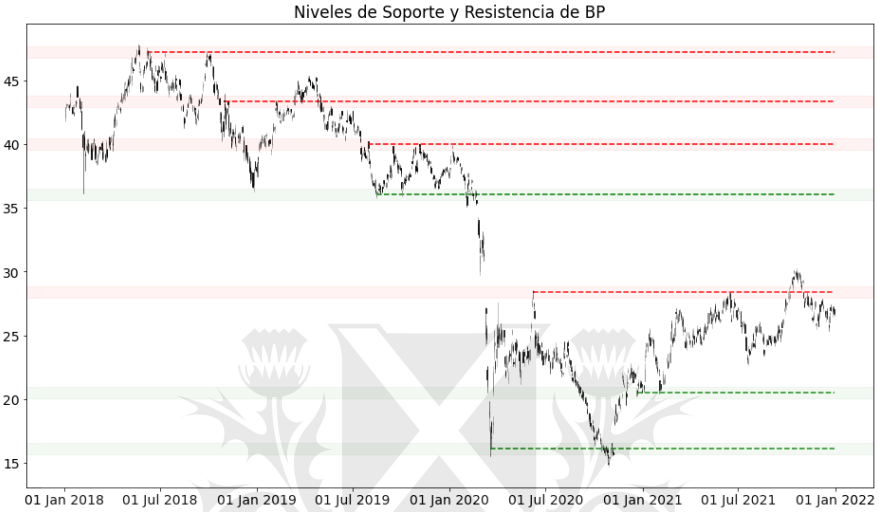
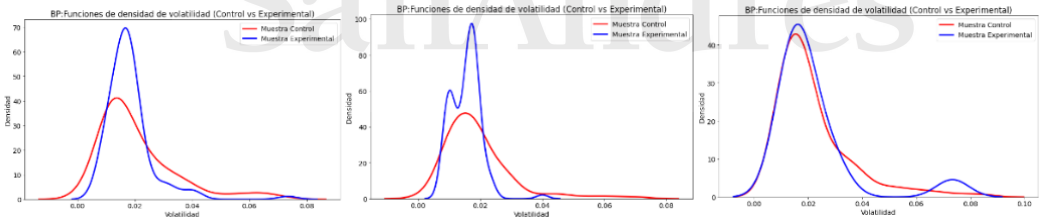


Figura 5.2 Funciones de densidad para BP

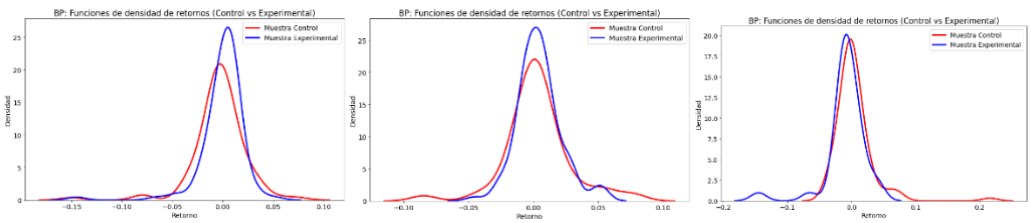
Funciones de Volatilidad:

<u>Soportes o Resistencias</u> <u>Solo Resistencias</u> <u>Solo Soportes</u>



Funciones de Retornos:

<u>Soportes o Resistencias</u> <u>Solo Resistencias</u> <u>Solo Soportes</u>



Tipo de Observaciones	Tipo de Muestra Experimental	Test Kolmogorov Smirnov	Test de diferencia de medias
Volatilidades	Soportes o Resistencias	0.004***	0.00***
	Solo Resistencias	0.006***	0.00***
	Solo Soportes	0.111	0.68
Retornos	Soportes o Resistencias	0.024**	0.18
	Solo Resistencias	1.61 e-05 ***	0.00***
	Solo Soportes	0.006***	0.03**

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Figura 5.3 Test Kolmogorov Smirnov (Estadístico)

	Volatilidades			Retornos		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	0.21**	0.37***	0.26***	0.17	0.34***	0.23***
British Petroleum	0.25***	0.17***	0.24	0.21**	0.34***	0.24***
Chevron	0.52***	0.35***	0.5***	0.12	0.34***	0.29***
Disney	0.18*	0.4***	0.3***	0.13	0.15	0.26***
Barrick Gold	0.49***	0.37***	0.16	0.1	0.19*	0.35***
Google	0.16	0.3***	0.4***	0.21**	0.47***	0.15
Johnson & Johnson	0.15	0.38***	0.19*	0.2**	0.19*	0.19*
JPMorgan	0.3***	0.49***	0.3***	0.22**	0.43***	0.17
Nike	0.14	0.39***	0.11	0.12	0.67***	0.22**
Pepsi	0.25***	0.45***	0.64***	0.24***	0.45***	0.3***
Procter & Gamble	0.29***	0.35***	0.36***	0.17	0.35***	0.12
PayPal	0.27***	0.48***	0.35***	0.15	0.25***	0.15
Starbucks	0.2**	0.45***	0.14	0.19**	0.33***	0.13
Walmart	0.25***	0.24***	0.29***	0.17	0.18*	0.12
Exxon Mobil	0.3***	0.47***	0.39***	0.19**	0.43***	0.27***

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Figura 5.4 Test de Diferencia de Medias(Estadístico)

	Volatilidades			Retornos		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	-1.68*	-3.63***	2.59***	0.68	4.9***	-3.01***
British Petroleum	3.5***	0.42***	3.42	1.34	3.69***	-2.12**
Chevron	4.88***	-0.21	4.78***	0.54	2.42**	-1.73*
Disney	-0.76	-1.54	-1.42	0.51	-1.29	-3.58***
Barrick Gold	4.3***	-0.81	2.38**	0.62	2.82***	-3.56***
Google	-2.86***	-2.37***	0.38	1.94**	4.03***	-0.44
Johnson & Johnson	-0.09	-1.72*	0.5	-1.63*	-1.18	-0.78
JPMorgan	-0.34	-3.77***	1.06	2.51***	6.7***	-2.35**
Nike	-2.58***	-2.8***	-0.93	-0.29	8.86***	-1.73*
Pepsi	-2.67***	-5.96***	1.43	2.46***	4.33***	-1.85*
Procter & Gamble	2.22**	-2.29**	3.29***	1.48	5.58***	-0.03
PayPal	3.11***	-0.87	2.78***	0.78	2.62***	-0.41
Starbucks	-0.07	-7.94***	0.41	1.15	0.87	0.89
Walmart	2.72***	2.36**	0.97	0.06	1.74*	-0.93
Exxon Mobil	4.79***	2.12**	5.06***	-1.19	3.57***	-2.1**

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Referirse al Anexo III para resultados de la prueba de sensibilidad al variar el ancho de las bandas de soporte/resistencia.

Volatilidades

Sin restringir la muestra experimental a solo un tipo de zona, es decir, muestreando observaciones indiferentemente de soportes o resistencias, en la prueba de Kolmogorov-Smirnov 9 instrumentos presentaron significatividad estadística al 1%, 2 instrumentos al 5%, 1 instrumento al 10%, y 3 instrumentos no evidenciaron diferencias en el comportamiento. No obstante, al dividir la muestra en un tipo de zona especifico (solo soportes o solo resistencias), los resultados fueron contundentes: todos los instrumentos exhiben diferencias al 1% de significatividad (en al menos uno de los dos grupos mencionados) en el comportamiento de la volatilidad de los retornos en zonas de soportes/resistencias contra las observaciones en zona de control. Analizando los gráficos de funciones de densidad, podemos observar que en la gran mayoría se evidencia una reducción de volatilidad para la muestra experimental, en línea con nuestra hipótesis inicial.

Por otro lado, para la prueba de diferencia de medias (sin restringir la muestra experimental) 2 instrumentos presentaron significatividad estadística al 1%, 1 instrumento al 5%, 1 instrumento al 10%, y 11 instrumentos no evidenciaron diferencias en el comportamiento. Al dividir la muestra en solo soportes y solo resistencias los resultaros demostraron que las volatilidades exhiben diferencia de medias en zona de resistencias, con 10 instrumentos presentando significatividad al 1%. Lo resultados para soportes, aunque mejorando el caso de indiferencia, no fueron contundentes.

Retornos

Para el caso de los retornos, los resultados de ambas pruebas son muy similares al caso de volatilidades mencionado anteriormente. En el caso en el que no diferenciamos entre soportes y resistencias, solo 1 instrumento presenta diferencias al 1% y 6 instrumentos al 5%, con el resto no presentando diferencia alguna entre las distribuciones. No obstante, al dividir la muestra entre soportes y resistencias, todos los instrumentos identifican algún tipo de significatividad en alguno de los dos casos. En especial, se exhibe una clara diferencia de distribuciones en zonas de resistencias, con 11 de los 15 instrumentos presentando significatividad al 1%. En zonas de soportes, 7 de los 15 instrumentos presentaron este nivel de significatividad. En la prueba de diferencia de medias ocurre algo muy similar, con 2 instrumentos presentando diferencia al 1% en el caso inicial, aumentando luego a 10 y 3 en los casos de solo resistencias y solo soportes respectivamente.

6. Conclusiones

Tras analizar los resultados obtenidos, un gran número de instrumentos exhiben diferencias en distribuciones de volatilidad y retornos al estar dentro de zonas de soportes/resistencias. No obstante, con la prueba de diferencia de medias no se encontraron resultados suficientes para poder establecer que la media (de volatilidad y de retorno) varia significativamente al estar dentro de zonas de soportes/resistencias que al no estarlo.

De esta manera, concluimos que nuestro estudio demostró resultados mixtos. Por un lado, queda evidenciado que cuando los precios ingresan a zonas de soporte y/o resistencia, su comportamiento cambia respecto a cuanto se encuentran fuera de estas zonas (todos los instrumentos demostraron significatividad al 1% en la prueba de volatilidades de Kolmogorov Smirnov). Al establecer que los retornos no se comportan de la misma manera en todos los niveles de precios, se puede pensar que existe la posibilidad de obtener retornos extraordinarios siguiendo una estrategia de análisis técnico por sobre el clásico *buy-and-hold*, respondiendo la pregunta de investigación y sugiriendo que los mercados no son eficientes.

Sin embargo, esta diferencia en distribuciones por si sola no valida nuestra hipótesis inicial de reducción de volatilidad en dichas zonas, ya que, al observar los gráficos de funciones de densidad, podemos identificar casos en que las distribuciones son significativamente diferentes pero la dirección es la opuesta a la esperada (ej. Amazon con muestreo experimental no restringido, donde la volatilidad es mayor en zonas de soportes/resistencias que en zonas de control).

El presente trabajo estuvo sujeto a ciertos supuestos que podrían haber influido en el resultado obtenido. En un primer lugar, esta investigación se basó en una definición de nivel de soporte y resistencia particular que incluía la permanencia de un máximo o mínimo local por cinco días y que dicho nivel fuera alcanzado con el mismo carácter dos veces. A su vez, dicho nivel luego sería reconocido como tal para toda la duración de la muestra. Es decir, los niveles una vez reconocidos no cuentan con una fecha final y cada vez que el precio se encuentre dentro de la banda, sin importar si fue quebrado o no anteriormente, se consideraran como dentro del nivel. Esto se basa en el supuesto que en un futuro dicho nivel podría volver a activarse. Sin embargo, debido a ello se podría estar incluyendo una mayor cantidad de datos en la muestra experimental de la correspondida (en caso de que los niveles queden invalidados una vez que se quiebran). Adicionalmente, se trabajó únicamente con precios de cierre, ignorando el *price action* intradiario (los máximos y mínimos de cada día, y el rechazo de estos, podrían aportar más información a la investigación).

En un segundo lugar, se determinó un ancho de banda en base al promedio de variación diaria de la muestra (Garzarelli et al. (2014)). De haber utilizado un parámetro diferente para determinar el ancho de cada nivel de soporte/resistencia, las observaciones dentro de la muestra experimental podrían haber variado significativamente, y consecuentemente, también los resultados obtenidos. Si bien se realizó un análisis de sensibilidad, el mismo fue para solo un ancho de banda elegido arbitrariamente, por lo que se podría profundizar este aspecto de la investigación.

Por último, no se utilizó ningún criterio en específico para la elección de los *tests* utilizados.

El presente trabajo posee resultados interesantes y puede ser de gran ayuda para futuras líneas de investigación. Sería interesante analizar como varían los resultados al modificar el método de identificación de soportes y resistencias, realizar un análisis de sensibilidad más detallado para confirmar que el ancho de banda elegido no condiciona los resultados, y explorar las diferentes opciones de test de hipótesis para comparar el comportamiento en la muestra experimental y

control. Por último, si se aumenta el número de instrumentos analizados se podría evaluar el comportamiento especifico de diferentes clases de activos (ej. Acciones defensivas con beta <1). Observando la similitud de comportamiento de British Petroleum, Chevron, y Exxon Mobil, creemos que se encontrarían resultados significativos al dividir el análisis por clase de activo.

7. Anexos

7.1. Anexo I: Resultados de simulación (Error Tipo 1)

Para validar el método de muestreo, se llevó a cabo una simulación con 1000 iteraciones en las que se tomaban 100 veces dos muestras aleatorias de 500 observaciones cada una (ambas del grupo control fuera de zonas de soporte o resistencia) y en cada una se hacía una prueba de Kolmogorov Smirnov para estudiar diferencias en las distribuciones. Si la prueba era significativa al 5%, se consideraba como una "falla" del modelo de muestreo, ya que se rechaza la hipótesis nula de igualdad de distribuciones cuando ambas muestras provenían del mismo grupo. Al dividir el número de "fallos" por el número de iteraciones, llegamos a la probabilidad de error tipo 1 en cada instrumento (probabilidad de rechazar H0 cuando H0 es cierta). A continuación, los resultados encontrados:

Instrumento	***Probabilidad de Error Tipo 1***
Amazon (AMZN)	4.7%
British Petroleum (BP)	4.8%
Chevron (CVX)	4.7%
Disney (DIS)	4.7%
Barrick Gold (GOLD)	4.7%
Google (GOOGL)	4.8%
Johnson & Johnson (JNJ)	4.8%
JPMorgan (JPM)	4.8%
Nike (NKE)	4.6%
Pepsi (PEP)	4.8%
Procter & Gamble (PG)	4.9%
PayPal (PYPL)	4.7%
Starbucks (SBUX)	4.8%
Walmart (WMT)	4.6%
Exxon Mobil (XOM)	4.8%

Tras esta prueba, queda en evidencia que la probabilidad de rechazar la hipótesis nula cuando la misma es verdadera es muy baja (menor a 5% para todos los instrumentos), por lo cual queda validado el modelo de muestreo propuesto para nuestro análisis.

7.2. Anexo II: Gráficos de precios y funciones de densidad

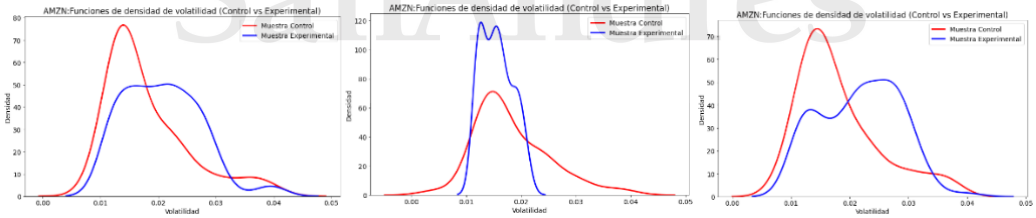
7.2.1. Amazon (AMZN)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

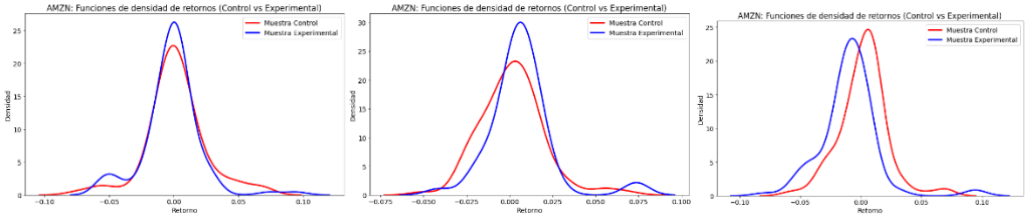
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



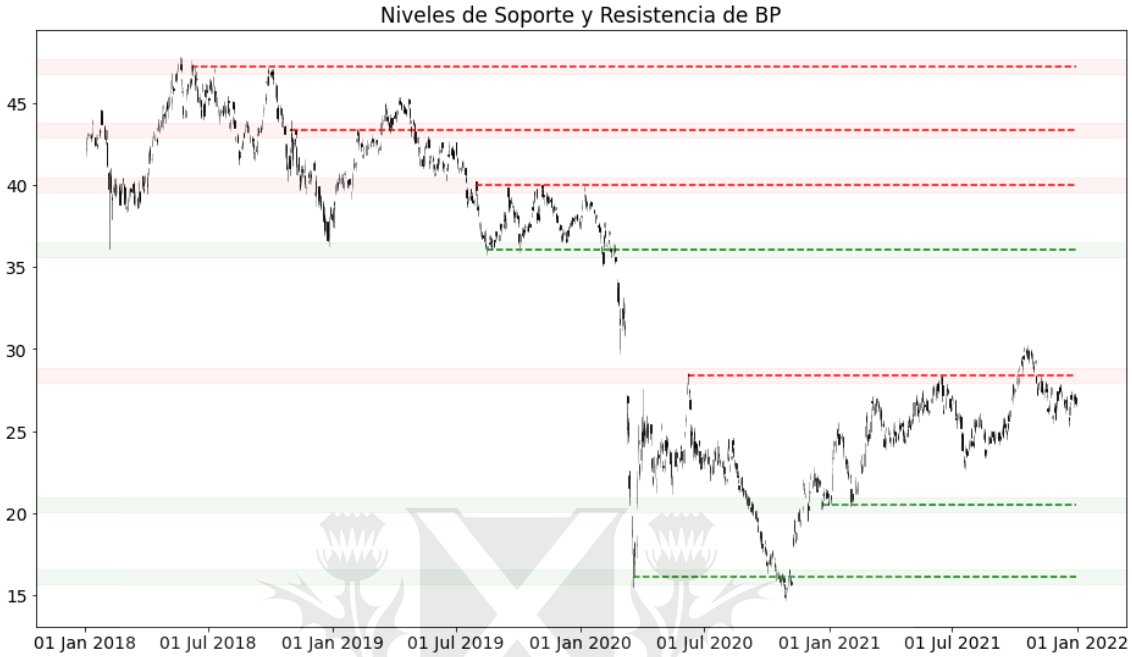
Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



7.2.2. British Petroleum (BP)

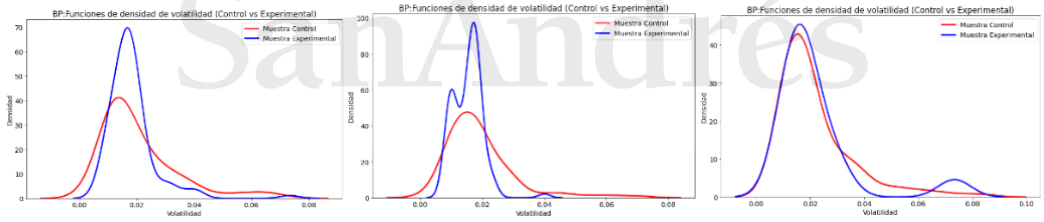
Precio con Soportes y Resistencias Identificados



Funciones de Densidad:

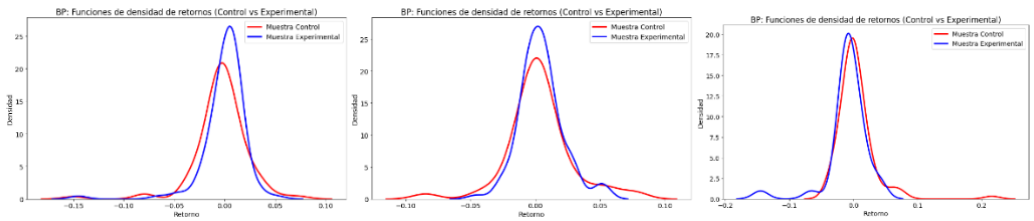
Funciones de Volatilidad:

Soportes o Resistencias Solo Resistencias Solo Soportes



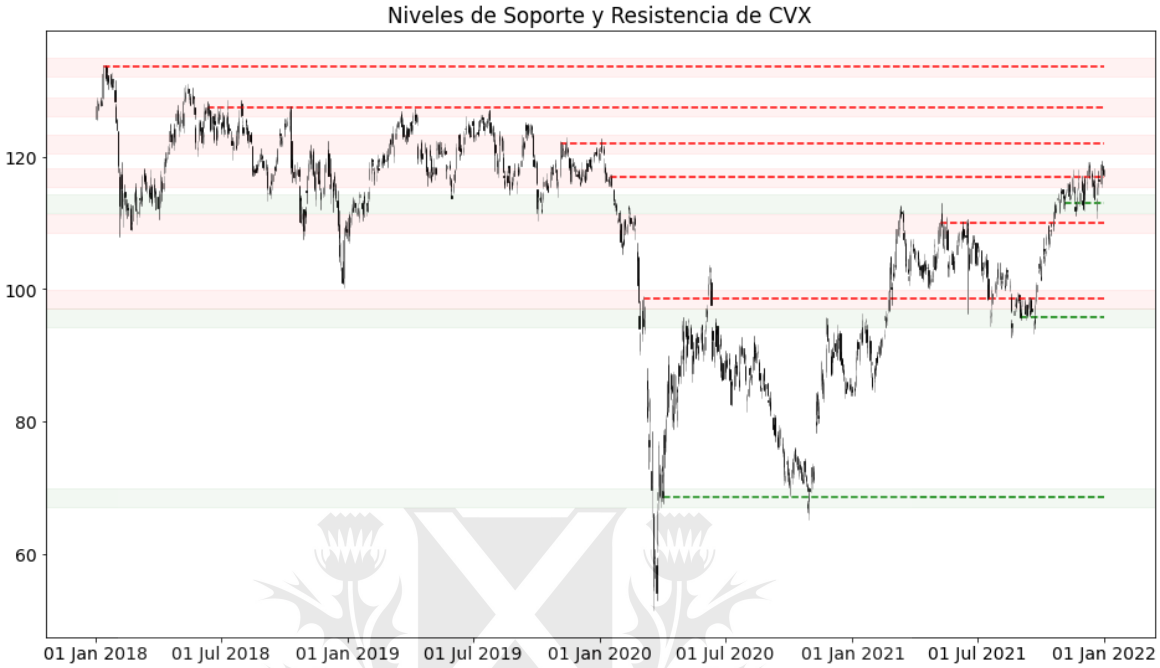
Funciones de Retornos:

Soportes o Resistencias Solo Resistencias Solo Soportes



7.2.3. Chevron (CVX)

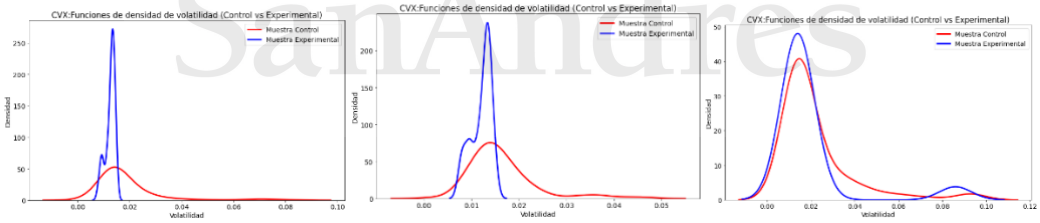
Precio con Soportes y Resistencias Identificados



Funciones de Densidad:

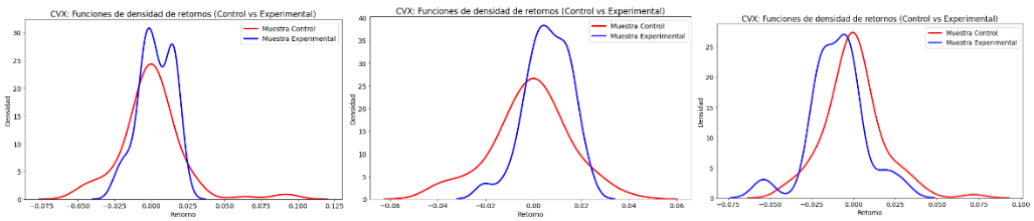
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



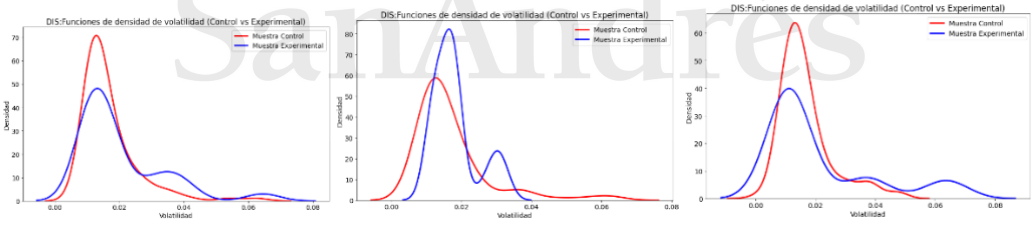
7.2.4. Disney (DIS)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

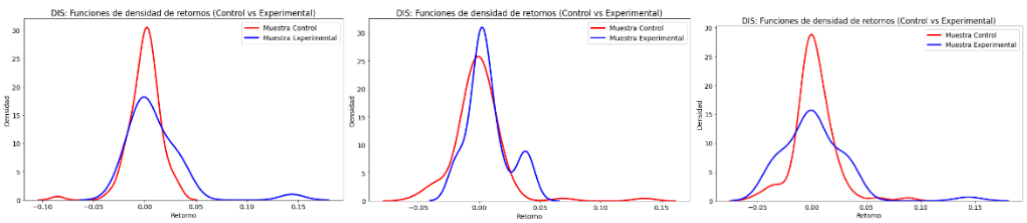
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



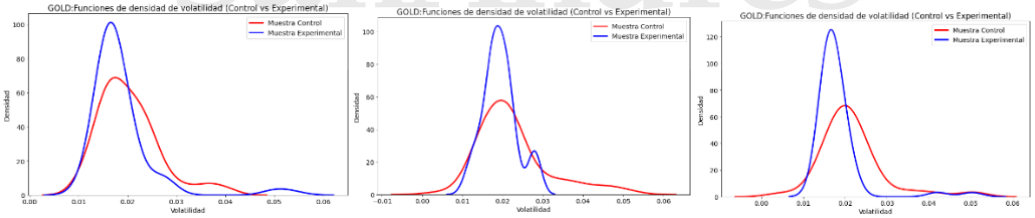
7.2.5. Barrick Gold (GOLD)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

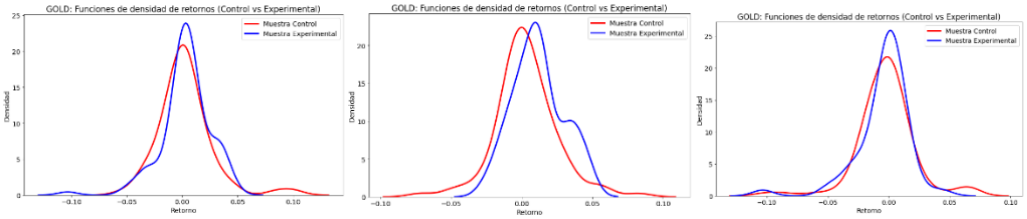
Funciones de Volatilidad:

Soportes o Resistencias Solo Resistencias Solo Soportes



Funciones de Retornos:

Soportes o Resistencias Solo Resistencias Solo Soportes



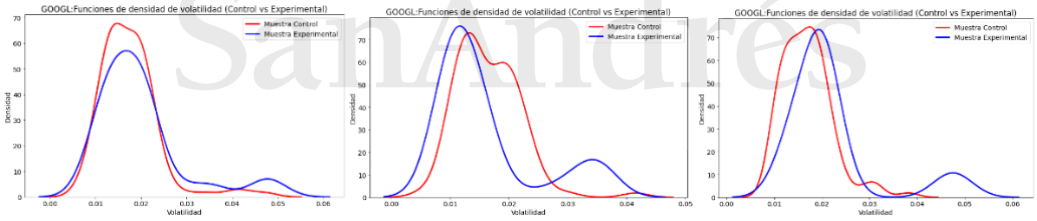
7.2.6. Google (GOOGL)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

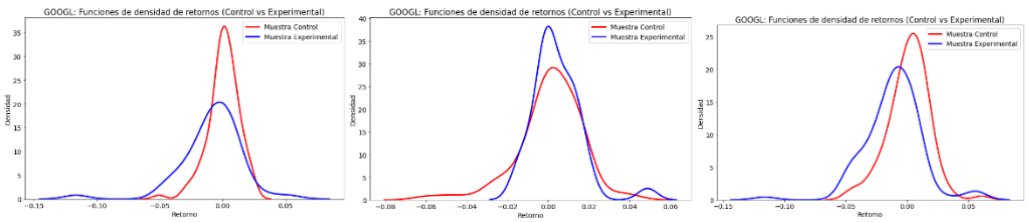
Funciones de Volatilidad:

Soportes o Resistencias Solo Resistencias Solo Soportes



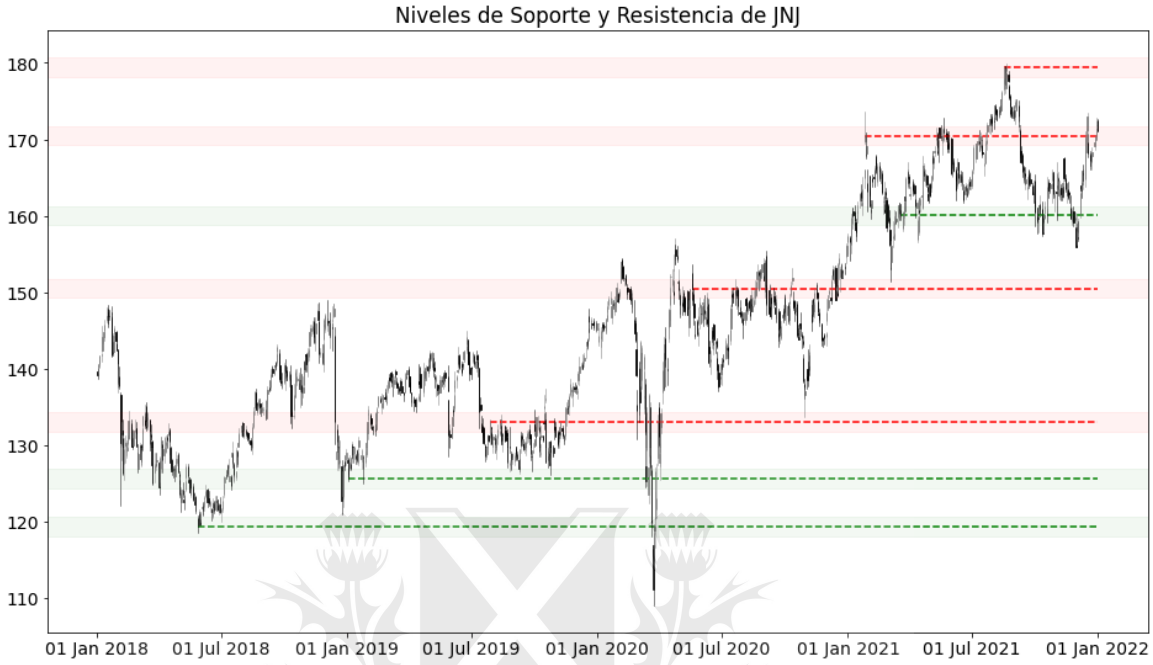
Funciones de Retornos:

Soportes o Resistencias Solo Resistencias Solo Soportes



7.2.7. Johnson & Johnson (JNJ)

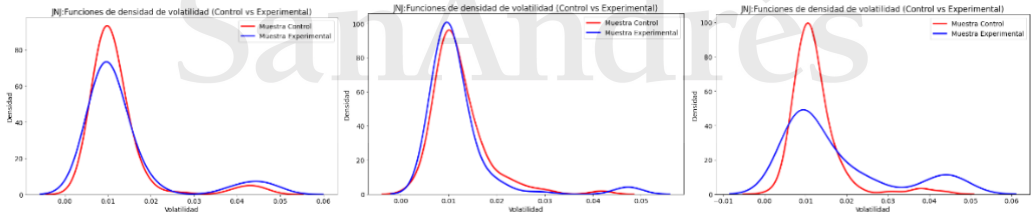
Precio con Soportes y Resistencias Identificados



Funciones de Densidad:

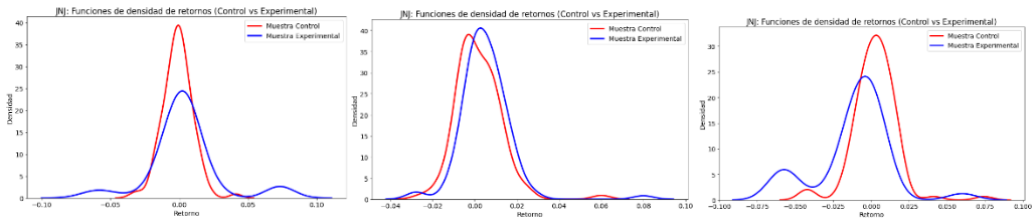
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



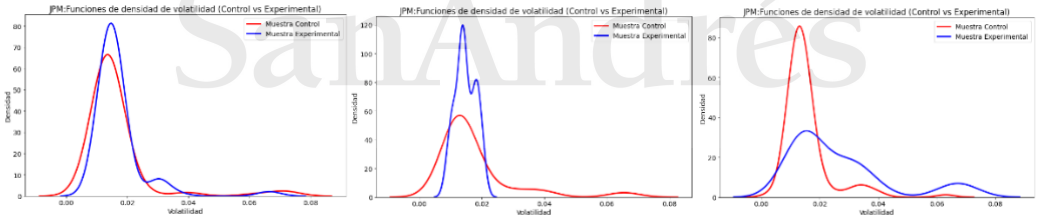
7.2.8. JPMorgan (JPM)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

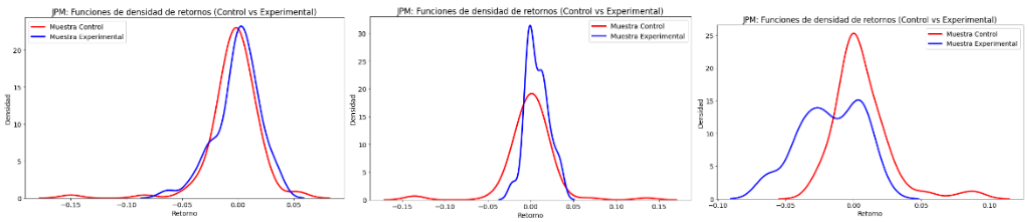
Funciones de Volatilidad:

Soportes o Resistencias Solo Resistencias Solo Soportes



Funciones de Retornos:

Soportes o Resistencias Solo Resistencias Solo Soportes



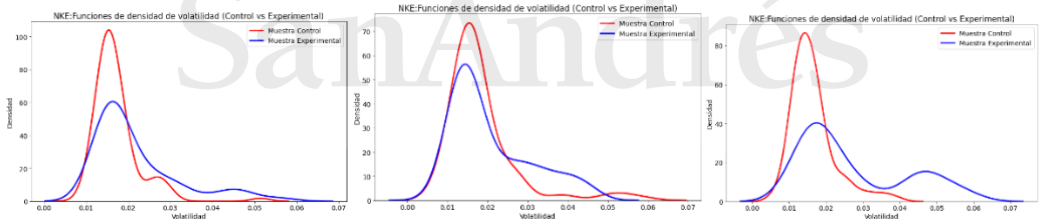
7.2.9. Nike (NKE)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

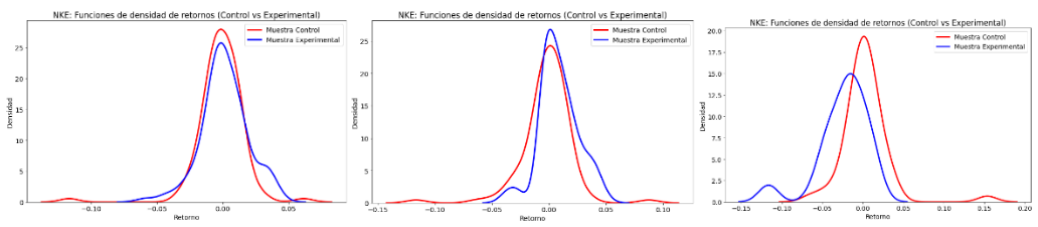
Funciones de Volatilidad:

Soportes o Resistencias Solo Resistencias Solo Soportes



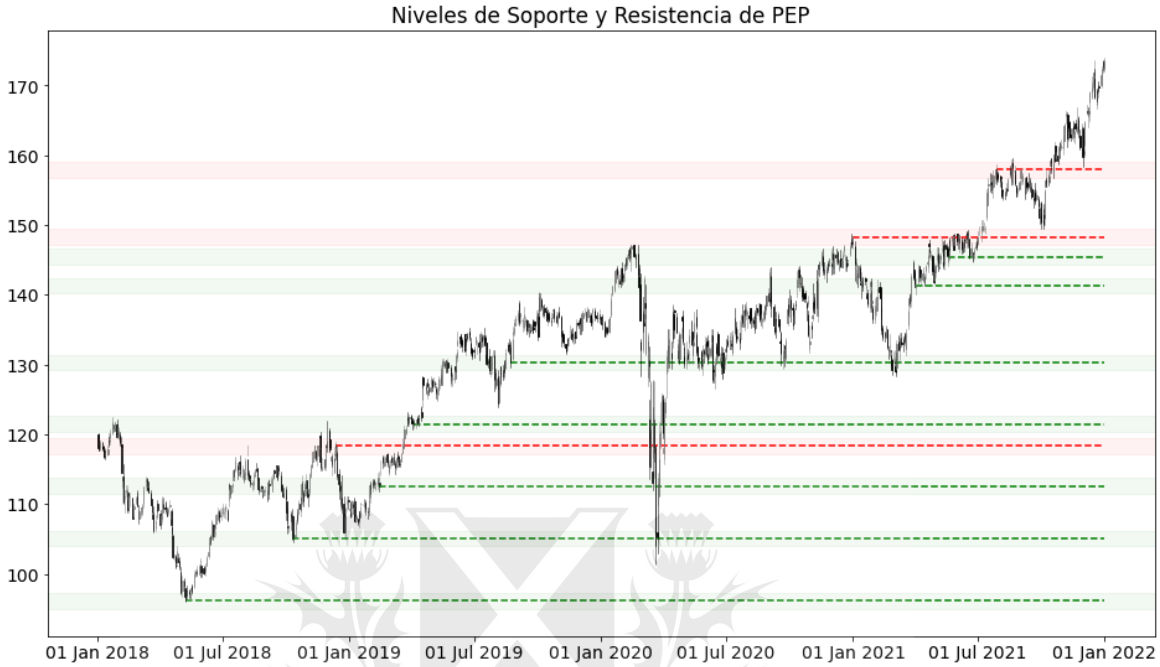
Funciones de Retornos:

Soportes o Resistencias Solo Resistencias Solo Soportes



7.2.10. Pepsi (PEP)

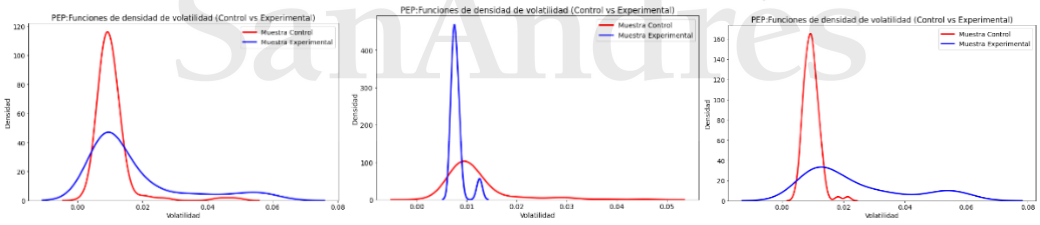
Precio con Soportes y Resistencias Identificados



Funciones de Densidad:

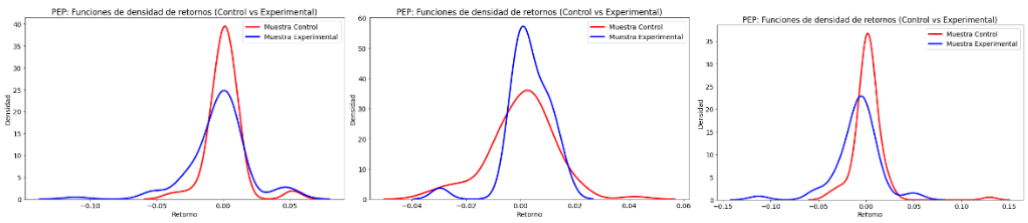
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



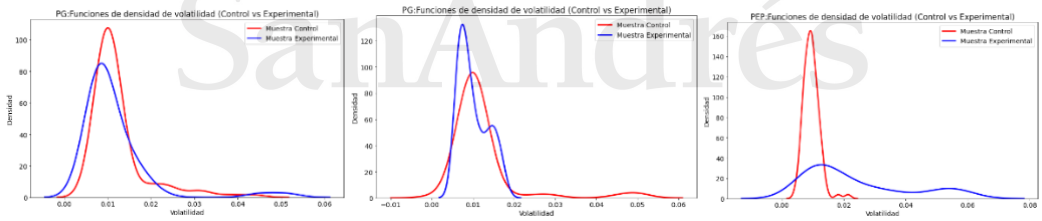
7.2.11. Procter & Gamble (PG)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

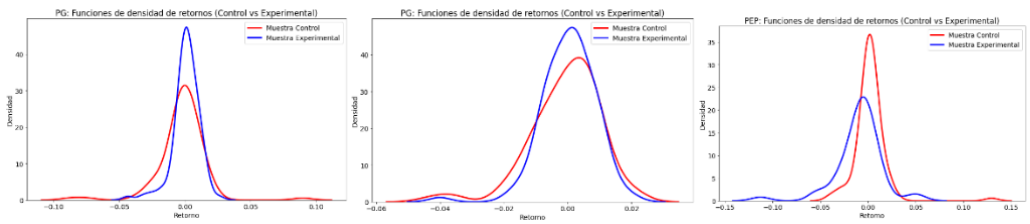
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



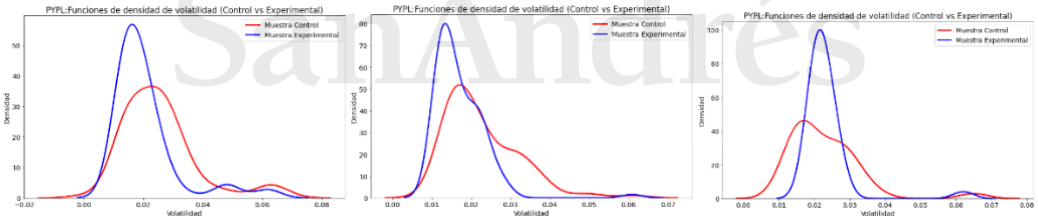
7.2.12. PayPal (PYPL)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

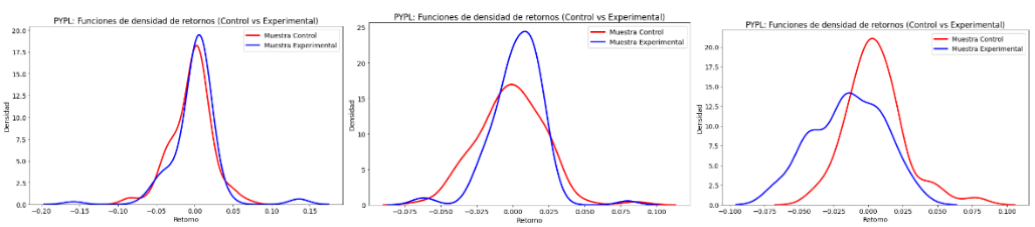
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



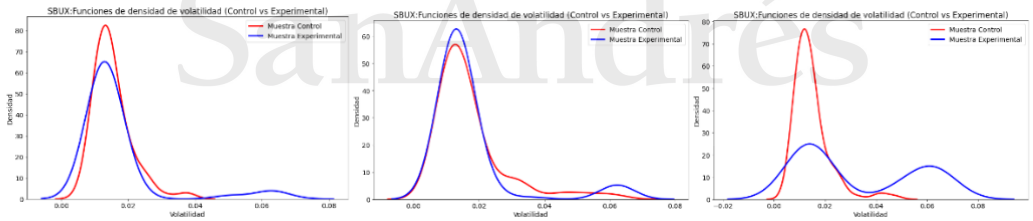
7.2.13. Starbucks (SBUX)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

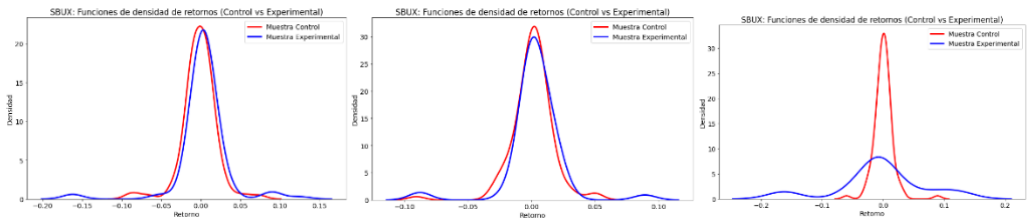
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



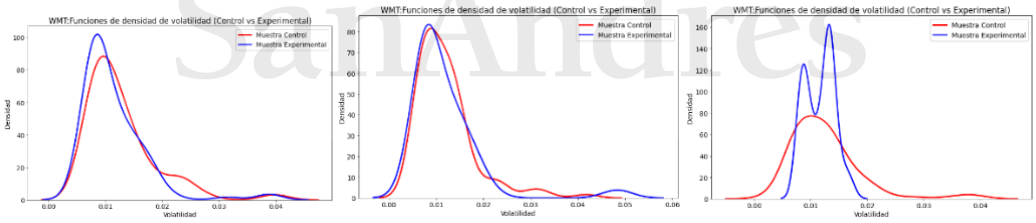
7.2.14. Walmart (WMT)

Precio con Soportes y Resistencias Identificados

Funciones de Densidad:

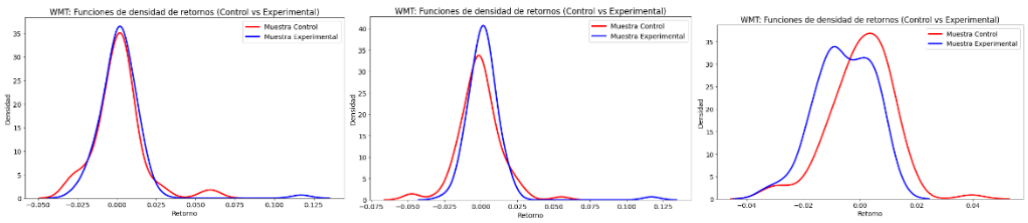
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



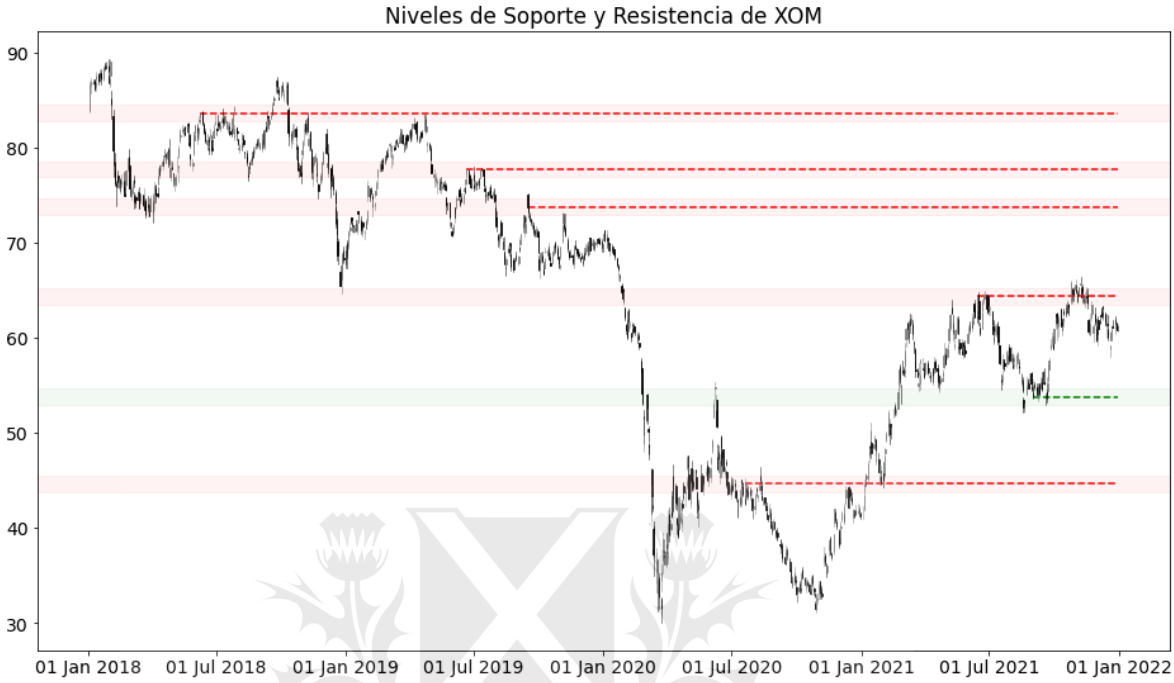
Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



7.2.15. Exxon Mobil (XOM)

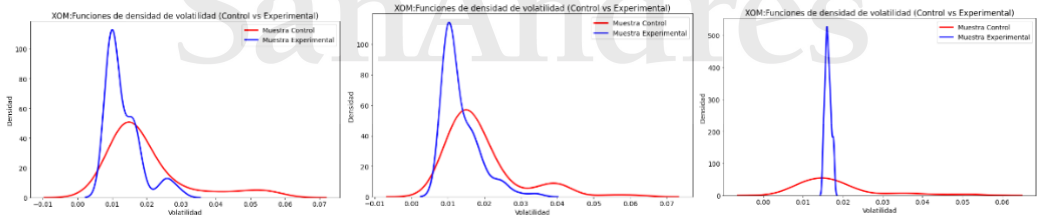
Precio con Soportes y Resistencias Identificados



Funciones de Densidad:

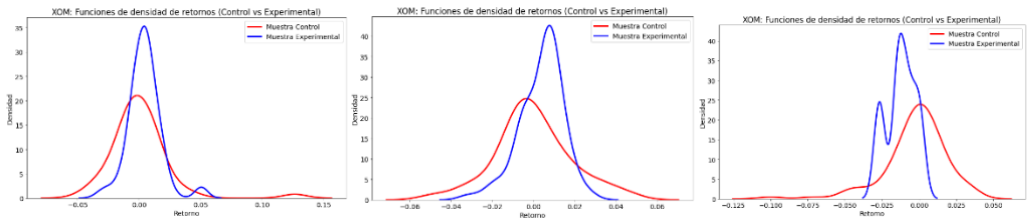
Funciones de Volatilidad:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



Funciones de Retornos:

Soportes o Resistencias | Solo Resistencias | Solo Soportes



7.3. Anexo III: Prueba de sensibilidad

A continuación, podemos encontrar los resultados tras modificar el ancho de las zonas de soporte/resistencia a ¼ del desvío estándar de cada serie de precios (comparados con el ancho de banda original). Resaltados en color verde se encuentran los casos en donde los resultados presentan mayor significatividad en un caso que en otro:

Figura 7.3.1 Test Kolmogorov Smirnov para Volatilidades (Estadístico)

	Bandas Anchas			Bandas Originales		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	0.26***	0.48***	0.51***	0.21**	0.37***	0.26***
British Petroleum	0.28***	0.39***	0.26***	0.25***	0.17***	0.24
Chevron	0.29***	0.25***	0.26***	0.52***	0.35***	0.5***
Disney	0.18*	0.35***	0.34***	0.18*	0.4***	0.3***
Barrick Gold	0.3***	0.21**	0.4***	0.49***	0.37***	0.16
Google	0.21**	0.43***	0.32***	0.16	0.3***	0.4***
Johnson & Johnson	0.21**	0.29***	0.39***	0.15	0.38***	0.19*
JPMorgan	0.16	0.24***	0.26***	0.3***	0.49***	0.3***
Nike	0.09	0.17	0.46***	0.14	0.39***	0.11
Pepsi	0.33***	0.5***	0.54***	0.25***	0.45***	0.64***
Procter & Gamble	0.15	0.11	0.53***	0.29***	0.35***	0.36***
PayPal	0.1	0.13	0.55***	0.27***	0.48***	0.35***
Starbucks	0.18*	0.24***	0.21**	0.2**	0.45***	0.14
Walmart	0.14	0.19*	0.45***	0.25***	0.24***	0.29***
Exxon Mobil	0.25***	0.29***	0.55***	0.3***	0.47***	0.39***

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Figura 7.3.2 Test Kolmogorov Smirnov para Retornos (Estadístico)

	Bandas Anchas			Bandas Originales		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	0.11	0.16	0.42***	0.17	0.34***	0.23***
British Petroleum	0.11	0.14	0.21**	0.21**	0.34***	0.24***
Chevron	0.14	0.11	0.13	0.12	0.34***	0.29***
Disney	0.17	0.14	0.24***	0.13	0.15	0.26***
Barrick Gold	0.09	0.14	0.14	0.1	0.19*	0.35***
Google	0.26***	0.16	0.11	0.21**	0.47***	0.15
Johnson & Johnson	0.16	0.21**	0.23***	0.2**	0.19*	0.19*
JPMorgan	0.21**	0.11	0.33***	0.22**	0.43***	0.17
Nike	0.09	0.17	0.18*	0.12	0.67***	0.22**
Pepsi	0.14	0.13	0.19*	0.24***	0.45***	0.3***
Procter & Gamble	0.1	0.2**	0.27***	0.17	0.35***	0.12
PayPal	0.14	0.1	0.25***	0.15	0.25***	0.15

Starbucks	0.16	0.21**	0.26***	0.19**	0.33***	0.13
Walmart	0.1	0.18*	0.27***	0.17	0.18*	0.12
Exxon Mobil	0.19*	0.2**	0.73***	0.19**	0.43***	0.27***

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

En el caso de las funciones de densidad para volatilidades, los resultados son variados, con algunos instrumentos presentando diferencias en significatividad sobre otros para ambos casos propuestos. No obstante, al analizar los resultados para las funciones de densidad de retornos, podemos observar que es evidente la superioridad del modelado de niveles de soporte y resistencia como plantea Garzarelli et al. (2014), y como se propuso en la sección 4.1.

Figura 7.3.3 Test de Diferencia de Medias Volatilidades (Estadístico)

	Bandas Anchas			Bandas Originales		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	-2.99***	4.38	-5.7***	-1.68*	-3.63***	2.59***
British Petroleum	3.43***	4.67***	0.16	3.5***	0.42***	3.42
Chevron	3.54***	2.96***	2.42**	4.88***	-0.21	4.78***
Disney	-2.92***	-0.88	-4.43***	-0.76	-1.54	-1.42
Barrick Gold	4.11***	2.87***	2.15**	4.3***	-0.81	2.38**
Google	-2.43**	1.37	-3.95***	-2.86***	-2.37***	0.38
Johnson & Johnson	0.97	4.11***	0.74	-0.09	-1.72*	0.5
JPMorgan	1.54	3.74***	-0.57	-0.34	-3.77***	1.06
Nike	0.03	-1.19	-4.15***	-2.58***	-2.8***	-0.93
Pepsi	-3.66***	1.46	-8.17***	-2.67***	-5.96***	1.43
Procter & Gamble	-0.02	1.45	-7.58***	2.22**	-2.29**	3.29***
PayPal	0.11	-1.2	-4.16***	3.11***	-0.87	2.78***
Starbucks	0.03	1.32	-3.02***	-0.07	-7.94***	0.41
Walmart	0.18	-1.22	-4.75***	2.72***	2.36**	0.97
Exxon Mobil	3.72***	3.76***	1.67***	4.79***	2.12**	5.06***

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Figura 7.3.4 Test de Diferencia de Medias Retornos (Estadístico)

	Bandas Anchas			Bandas Originales		
	Ambos	Resistencias	Soportes	Ambos	Resistencias	Soportes
Amazon	-0.27	0.19	5.46***	0.68	4.9***	-3.01***
British Petroleum	0.35	-0.29	2.55***	1.34	3.69***	-2.12**
Chevron	-0.15	0.11	0.08	0.54	2.42**	-1.73*
Disney	-2.49***	0.02	-2.16**	0.51	-1.29	-3.58***
Barrick Gold	0.26	-1.3	1.33	0.62	2.82***	-3.56***
Google	2.59***	0.46	0.16	1.94**	4.03***	-0.44
Johnson & Johnson	1.61	-3.02***	2.6***	-1.63*	-1.18	-0.78
JPMorgan	2.38**	-0.55	1.45	2.51***	6.7***	-2.35**
Nike	-0.84	0.0	1.59	-0.29	8.86***	-1.73*
Pepsi	0.39	1.48	0.67	2.46***	4.33***	-1.85*
Procter & Gamble	0.49	-0.61	2.59***	1.48	5.58***	-0.03
PayPal	0.31	0.49	2.35**	0.78	2.62***	-0.41
Starbucks	0.63	-1.07	1.75*	1.15	0.87	0.89
Walmart	-0.5	-1.46	2.64***	0.06	1.74*	-0.93
Exxon Mobil	-0.36	-1.2	7.22***	-1.19	3.57***	-2.1**

*** = Diferencia significativa al 1%

** = Diferencia significativa al 5%

* = Diferencia significativa al 10%

Para el test de diferencias de medias, podemos observar que se presenta un comportamiento similar al mencionado en el test Kolmogorov-Smirnov. Para ambos casos de estudio (funciones de densidad de volatilidad y funciones de densidad de retorno), el modelo planteado por Garzarelli et al. (2014) es el que presenta mejores resultados en términos de significatividad, por lo cual optamos por utilizarlo a lo largo de nuestro estudio.

7.4. Anexo IV: Código de Python (Caso grupo experimental con niveles de soporte y resistencia)

```python
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import yfinance as yf
import numpy as np
import math
from mplfinance.original_flavor import candlestick_ohlc
import matplotlib.dates as mpl_dates
import matplotlib.pyplot as plt
import pingouin as pg
from scipy.stats import ks_2samp
import statsmodels.api as sm
from statsmodels.graphics.gofplots import qqplot_2samples

plt.rcParams['figure.figsize'] = [12, 7]
plt.rc('font', size=14)
def get_stock_price(symbol):
  df = yf.download(symbol, start='2018-01-01', end='2022-01-01', threads= False)
  df['Date'] = pd.to_datetime(df.index)
  df['Date'] = df['Date'].apply(mpl_dates.date2num)
  df = df.loc[:,['Date', 'Open', 'High', 'Low', 'Close']]
  return df

TICKERS = ["AMZN", "BP","CVX","DIS","GOLD","GOOGL","JNJ","JPM"
           ,"NKE","PEP","PG","PYPL","SBUX","WMT", "XOM"]

for symbol in TICKERS:

    df = get_stock_price(symbol)

    def is_far_from_resistance(value, resistencia, df):
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = 1/4*np.std(df["Close"])
      return np.sum([abs(value-level)<(1*banda) for _,level in resistencia])==0

    def is_far_from_support(value, soporte, df):
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = 1/4*np.std(df["Close"])
      return np.sum([abs(value-level)<(1*banda) for _,level in soporte])==0

    def detect_level(df):
      levels = []
      soporte = []
      resistencia = []
      RESISTENCIA =[]
      SOPORTE = []
      max_list = []
      min_list = []
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = 1/4*np.std(df["Close"])
      for i in range(5, len(df)-5):
          high_range = df['Close'][i-5:i+5]
          current_max = high_range.max()
          if current_max not in max_list:
              max_list = []
          max_list.append(current_max)

          if len(max_list) == 5 :
              up = df["Close"][i-5:]-(current_max+banda)
              down = df["Close"][i-5:]-(current_max-banda)
              up_crossings = np.where(np.diff(np.signbit(up)))[0]
              down_crossings = np.where(np.diff(np.signbit(down)))[0]
              fechafinal = down.index[down ==down[down_crossings[len(down_crossings)-1]]].tolist()[-1]
              if len(up_crossings)==0:
                  if len(down_crossings)>=4 and is_far_from_resistance(current_max, resistencia, df):
                      resistencia.append((high_range.idxmax(), current_max ))
                      levels.append((high_range.idxmax(), current_max,fechafinal ))
                      RESISTENCIA.append((i-5, high_range.idxmax(), current_max,fechafinal ))
                  if len(down_crossings)<=2:
                      max_list = []

              if 2>len(up_crossings)>0 and len(down_crossings)>4:
                  if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                      if up_crossings[0]>=60:
                          resistencia.append((high_range.idxmax(), current_max ))
                          levels.append((high_range.idxmax(), current_max, fechafinal ))
                          RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))
```

```
            if 4>=len(up_crossings)>=2 and len(down_crossings)>6:
                if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                    if up_crossings[0]>=60:
                        resistencia.append((high_range.idxmax(), current_max ))
                        levels.append((high_range.idxmax(), current_max, fechafinal ))
                        RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))

            if 10>len(up_crossings)>4 and len(down_crossings)>4 and len(up_crossings)<len(down_crossings) :
                if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                    if up_crossings[0]>=60:
                        resistencia.append((high_range.idxmax(), current_max ))
                        levels.append((high_range.idxmax(), current_max ,fechafinal))
                        RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))


        low_range = df['Close'][i-5:i+5]
        current_min = low_range.min()
        if current_min not in min_list:
            min_list = []
        min_list.append(current_min)
        if len(min_list) == 5:
            up = df["Close"][i-5:]-(current_min+banda)
            down = df["Close"][i-5:]-(current_min-banda)
            up_crossings = np.where(np.diff(np.signbit(up)))[0]
            down_crossings = np.where(np.diff(np.signbit(down)))[0]
            fechafinal = up.index[up ==up[up_crossings[len(up_crossings)-1]]].tolist()[-1]
            if len(down_crossings)==0:
                    if len(up_crossings)>4 and is_far_from_support(current_min, soporte, df):
                        soporte.append((low_range.idxmin(), current_min))
                        levels.append((low_range.idxmin(), current_min, fechafinal))
                        SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))
                    if len(up_crossings)<=2:
                        max_list = []
            if 2>len(down_crossings)>0 and len(up_crossings)>=2:
                if down_crossings[0]> up_crossings[2] and is_far_from_support(current_min, soporte, df):
                    if down_crossings[0]>=240:
                        soporte.append((low_range.idxmin(), current_min))
                        levels.append((low_range.idxmin(), current_min, fechafinal))
                        SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))

            if 8>=len(down_crossings)>=2 and len(up_crossings)>3:
                if down_crossings[0]>=120 and is_far_from_support(current_min, soporte, df):
                    soporte.append((low_range.idxmin(), current_min))
                    levels.append((low_range.idxmin(), current_min, fechafinal))
                    SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))


    return SOPORTE, RESISTENCIA

def plot_all(levels, df):
    fig, ax = plt.subplots(figsize=(16, 9))
    candlestick_ohlc(ax,df.values,width=0.6, colorup='gray',
      colordown='black', alpha=0.8)
    date_format = mpl_dates.DateFormatter('%d %b %Y')
    ax.xaxis.set_major_formatter(date_format)
    banda = np.mean(abs(df['Close'] - df['Close'].shift(1)))
    #banda = 1/2*np.std(df["Close"])
    for level in levels[0]:
      plt.hlines(level[2], xmin = level[1], xmax =
        df["Date"].tail(1).item(), colors='green', linestyle='--')
      plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='green')
    for level in levels[1]:
      plt.hlines(level[2], xmin = level[1], xmax =
        df["Date"].tail(1).item(), colors='red', linestyle='--')
      plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='red')
    plt.title(str(symbol))
    fig.show()

plot_all(detect_level(df), df)

c = detect_level(df)

cantsoportes = len(c[0])
cantresistencias =len(c[1])

sop=c[0]
res=c[1]
posicionS  = []
precioS = []
posicionCS = []
posicionR = []
precioR = []
posicionCR = []

for i in range(0,len(sop)):
    x = sop[i]
    posicionS.append(x[0])
    precioS.append(x[2])

for i in range(0,len(res)):
    x = res[i]
    posicionR.append(x[0])
    precioR.append(x[2])
```

```
banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
#banda = (1/4)*np.std(df["Close"])

for i in range(0,len(sop)):
    up = df["Close"][posicionS[i]:]-(precioS[i]+banda)
    up_crossings = np.where(np.diff(np.signbit(up)))[0]
    if len(up_crossings)>1:
        posicionCS.append(up_crossings[1])

for i in range(0,len(res)):
    down = df["Close"][posicionR[i]:]-(precioR[i]-banda)
    down_crossings = np.where(np.diff(np.signbit(down)))[0]
    if len(down_crossings)>1:
        posicionCR.append(down_crossings[1])

x = detect_level(df)
precios = df["Close"]
Soporte = np.zeros(len(precios))
Resistencia = np.zeros(len(precios))

Is_in_zone = np.zeros((len(precios),1))


df = df.iloc[: , 1:]
posicionff = df.reset_index()
posicionff = posicionff.reset_index()
posicionff.Date=pd.to_datetime(posicionff.Date)

PosFFS = []
PosFFR = []

for i in range(0,len(sop)):
    fecha = np.where(posicionff.Date==pd.Timestamp(sop[i][3]))
    PosFFS.append(fecha[0][0])

for i in range(0,len(res)):
    fecha = np.where(posicionff.Date==pd.Timestamp(res[i][3]))
    PosFFR.append(fecha[0][0])

for i in range(0,len(x[0])):
    for j in range(x[0][i][0],PosFFS[i],1):
        if precios[j]> (x[0][i][2]-banda) and precios[j]< (x[0][i][2]+banda):
            Soporte[j] = 1
            Is_in_zone[j] = 1
for i in range(0,len(x[1])):
        for j in range(x[1][i][0],PosFFR[i],1):
            if (x[1][i][2]-banda)<precios[j]< (x[1][i][2]+banda):
                Resistencia[j] =1
                Is_in_zone[j] = 1

# ELIMINAMOS EL SESGO DE SELECCION

for i in range(0,len(posicionCS)):
    Soporte[posicionS[i]:(posicionCS[i]+posicionS[i])] = 0
    Is_in_zone[posicionS[i]:(posicionCS[i]+posicionS[i])] = 0

for i in range(0,len(posicionCR)):
        Resistencia[posicionR[i]:(posicionCR[i]+posicionR[i])] = 0
        Is_in_zone[posicionR[i]:(posicionCR[i]+posicionR[i])] = 0

# Modelado de Volatilidad - EWMA

Retornos = precios.pct_change()

Lambda = 0.94
Sigma_0 = 0
Variance = np.zeros((len(Retornos),1))

for i in range(0,len(Retornos)):
    if (i == 0):
        Var_0 = Sigma_0**2
    else:
        Var = Lambda*Var_0 + (1-Lambda)*Retornos[i]**2
        Variance[i] = Var
        Var_0 = Var

Sigma_t_EWMA = np.sqrt(Variance)

Fechas = precios.index.tolist()
Sigma_t_EWMA = pd.DataFrame(Sigma_t_EWMA, index = Fechas)
Is_in_zone = pd.DataFrame(Is_in_zone, index = Fechas)

plt.plot(Retornos)
plt.plot(Sigma_t_EWMA)
plt.xlabel("Fecha")
plt.ylabel("Retorno / Volatilidad EWMA")
plt.title(symbol + ": Retornos y Volatilidad EWMA")
plt.legend(["Retornos","Volatilidad EWMA"])
plt.show()
```

```
fig,ax=plt.subplots()
ax.plot(Sigma_t_EWMA)
ax.set_xlabel("Fecha")
ax.set_ylabel("Volatilidad Diaria")
ax2 = ax.twinx()
ax2.plot(df["Close"], color="black")
ax2.set_ylabel("Precio")
plt.title(symbol +": Precios y Volatilidad EWMA")
plt.show()

Retornos = pd.DataFrame(Retornos)

Datos = pd.concat([Retornos,Sigma_t_EWMA,Is_in_zone], axis = 1, ignore_index= True)
Datos.columns = ['Retorno','Volatilidad', 'Esta en zona']


control_Vol = Datos.Volatilidad[Datos['Esta en zona']==0]
experimental_Vol = Datos.Volatilidad[Datos['Esta en zona']==1]

control_Ret = Datos.Retorno[Datos['Esta en zona']==0]
experimental_Ret = Datos.Retorno[Datos['Esta en zona']==1]

Muestra_Aleatoria_Control_Vol = control_Vol.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Vol = experimental_Vol.sample(100, replace=True)

Muestra_Aleatoria_Control_Ret = control_Ret.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Ret = experimental_Ret.sample(100, replace=True)


import seaborn as sns

sns.distplot(Muestra_Aleatoria_Control_Vol, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Vol, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="b")
plt.ylabel("Densidad")
plt.xlabel("Volatilidad")
plt.title(symbol + ":Funciones de densidad de volatilidad (Control vs Experimental)")
plt.legend(["Muestra Control", "Muestra Experimental"])
plt.show()

import statistics
statistics.mean(Muestra_Aleatoria_Control_Vol)
statistics.mean(Muestra_Aleatoria_Experimental_Vol)

sns.distplot(Muestra_Aleatoria_Control_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="b")
plt.ylabel("Densidad")
plt.xlabel("Retorno")
plt.title(symbol +": Funciones de densidad de retornos (Control vs Experimental)")
plt.legend(["Muestra Control", "Muestra Experimental"])
plt.show()

Vol_Results = np.zeros(100)
Ret_Results = np.zeros(100)

for j in range(0,100):

    Vol_Mistakes = 0
    Ret_Mistakes = 0

    for i in range(0,1000):

            Vol1 = control_Vol.sample(500, replace=True)
            Vol2 = control_Vol.sample(500, replace=True)
            Ret1 = control_Ret.sample(500,replace=True)
            Ret2 = control_Ret.sample(500,replace=True)

            Vol_pvalue = ks_2samp(Vol1, Vol2)[1]
            Ret_pvalue = ks_2samp(Ret1, Ret2)[1]

            if Vol_pvalue < 0.05:
                Vol_Mistakes = Vol_Mistakes + 1

            if Ret_pvalue < 0.05:
                Ret_Mistakes = Ret_Mistakes + 1

    Vol_Results[j] = Vol_Mistakes
    Ret_Results[j]=Ret_Mistakes

Prob_ET1_Vol = statistics.mean(Vol_Results)/1000
Prob_ET1_Ret = statistics.mean(Ret_Results)/1000

print(symbol)
print("Error tipo 1 Volatilidad")
print(Prob_ET1_Vol)
print("Error tipo 1 retorno ")
print(Prob_ET1_Ret)
```

```
    # Test de Kolmogorov-Smirnov
    # H0: Ambas distribuciones son iguales
    # HA: Distribuciones son distintas

    ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret)
    ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol)
    print(ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol))
    print("TABLA 1 - NECESITO EL PVALOR y t - RETORNO")
    print(ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret))

    pp_x = sm.ProbPlot(Muestra_Aleatoria_Control_Ret)
    pp_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Ret)

    qqplot_2samples(pp_x,pp_y,xlabel = "Cuantiles de retornos Control"
                    , ylabel = "Cuantiles de retornos Experimentales", line = "45")
    plt.show()

    pp2_x = sm.ProbPlot(Muestra_Aleatoria_Control_Vol)
    pp2_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Vol)

    qqplot_2samples(pp2_x,pp2_y, line = "45", xlabel = "Cuantiles de volatilidades Control"
                    , ylabel = "Cuantiles de volatilidades Experimentales")
    plt.show()

    # Test t de diferencia de medias

    pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
             , correction = False).round(2)
    pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
             , correction = False).round(2)

    print(pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
                   , correction = False).round(2))
    print(pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
                   , correction = False).round(2))

    experimental_Ret.std()
    control_Ret.std()
```

7.5. Anexo IV: Código de Python (Caso grupo experimental con solo niveles de soporte

```
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import yfinance as yf
import numpy as np
import math
from mplfinance.original_flavor import candlestick_ohlc
import matplotlib.dates as mpl_dates
import matplotlib.pyplot as plt
import pingouin as pg
from scipy.stats import ks_2samp
import statsmodels.api as sm
from statsmodels.graphics.gofplots import qqplot_2samples

plt.rcParams['figure.figsize'] = [12, 7]
plt.rc('font', size=14)


def get_stock_price(symbol):
  df = yf.download(symbol, start='2018-01-01', end='2022-01-01', threads= False)
  df['Date'] = pd.to_datetime(df.index)
  df['Date'] = df['Date'].apply(mpl_dates.date2num)
  df = df.loc[:,['Date', 'Open', 'High', 'Low', 'Close']]
  return df

TICKERS = ["AMZN", "BP","CVX","DIS","GOLD","GOOGL","JNJ","JPM","NKE","PEP","PG"
           ,"PYPL","SBUX","WMT", "XOM"]

for symbol in TICKERS:
    df = get_stock_price(symbol)
    def is_far_from_resistance(value, resistencia, df):
          banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
          #banda = (1/4)*np.std(df["Close"])
          return np.sum([abs(value-level)<(1*banda) for _,level in resistencia])==0

    def is_far_from_support(value, soporte, df):
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = (1/4)*np.std(df["Close"])
      return np.sum([abs(value-level)<(1*banda) for _,level in soporte])==0
```

```python
def detect_level(df):
  levels = []
  soporte = []
  resistencia = []
  RESISTENCIA =[]
  SOPORTE = []
  max_list = []
  min_list = []
  banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
  for i in range(5, len(df)-5):
      high_range = df['Close'][i-5:i+5]
      current_max = high_range.max()
      if current_max not in max_list:
          max_list = []
      max_list.append(current_max)
      if len(max_list) == 5 :
          up = df["Close"][i-5:]-(current_max+banda)
          down = df["Close"][i-5:]-(current_max-banda)
          up_crossings = np.where(np.diff(np.signbit(up)))[0]
          down_crossings = np.where(np.diff(np.signbit(down)))[0]
          fechafinal = down.index[down ==down[down_crossings[len(down_crossings)-1]]].tolist()[-1]
          if len(up_crossings)==0:
              if len(down_crossings)>=4 and is_far_from_resistance(current_max, resistencia, df):
                  resistencia.append((high_range.idxmax(), current_max ))
                  levels.append((high_range.idxmax(), current_max,fechafinal ))
                  RESISTENCIA.append((i-5, high_range.idxmax(), current_max,fechafinal ))
              if len(down_crossings)<=2:
                  max_list = []
          if 2>len(up_crossings)>0 and len(down_crossings)>4:
              if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                  if up_crossings[0]>=60:
                      resistencia.append((high_range.idxmax(), current_max ))
                      levels.append((high_range.idxmax(), current_max, fechafinal ))
                      RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))
          if 4>=len(up_crossings)>=2 and len(down_crossings)>6:
              if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                  if up_crossings[0]>=60:
                      resistencia.append((high_range.idxmax(), current_max ))
                      levels.append((high_range.idxmax(), current_max, fechafinal ))
                      RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))

          if 10>len(up_crossings)>4 and len(down_crossings)>4 and len(up_crossings)<len(down_crossings) :
              if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                  if up_crossings[0]>=60:
                      resistencia.append((high_range.idxmax(), current_max ))
                      levels.append((high_range.idxmax(), current_max ,fechafinal))
                      RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))
      low_range = df['Close'][i-5:i+5]
      current_min = low_range.min()
      if current_min not in min_list:
          min_list = []
      min_list.append(current_min)
      if len(min_list) == 5:
          up = df["Close"][i-5:]-(current_min+banda)
          down = df["Close"][i-5:]-(current_min-banda)
          up_crossings = np.where(np.diff(np.signbit(up)))[0]
          down_crossings = np.where(np.diff(np.signbit(down)))[0]
          fechafinal = up.index[up ==up[up_crossings[len(up_crossings)-1]]].tolist()[-1]
          if len(down_crossings)==0:
                if len(up_crossings)>4 and is_far_from_support(current_min, soporte, df):
                    soporte.append((low_range.idxmin(), current_min))
                    levels.append((low_range.idxmin(), current_min, fechafinal))
                    SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))
                if len(up_crossings)<=2:
                    max_list = []
          if 2>len(down_crossings)>0 and len(up_crossings)>=2:
              if down_crossings[0]> up_crossings[2] and is_far_from_support(current_min, soporte, df):
                  if down_crossings[0]>=240:
                      soporte.append((low_range.idxmin(), current_min))
                      levels.append((low_range.idxmin(), current_min, fechafinal))
                      SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))

          if 8>=len(down_crossings)>=2 and len(up_crossings)>3:
              if down_crossings[0]>=120 and is_far_from_support(current_min, soporte, df):
                  soporte.append((low_range.idxmin(), current_min))
                  levels.append((low_range.idxmin(), current_min, fechafinal))
                  SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))


  return SOPORTE, RESISTENCIA
```

```
def plot_all(levels, df):
  fig, ax = plt.subplots(figsize=(16, 9))
  candlestick_ohlc(ax,df.values,width=0.6, colorup='gray',
    colordown='black', alpha=0.8)
  date_format = mpl_dates.DateFormatter('%d %b %Y')
  ax.xaxis.set_major_formatter(date_format)
  banda = np.mean(abs(df['Close'] - df['Close'].shift(1)))
  for level in levels[0]:
    plt.hlines(level[2], xmin = level[1], xmax =
      df["Date"].tail(1).item(), colors='green', linestyle='--')
    plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='green')
  for level in levels[1]:
    plt.hlines(level[2], xmin = level[1], xmax =
      df["Date"].tail(1).item(), colors='red', linestyle='--')
    plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='red')
  plt.title(str(symbol))
  fig.show()

plot_all(detect_level(df), df)


c = detect_level(df)

cantsoportes = len(c[0])
cantresistencias =len(c[1])

sop=c[0]
res=c[1]
posicionS  = []
precioS = []
posicionCS = []
posicionR = []
precioR = []
posicionCR = []
# Soportes
for i in range(0,len(sop)):
    x = sop[i]
    posicionS.append(x[0])
    precioS.append(x[2])

for i in range(0,len(res)):
    x = res[i]
    posicionR.append(x[0])
    precioR.append(x[2])

banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
for i in range(0,len(sop)):
    up = df["Close"][posicionS[i]:]-(precioS[i]+banda)
    up_crossings = np.where(np.diff(np.signbit(up)))[0]
    posicionCS.append(up_crossings[1])

x = detect_level(df)
banda = np.mean(abs(df['Close'] - df['Close'].shift(1)))
precios = df["Close"]
Soporte = np.zeros(len(precios))
Resistencia = np.zeros(len(precios))

Is_in_zone = np.zeros((len(precios),1))

df = df.iloc[: , 1:]
posicionff = df.reset_index()
posicionff = posicionff.reset_index()
posicionff.Date=pd.to_datetime(posicionff.Date)

PosFFS = []
PosFFR = []

for i in range(0,len(sop)):
    fecha = np.where(posicionff.Date==pd.Timestamp(sop[i][3]))
    PosFFS.append(fecha[0][0])

for i in range(0,len(x[0])):
    for j in range(x[0][i][0],PosFFS[i],1):
        if precios[j]> (x[0][i][2]-banda) and precios[j]< (x[0][i][2]+banda):
            Soporte[j] = 1
            Is_in_zone[j] = 1


for i in range(0,len(sop)):
    Soporte[posicionS[i]:(posicionCS[i]+posicionS[i])] = 0
    Is_in_zone[posicionS[i]:(posicionCS[i]+posicionS[i])] = 0

# Modelado de Volatilidad - EWMA

Retornos = precios.pct_change()

Lambda = 0.94
Sigma_0 = 0
Variance = np.zeros((len(Retornos),1))
```

```
for i in range(0,len(Retornos)):
    if (i == 0):
        Var_0 = Sigma_0**2
    else:
        Var = Lambda*Var_0 + (1-Lambda)*Retornos[i]**2
        Variance[i] = Var
        Var_0 = Var

Sigma_t_EWMA = np.sqrt(Variance)

Fechas = precios.index.tolist()
Sigma_t_EWMA = pd.DataFrame(Sigma_t_EWMA, index = Fechas)
Is_in_zone = pd.DataFrame(Is_in_zone, index = Fechas)

plt.plot(Retornos)
plt.plot(Sigma_t_EWMA)
plt.xlabel("Fecha")
plt.ylabel("Retorno / Volatilidad EWMA")
plt.title(symbol + ": Retornos y Volatilidad EWMA")
plt.legend(["Retornos","Volatilidad EWMA"])
plt.show()


fig,ax=plt.subplots()
ax.plot(Sigma_t_EWMA)
ax.set_xlabel("Fecha")
ax.set_ylabel("Volatilidad Diaria")
ax2 = ax.twinx()
ax2.plot(df["Close"], color="black")
ax2.set_ylabel("Precio")
plt.title(symbol +": Precios y Volatilidad EWMA")
plt.show()

Retornos = pd.DataFrame(Retornos)

Datos = pd.concat([Retornos,Sigma_t_EWMA,Is_in_zone], axis = 1, ignore_index= True)
Datos.columns = ['Retorno','Volatilidad', 'Esta en zona']

control_Vol = Datos.Volatilidad[Datos['Esta en zona']==0]
experimental_Vol = Datos.Volatilidad[Datos['Esta en zona']==1]

control_Ret = Datos.Retorno[Datos['Esta en zona']==0]
experimental_Ret = Datos.Retorno[Datos['Esta en zona']==1]

Muestra_Aleatoria_Control_Vol = control_Vol.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Vol = experimental_Vol.sample(100, replace=True)
Muestra_Aleatoria_Control_Ret = control_Ret.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Ret = experimental_Ret.sample(100, replace=True)


import seaborn as sns

sns.distplot(Muestra_Aleatoria_Control_Vol, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Vol, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="b")
plt.ylabel("Densidad")
plt.xlabel("Volatilidad")
plt.title(symbol + ":Funciones de densidad de volatilidad (Control vs Experimental)")
plt.legend(["Muestra Control", "Muestra Experimental"])
plt.show()

import statistics
statistics.mean(Muestra_Aleatoria_Control_Vol)
statistics.mean(Muestra_Aleatoria_Experimental_Vol)

sns.distplot(Muestra_Aleatoria_Control_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="b")
plt.ylabel("Densidad")
plt.xlabel("Retorno")
plt.title(symbol +": Funciones de densidad de retornos (Control vs Experimental)")
plt.legend(["Muestra Control", "Muestra Experimental"])
plt.show()


Vol_Results = np.zeros(100)
Ret_Results = np.zeros(100)
for j in range(0,100):

    Vol_Mistakes = 0
    Ret_Mistakes = 0

    for i in range(0,1000):

            Vol1 = control_Vol.sample(500, replace=True)
            Vol2 = control_Vol.sample(500, replace=True)
            Ret1 = control_Ret.sample(500,replace=True)
            Ret2 = control_Ret.sample(500,replace=True)
```

```
            Vol_pvalue = ks_2samp(Vol1, Vol2)[1]
            Ret_pvalue = ks_2samp(Ret1, Ret2)[1]

            if Vol_pvalue < 0.05:
                Vol_Mistakes = Vol_Mistakes + 1

            if Ret_pvalue < 0.05:
                Ret_Mistakes = Ret_Mistakes + 1

        Vol_Results[j] = Vol_Mistakes
        Ret_Results[j]=Ret_Mistakes

    Prob_ET1_Vol = statistics.mean(Vol_Results)/1000
    Prob_ET1_Ret = statistics.mean(Ret_Results)/1000
    print(symbol)
    print("Error tipo 1 Volatilidad")
    print(Prob_ET1_Vol)
    print("Error tipo 1 retorno ")
    print(Prob_ET1_Ret)

    # Ahora hago test de Kolmogorov-Smirnov
    # H0: Ambas distribuciones son iguales
    # HA: Distribuciones son distintas

    ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret)
    ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol)

    print(symbol)
    print("TABLA 1 - NECESITO EL PVALOR y t- VOLATILIDAD")
    print(ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol))
    print("TABLA 1 - NECESITO EL PVALOR y t - RETORNO")
    print(ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret))


    pp_x = sm.ProbPlot(Muestra_Aleatoria_Control_Ret)
    pp_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Ret)

    qqplot_2samples(pp_x,pp_y,xlabel = "Cuantiles de retornos Control"
                    , ylabel = "Cuantiles de retornos Experimentales", line = "45")
    plt.show()

    pp2_x = sm.ProbPlot(Muestra_Aleatoria_Control_Vol)
    pp2_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Vol)

    qqplot_2samples(pp2_x,pp2_y, line = "45", xlabel = "Cuantiles de volatilidades Control"
                    , ylabel = "Cuantiles de volatilidades Experimentales")
    plt.show()
    # Test t de diferencia de medias

    pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
             , correction = False).round(2)
    pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
             , correction = False).round(2)

    print(pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
                   , correction = False).round(2))
    print(pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
                   , correction = False).round(2))

    experimental_Ret.std()
    control_Ret.std()
```

7.6. Anexo IV: Código de Python (Caso grupo experimental con solo niveles de resistencia)

```
import pandas as pd
import yfinance as yf
import numpy as np
import math
from mplfinance.original_flavor import candlestick_ohlc
import matplotlib.dates as mpl_dates
import matplotlib.pyplot as plt
import pingouin as pg
from scipy.stats import ks_2samp
import statsmodels.api as sm
from statsmodels.graphics.gofplots import qqplot_2samples

plt.rcParams['figure.figsize'] = [12, 7]
plt.rc('font', size=14)
```

```
def get_stock_price(symbol):
  df = yf.download(symbol, start='2018-01-01', end='2022-01-01', threads= False)
  df['Date'] = pd.to_datetime(df.index)
  df['Date'] = df['Date'].apply(mpl_dates.date2num)
  df = df.loc[:,['Date', 'Open', 'High', 'Low', 'Close']]
  return df

TICKERS = ["AMZN", "BP","CVX","DIS","GOLD","GOOGL","JNJ","JPM","NKE","PEP","PG"
           ,"PYPL","SBUX","WMT", "XOM"]

for symbol in TICKERS:
    df = get_stock_price(symbol)

    def is_far_from_resistance(value, resistencia, df):
          banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
          #banda = 1/2*np.std(df["Close"])
          return np.sum([abs(value-level)<(2.3*banda) for _,level in resistencia])==0

    def is_far_from_support(value, soporte, df):
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = 1/4*np.std(df["Close"])
      return np.sum([abs(value-level)<(2.3*banda) for _,level in soporte])==0

    def detect_level(df):
      levels = []
      soporte = []
      resistencia = []
      RESISTENCIA =[]
      SOPORTE = []
      max_list = []
      min_list = []
      banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))
      #banda = 1/4*np.std(df["Close"])
      for i in range(5, len(df)-5):
          high_range = df['Close'][i-5:i+5]
          current_max = high_range.max()
          if current_max not in max_list:
              max_list = []
          max_list.append(current_max)
          if len(max_list) == 5 :
              up = df["Close"][i-5:]-(current_max+banda)
              down = df["Close"][i-5:]-(current_max-banda)
              up_crossings = np.where(np.diff(np.signbit(up)))[0]
              down_crossings = np.where(np.diff(np.signbit(down)))[0]
              fechafinal = down.index[down ==down[down_crossings[len(down_crossings)-1]]].tolist()[-1]

              if len(up_crossings)==0:
                  if len(down_crossings)>=4 and is_far_from_resistance(current_max, resistencia, df):
                      resistencia.append((high_range.idxmax(), current_max ))
                      levels.append((high_range.idxmax(), current_max,fechafinal ))
                      RESISTENCIA.append((i-5, high_range.idxmax(), current_max,fechafinal ))
                  if len(down_crossings)<=2:
                      max_list = []
              if 2>len(up_crossings)>0 and len(down_crossings)>4:
                  if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                      if up_crossings[0]>=60:
                          resistencia.append((high_range.idxmax(), current_max ))
                          levels.append((high_range.idxmax(), current_max, fechafinal ))
                          RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))
              if 4>=len(up_crossings)>=2 and len(down_crossings)>6:
                  if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                      if up_crossings[0]>=60:
                          resistencia.append((high_range.idxmax(), current_max ))
                          levels.append((high_range.idxmax(), current_max, fechafinal ))
                          RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))

              if 10>len(up_crossings)>4 and len(down_crossings)>4 and len(up_crossings)<len(down_crossings) :
                  if up_crossings[0]> down_crossings[2] and is_far_from_resistance(current_max, resistencia, df):
                      if up_crossings[0]>=60:
                          resistencia.append((high_range.idxmax(), current_max ))
                          levels.append((high_range.idxmax(), current_max ,fechafinal))
                          RESISTENCIA.append((i-5,high_range.idxmax(), current_max,fechafinal ))


          low_range = df['Close'][i-5:i+5]
          current_min = low_range.min()
          if current_min not in min_list:
              min_list = []
          min_list.append(current_min)
          if len(min_list) == 5:
              up = df["Close"][i-5:]-(current_min+banda)
              down = df["Close"][i-5:]-(current_min-banda)
              up_crossings = np.where(np.diff(np.signbit(up)))[0]
              down_crossings = np.where(np.diff(np.signbit(down)))[0]
              fechafinal = up.index[up ==up[up_crossings[len(up_crossings)-1]]].tolist()[-1]
              if len(down_crossings)==0:
                      if len(up_crossings)>4 and is_far_from_support(current_min, soporte, df):
                          soporte.append((low_range.idxmin(), current_min))
                          levels.append((low_range.idxmin(), current_min, fechafinal))
                          SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))
                      if len(up_crossings)<=2:
                          max_list = []
```

```
            if 2>len(down_crossings)>0 and len(up_crossings)>=2:
                if down_crossings[0]> up_crossings[2] and is_far_from_support(current_min, soporte, df):
                    if down_crossings[0]>=240:
                        soporte.append((low_range.idxmin(), current_min))
                        levels.append((low_range.idxmin(), current_min, fechafinal))
                        SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))

            if 8>=len(down_crossings)>=2 and len(up_crossings)>3:
                if down_crossings[0]>=120 and is_far_from_support(current_min, soporte, df):
                    soporte.append((low_range.idxmin(), current_min))
                    levels.append((low_range.idxmin(), current_min, fechafinal))
                    SOPORTE.append((i-5,low_range.idxmin(), current_min, fechafinal))


    return SOPORTE, RESISTENCIA

def plot_all(levels, df):
    fig, ax = plt.subplots(figsize=(16, 9))
    candlestick_ohlc(ax,df.values,width=0.6, colorup='gray',
      colordown='black', alpha=0.8)
    date_format = mpl_dates.DateFormatter('%d %b %Y')
    ax.xaxis.set_major_formatter(date_format)
    banda = np.mean(abs(df['Close'] - df['Close'].shift(1)))
    #banda = 1*np.std(df["Close"])
    for level in levels[0]:
      plt.hlines(level[2], xmin = level[1], xmax =
        df["Date"].tail(1).item(), colors='green', linestyle='--')
      plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='green')
    for level in levels[1]:
      plt.hlines(level[2], xmin = level[1], xmax =
        df["Date"].tail(1).item(), colors='red', linestyle='--')
      plt.axhspan(level[2]-banda, level[2]+banda, alpha=0.05, color='red')
    plt.title(str(symbol))
    fig.show()

plot_all(detect_level(df), df)


c = detect_level(df)


cantsoportes = len(c[0])
cantresistencias =len(c[1])
sop=c[0]
res=c[1]
posicionS  = []
precioS = []
posicionCS = []
posicionR = []
precioR = []
posicionCR = []
# Resistencia

for i in range(0,len(res)):
    x = res[i]
    posicionR.append(x[0])
    precioR.append(x[2])


banda =  np.mean(abs(df['Close'] - df['Close'].shift(1)))


for i in range(0,len(res)):
    down = df["Close"][posicionR[i]:]-(precioR[i]-banda)
    down_crossings = np.where(np.diff(np.signbit(down)))[0]
    posicionCR.append(down_crossings[1])

x = detect_level(df)
banda = np.mean(abs(df['Close'] - df['Close'].shift(1)))
precios = df["Close"]
Soporte = np.zeros(len(precios))
Resistencia = np.zeros(len(precios))

Is_in_zone = np.zeros((len(precios),1))

df = df.iloc[: , 1:]
posicionff = df.reset_index()
posicionff = posicionff.reset_index()
posicionff.Date=pd.to_datetime(posicionff.Date)

PosFFS = []
PosFFR = []

for i in range(0,len(res)):
    fecha = np.where(posicionff.Date==pd.Timestamp(res[i][3]))
    PosFFR.append(fecha[0][0])
```

```
for i in range(0,len(x[1])):
        for j in range(x[1][i][0],PosFFR[i],1):
            if (x[1][i][2]-banda)<precios[j]< (x[1][i][2]+banda):
                Resistencia[j] =1
                Is_in_zone[j] = 1


for i in range(0,len(res)):
        Resistencia[posicionR[i]:(posicionCR[i]+posicionR[i])] = 0
        Is_in_zone[posicionR[i]:(posicionCR[i]+posicionR[i])] = 0

# Modelado de Volatilidad - EWMA

Retornos = precios.pct_change()

Lambda = 0.94
Sigma_0 = 0
Variance = np.zeros((len(Retornos),1))

for i in range(0,len(Retornos)):
    if (i == 0):
        Var_0 = Sigma_0**2
    else:
        Var = Lambda*Var_0 + (1-Lambda)*Retornos[i]**2
        Variance[i] = Var
        Var_0 = Var

Sigma_t_EWMA = np.sqrt(Variance)

Fechas = precios.index.tolist()
Sigma_t_EWMA = pd.DataFrame(Sigma_t_EWMA, index = Fechas)
Is_in_zone = pd.DataFrame(Is_in_zone, index = Fechas)

plt.plot(Retornos)
plt.plot(Sigma_t_EWMA)
plt.xlabel("Fecha")
plt.ylabel("Retorno / Volatilidad EWMA")
plt.title(symbol + ": Retornos y Volatilidad EWMA")
plt.legend(["Retornos","Volatilidad EWMA"])
plt.show()
fig,ax=plt.subplots()
ax.plot(Sigma_t_EWMA)
ax.set_xlabel("Fecha")
ax.set_ylabel("Volatilidad Diaria")
ax2 = ax.twinx()
ax2.plot(df["Close"], color="black")
ax2.set_ylabel("Precio")
plt.title(symbol +": Precios y Volatilidad EWMA")
plt.show()

Retornos = pd.DataFrame(Retornos)

Datos = pd.concat([Retornos,Sigma_t_EWMA,Is_in_zone], axis = 1, ignore_index= True)
Datos.columns = ['Retorno','Volatilidad', 'Esta en zona']

# Ahora divido la muestra entre control y experimental

control_Vol = Datos.Volatilidad[Datos['Esta en zona']==0]
experimental_Vol = Datos.Volatilidad[Datos['Esta en zona']==1]

control_Ret = Datos.Retorno[Datos['Esta en zona']==0]
experimental_Ret = Datos.Retorno[Datos['Esta en zona']==1]

Muestra_Aleatoria_Control_Vol = control_Vol.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Vol = experimental_Vol.sample(100, replace=True)

Muestra_Aleatoria_Control_Ret = control_Ret.sample(100, replace=True)
Muestra_Aleatoria_Experimental_Ret = experimental_Ret.sample(100, replace=True)

import seaborn as sns

sns.distplot(Muestra_Aleatoria_Control_Vol, hist = False, kde = True, kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Vol, hist = False, kde = True, kde_kws = {"linewidth":3}, color ="
plt.ylabel("Densidad")
plt.xlabel("Volatilidad")
plt.title(symbol + ":Funciones de densidad de volatilidad (Control vs Experimental)")
plt.legend(["Muestra Control", "Muestra Experimental"])
plt.show()

import statistics
statistics.mean(Muestra_Aleatoria_Control_Vol)
statistics.mean(Muestra_Aleatoria_Experimental_Vol)
sns.distplot(Muestra_Aleatoria_Control_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="r")
sns.distplot(Muestra_Aleatoria_Experimental_Ret, hist = False, kde = True
             , kde_kws = {"linewidth":3}, color ="b")
```

```
    plt.ylabel("Densidad")
    plt.xlabel("Retorno")
    plt.title(symbol +": Funciones de densidad de retornos (Control vs Experimental)")
    plt.legend(["Muestra Control", "Muestra Experimental"])
    plt.show()

    Vol_Results = np.zeros(100)
    Ret_Results = np.zeros(100)

    for j in range(0,100):

        Vol_Mistakes = 0
        Ret_Mistakes = 0

        for i in range(0,1000):

                Vol1 = control_Vol.sample(500, replace=True)
                Vol2 = control_Vol.sample(500, replace=True)
                Ret1 = control_Ret.sample(500,replace=True)
                Ret2 = control_Ret.sample(500,replace=True)

                Vol_pvalue = ks_2samp(Vol1, Vol2)[1]
                Ret_pvalue = ks_2samp(Ret1, Ret2)[1]

                if Vol_pvalue < 0.05:
                    Vol_Mistakes = Vol_Mistakes + 1

                if Ret_pvalue < 0.05:
                    Ret_Mistakes = Ret_Mistakes + 1

        Vol_Results[j] = Vol_Mistakes
        Ret_Results[j]=Ret_Mistakes

    Prob_ET1_Vol = statistics.mean(Vol_Results)/1000
    Prob_ET1_Ret = statistics.mean(Ret_Results)/1000

    print(symbol)
    print("Error tipo 1 Volatilidad")
    print(Prob_ET1_Vol)
    print("Error tipo 1 retorno ")
    print(Prob_ET1_Ret)

    # Ahora hago test de Kolmogorov-Smirnov
    # H0: Ambas distribuciones son iguales
    # HA: Distribuciones son distintas

    ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret)
    ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol)

    print(symbol)
    print("TABLA 1 - NECESITO EL PVALOR y t- VOLATILIDAD")
    print(ks_2samp(Muestra_Aleatoria_Control_Vol, Muestra_Aleatoria_Experimental_Vol))
    print("TABLA 1 - NECESITO EL PVALOR y t - RETORNO")
    print(ks_2samp(Muestra_Aleatoria_Control_Ret, Muestra_Aleatoria_Experimental_Ret))

    pp_x = sm.ProbPlot(Muestra_Aleatoria_Control_Ret)
    pp_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Ret)

    qqplot_2samples(pp_x,pp_y,xlabel = "Cuantiles de retornos Control"
                    , ylabel = "Cuantiles de retornos Experimentales", line = "45")
    plt.show()

    pp2_x = sm.ProbPlot(Muestra_Aleatoria_Control_Vol)
    pp2_y = sm.ProbPlot(Muestra_Aleatoria_Experimental_Vol)

    qqplot_2samples(pp2_x,pp2_y, line = "45", xlabel = "Cuantiles de volatilidades Control"
                    , ylabel = "Cuantiles de volatilidades Experimentales")
    plt.show()

    # Test t de diferencia de medias

    pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
             , correction = False).round(2)
    pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
             , correction = False).round(2)
    print(symbol)
    print("TABLA 2 - NECESITO EL PVALOR y el t - volatilidad")
    print(pg.ttest(x = Muestra_Aleatoria_Control_Vol, y = Muestra_Aleatoria_Experimental_Vol
                   , correction = False).round(2))
    print("TABLA 2 - NECESITO EL PVALOR y el t - RETORNO")
    print(pg.ttest(x = Muestra_Aleatoria_Control_Ret, y = Muestra_Aleatoria_Experimental_Ret
                   , correction = False).round(2))

    experimental_Ret.std()
    control_Ret.std()
```

8. Bibliografía

Ball, R., & Brown, P. (1968). *An Empirical Evaluation of Accounting Income Numbers*. Journal of Accounting Research

Bollerslev, Tim. "Generalized Autoregressive Conditional Heteroskedasticity." *Journal of Econometrics*, vol. 31, no. 3, 1986, https://doi.org/10.1016/0304-4076(86)90063-1.

Borda, R. Arce. "20 Años De Modelos ARCH: Una Visión De Conjunto De Las Distintas Variantes De La Familia." *Redalyc.org*, Asociación Internacional De Economía Aplicada, 1 Jan. 1970, https://www.redalyc.org/articulo.oa?id=30122111.

Brock, William, et al. "Simple Technical Trading Rules and the Stochastic Properties of Stock Returns." *The Journal of Finance*, vol. 47, no. 5, 1992, https://doi.org/10.1111/j.1540-6261.1992.tb04681.x.

Bulkowski, T. N. (2002) Trading Classic Chart Patterns, John Wiley and Sons, Inc., New Jersey.

Campbell, John Y, et al. *The Econometrics of Financial Markets*. New Age International Pvt Ltd Publishers, 2007.

Dolley, J. C. (1933). *Characteristics and Procedure of Common Stock Split-ups*. Harvard Business Review

Engle, Robert F. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation." *Econometrica*, vol. 50, no. 4, 1982, https://doi.org/10.2307/1912773.

Fama, E., Fisher, L., Jensen, M., & Roll, R. (1969). *The Adjustment of Stock Prices to New Information*. International Economic Review

Garzarelli, F., Cristelli, M., Pompa, G, Zazzaria, A. "Memory Effects in Stock Price Dynamics: Evidences of Technical Trading." *Scientific Reports*, vol. 4, no. 1, 2014, https://doi.org/10.1038/srep04487.

Krivin, Dmitry, et al. "Determination of the Appropriate Event Window Length in Individual Stock Event Studies." SSRN Electronic Journal, 2003, doi:10.2139/ssrn.46616

Malkiel, Burton G. "The Efficient Market Hypothesis and Its Critics." *Journal of Economic Perspectives*, vol. 17, no. 1, 2003 https://doi.org/10.1257/089533003321164958.

Mazza, Paolo. "Price Dynamics and Market Liquidity: An Intraday Event Study on Euronext." *SSRN Electronic Journal*, 2012, https://doi.org/10.2139/ssrn.2130611.

Murphy, John J. *Technical Analysis of the Financial Markets: a Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance, 1999.

Nelson, Daniel B. "Conditional Heteroskedasticity in Asset Returns: A New Approach." *Econometrica*, vol. 59, no. 2, 1991, https://doi.org/10.2307/2938260.

Osler, Carol L., Support for Resistance: Technical Analysis and Intraday Exchange Rates. Economic Policy Review, Vol. 6, No. 2, July 2000, Available at SSRN: https://ssrn.com/abstract=888805

Sundara, N. (n.d.). Volatility forecasting - A comparisson of GARCH (1,1) and EWMA models. Retrieved March 15, 2022, from

https://www.researchgate.net/publication/280545501_Volatility_Forecasting_-_A_Comparison_of_GARCH11_and_EWMA_models

Tweneboah-Koduah, S., Atsu, F., Prasad, R. “Reaction of Stock Volatility to Data Breach: an Event Study.” *Journal of Cyber Security and Mobility*, 2020, pp. 1–19., https://doi.org/10.13052/jcsm2245-1439.931.