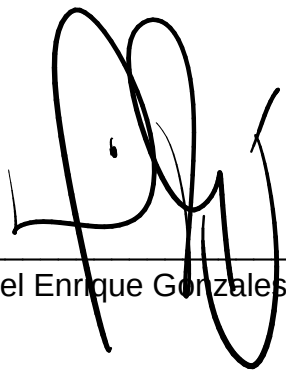
Santiago de Cali, 12 de abril de 2023

Señores
Pontificia Universidad Javeriana Cali
Andrés Felipe Amador Rodríguez
Director Carrera Matemáticas Aplicadas Cali

Cordial saludo,

Me permito informarle que el estudiante de Matemáticas Aplicadas Juan Esteban Galindo con código 8922290, trabajo bajo mi dirección en el Trabajo de Grado denominado **"Pronostico del precio del aceite de soya mediante modelos Holt-Winters, ARIMA y de Redes Neuronales Artificiales"** el cual está listo para ser sustentado.

Atentamente,



Daniel Enrique Gonzales Gómez

Santiago de Cali, 12 de abril de 2023

Señores
Pontificia Universidad Javeriana Cali
Andrés Felipe Amador Rodríguez
Director Carrera Matemáticas Aplicadas Cali

Cordial saludo,

Me permito presentar el proyecto de grado denominado "**Pronostico del precio del aceite de soya mediante modelos Holt-Winters, ARIMA y de Redes Neuronales Artificiales"** con el fin de cumplir con los requisitos exigidos por la Universidad para optar por el título de Profesional en Matemáticas Aplicadas.

Atentamente,

_____
Juan Esteban Galindo Martinez
C.C. 1234195974
ID:8922290

PRONÓSTICO DEL PRECIO DEL ACEITE DE SOYA MEDIANTE MODELOS HOLT-WINTERS, ARIMA Y REDES NEURONALES ARTIFICIALES

JUAN ESTEBAN GALINDO MARTÍNEZ

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA Y CIENCIAS
SANTIAGO DE CALI
2023

PRONÓSTICO DEL PRECIO DEL ACEITE DE SOYA MEDIANTE MODELOS HOLT-WINTERS, ARIMA Y REDES NEURONALES ARTIFICIALES

JUAN ESTEBAN GALINDO MARTÍNEZ

Trabajo de grado para optar por el título de matemático aplicado

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA Y CIENCIAS
SANTIAGO DE CALI
2023

Índice general

Capítulo 1

Introducción

En los negocios en general existe un tipo de **vendedor** que no puede o no desea vender su producto de manera inmediata. Por ejemplo, los productores de materias primas. Entre el presente y la venta el precio pude aumentar o mantenerse estable, en ese caso el vendedor se vería beneficiado. Por el contrario, si baja se vería perjudicado. Similarmente, existe un tipo de **comprador** que requiere un producto al cabo de cierto tiempo. Por ejemplo, las empresas de procesamiento de alimentos que con anticipación saben que necesitaran ciertos insumos para su proceso. Si el precio del producto sube el comprador se verá afectado, mientras que si se mantiene o cae se verá favorecido.

Es justamente para evitar este riesgo que surgieron las negociaciones por adelantado, en las cuales las partes se comprometen a fijar un precio independientemente de los cambios que sufra este en el mercado. Esta práctica sentó las bases de lo que hoy conocemos como mercados de futuros. En ellos se negocia por adelantado, pero a través de un intermediario imparcial que le da trasparencia y confiabilidad a la negociación. Este tipo de contratos se encuentran regulados y centralizados para garantizar su cumplimiento en la fecha, cantidad y calidad que se haya acordado.

En los mercados de futuros no solo participan aquellos que intentan cubrirse de movimientos desfavorables en precios, llamados **coberturistas**, sino que existen otros participantes que intentan generar ganancias a través de las especulaciones que tienen acerca de los movimientos del mercado, llamados **especuladores**. Dada la facilidad para participar en estos mercados de manera virtual el volumen de especuladores crece cada vez más. En todo caso, tanto para especuladores como para coberturistas poder predecir de manera correcta los movimientos del mercado se refleja en una maximización de las ganancias o minimización de las perdidas.

Dos técnicas clásicas en el estudio de series de tiempo son la metodología de Holt-Winters y la metodología Box-Jenkins. Pero según Bontempi[9] en los últimos años los modelos de aprendizaje automático han llamado la atención y se han establecido como serios competidores de los modelos estadísticos clásicos en la comunidad de pronósticos.

En [8] se concluyó que las técnicas de aprendizaje profundo funcionan mejor para problemas de series temporales en comparación con los métodos clásicos cuando se configuran y entrenan correctamente.

Debido a esto se investigo acerca de las técnicas más utilizadas para pronósticos de series de tiempo utilizando aprendizaje profundo.

En [5] se investigaron y compararon las arquitecturas más utilizadas recientemente, los resultados muestran que la memoria a largo plazo (LSTM) fue la arquitectura más precisa.

En [7] se menciona que los modelos de redes neuronales recurrentes LSTM son uno de los mejores modelos para la extracción de patrones de las características de entrada.

Así pues, en este proyecto de investigación se plantea un modelo de pronósticos para los precios de los futuros de aceite de soya en la bolsa de Chicago comparando las dos técnicas clásicas con las técnicas modernas.

Capítulo 2

Marco teórico

A continuación se presenta un resumen con algunos conceptos fundamentales para el análisis de series de tiempo extraídos de Hanke[1].

Serie de tiempo es una sucesión de datos que se recopilan periódicamente. Es decir, es aquel conjunto de mediciones de una variable que se ejecutan cada cierto tiempo. Ejemplos de series de tiempo son las ventas mensuales de una compañía o la cantidad de pacientes que ingresan a urgencias en un hospital cada día.

Pronóstico es una estimación acerca del comportamiento de una variable en periodos futuros basándose en la información disponible de los periodos pasados.

La notación más común para pronósticos es:

Y_t = valor de una serie en el periodo t
$\hat{Y}_t$ = valor pronosticado de la serie en el periodo t

Existen diversas técnicas de pronósticos para series de tiempo y para elegir una adecuada es muy importante identificar y comprender sus patrones. La descomposición de la serie en cuatro componentes es útil para el análisis de dichos patrones.

Componentes de una serie de tiempo

- **Tendencia (T):** Crecimiento o decrecimiento general a largo plazo.
- **Componente cíclico (C):** Oscilaciones a lo largo de la tendencia que no siguen un periodo fijo. Este componente es difícil de identificar y por lo general se incluye como parte de la tendencia.
- **Componente estacional (S):** Patrón que tienen en común los mismos periodos de estudio en distintos momentos. Por ejemplo, los meses enero de cada año o los sábados de cada semana.
- **Componente irregular (I):** Es el componente que resulta al eliminar las demás componentes de la serie.

Autocorrelación es el nivel de correlación o relación lineal que hay entre una variable y ella misma retrasada. Este concepto es empleado para el estudio de patrones como la tendencia y la estacionalidad. El coeficiente de autocorrelación entre Y_t y Y_{t-k}, donde k es la cantidad de retrasos, se calcula asi:

$$r_k = \frac{\sum_{t=k+1}^{n}(Y_t - \overline{Y})(Y_{t-k} - \overline{Y})}{\sum_{t=1}^{n}(Y_t - \overline{Y})^2} \tag{2.1}$$

donde

r_k = coeficiente de autocorrelación para un retraso de k periodos
$\overline{Y}$ = media de los valores de la serie
Y_t = observación en el periodo t
Y_{t-k} = observación k periodos anteriores o durante un periodo $t-k$

Correlograma es un gráfico de los coeficientes de autocorrelación para varios retrasos de tiempo.

Medición del error de pronóstico

En cada paso del tiempo, el residuo o error de pronóstico se define como la diferencia entre el valor de original la serie y el valor pronosticado.

$$\varepsilon_t = Y_t - \hat{Y}_t \tag{2.2}$$

Hay varios métodos para evaluar la exactitud de una técnica de pronósticos y se utilizan tanto para comparar dos o más técnicas de pronósticos como para medir la confiabilidad de una técnica en particular:

- **Error medio absoluto (MAE):** Consiste en promediar el valor absoluto de los errores de pronóstico, por lo cual tiene las mismas unidades de la serie original y se interpreta como el tamaño promedio de los errores sin tener en cuenta la dirección de estos.

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|Y_t - \hat{Y}_t| \tag{2.3}$$

- **Error cuadrático medio (MSE):** Se calcula promediando los errores de pronóstico al cuadrado, por lo que no tiene las mismas unidades de la serie original. La idea de elevar al cuadrado es evidenciar cuando existen errores grandes a pesar de que no sean muchos, ya que esto podría pasarse por alto cuando solo se promedian los errores.

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(Y_t - \hat{Y}_t)^2 \tag{2.4}$$

- **Raíz cuadrada del MSE (RMSE) :** Consiste en calcular la raíz cuadrada del MSE, así el indicador queda nuevamente en las unidades de la serie original y esto facilita su interpretación.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Y_t - \hat{Y}_t)^2} \tag{2.5}$$

- **Error porcentual absoluto medio (MAPE):** Se calcula promediando los errores porcentuales absolutos. El error porcentual absoluto en cada periodo es la razón entre el error absoluto y el valor absoluto de la serie original en ese periodo.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|Y_t - \hat{Y}_t|}{|Y_t|} \tag{2.6}$$

- **Error porcentual medio (MPE):** Se calcula promediando los errores porcentuales. Al no tomarse el error absoluto sino el error con su respectivo signo podemos utilizar el indicador para saber si la técnica de pronóstico esta sesgada, esto es, si frecuentemente sobreestima o subestima la serie original. Si el resultado es cercano a cero significa que la técnica no está sesgada, mientras que si es grande y positivo está sobreestimando y si es grande y negativo está subestimando.

$$MPE = \frac{1}{n}\sum_{t=1}^{n}\frac{(Y_t - \hat{Y}_t)}{Y_t} \tag{2.7}$$

2.1 Metodología Box-Jenkins

Se denomina metodología Box-Jenkins a una técnica utilizada para la construcción de un modelo de pronósticos de series de tiempo utilizando modelos ARIMA.

Esta metodología se diferencia de muchas otras porque no requiere algún patrón especifico en la serie que se desea pronosticar. Por lo que resulta apropiada para una gran cantidad de series independientemente de si son estacionarias o no. Incluso puede trabajar con series estacionales. Esto hace que sea una técnica ampliamente utilizada.

El proceso de construcción consiste en identificar un modelo ARIMA tentativo, estimar los parámetros y validar si la desviación es aceptable. En caso de que no lo sea, se debe realizar otra selección de modelo hasta que se considere apto.

Esta metodología supone que la serie puede ser representada por los siguientes tipos de modelos:

2.1.1 Modelos autorregresivos AR(p)

Los modelos autorregresivos de orden p generan un pronóstico a partir de una regresión en la que se utiliza la serie original retrasada $t-p$ veces como variables explicativas.

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + \varepsilon_t \tag{2.8}$$

donde

Y_t = Variable de respuesta.
$Y_{t-1}, Y_{t-2}, ..., Y_{t-p}$ = Variable de respuesta retrasada.
$\phi_0, \phi_1, ..., \phi_p$ = Coeficientes de regresión a estimar.
ε_t = Error de estimación.

2.1.2 Modelos de promedios móviles MA(q)

Los modelos de promedio móvil de orden k generan un pronóstico a partir de una combinación lineal de los $t-q$ errores pasados.

$$Y_t = \mu + \varepsilon_t - \omega_1 \varepsilon_{t-1} - \omega_2 \varepsilon_{t-2} - ... - \omega_q \varepsilon_{t-q} \tag{2.9}$$

donde

Y_t = Variable de respuesta.
μ = Promedio constante.
$\omega_1, \omega_2, ..., \omega_q$ = Parámetros a estimar.
ε_t = Error de estimación.
$\varepsilon_{t-1}, \varepsilon_{t-2}, ..., \varepsilon_{t-q}$ = Errores en periodos anteriores.

2.1.3 Modelos de promedio móvil autorregresivos ARMA(p,q)

Los modelos ARMA de orden p y q son la combinación de los modelos autorregresivos de orden p y los modelos de promedios móviles de orden q y depende tanto de los errores pasados como de las observaciones pasadas de la serie.

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + ... + \phi_p Y_{t-p} + \varepsilon_t - \omega_1 \varepsilon_{t-1} - \omega_2 \varepsilon_{t-2} - ... - \omega_q \varepsilon_{t-q} \tag{2.10}$$

2.1.4 Modelos de promedios móviles integrados autorregresivos ARIMA(p,d,q)

Cuando la serie que se quiere pronosticar no es estacionaria muchas veces es posible convertirla en estacionaria a través de diferencias. Al obtener la serie estacionaria se puede ajustar un modelo ARMA y la cantidad de diferencias requeridas se denota como el parámetro d. Así pues, los modelos ARIMA complementan los modelos arma para series no estacionarias.

2.1.5 Modelos de promedios móviles integrados autorregresivos estacionales ARIMA$(p,d,q)(P,D,Q)[m]$

Existe una versión de los modelos ARIMA especial para las series que tiene un patrón estacional de longitud m. Funciona igual que la versión estándar, pero incluye modelos ARIMA para capturar la variación de la parte estacional. También son llamados modelos SARIMA.

Cristerio de seleccion de modelo

En casos donde varios modelos se ajustan correctamente a los datos, es preferible el que tenga menor cantidad de parametros, por simplicidad. Pero si tienen igual cantidad de parametros es preferible el que tenga menor error cuadratico medio. Sin embargo, un modelo con mas parametros puede tener error cuadratico medio bastante menor. Para evitar esa ambiguedad se diseño el criterio de informacion de Akeike (AIC)

$$AIC = \ln \hat{\sigma}^2 + \frac{2}{n} r \tag{2.11}$$

donde

$\hat{\sigma}^2$ = Error cuadrático medio.
n = Número de observaciones.
r = Número total de parámetros en el modelo ARIMA.

2.2 Metodología de Redes Neuronales

A continuación se presenta un resumen con algunos conceptos fundamentales sobre redes neuronales extraídos de Konasani[3] y de Gopal[4].

Las **redes neuronales artificiales** (ANN) son sistemas de reconocimiento de patrones cuya estructura y funcionamiento están inspiradas en las redes neuronales biológicas. Estas intentan emular el proceso de solución de problemas del cerebro humano. El cual consiste en acumular constantemente el conocimiento ganado en la solución de problemas previos y adaptarlo para aplicarlo a nuevas situaciones. Es por esto que las redes neuronales funcionan a través de ejemplos, con los que esta se entrena e intenta descifrar las relaciones que existen en dicha información para poder replicarlos.

2.2.1 Estructura básica de una red neuronal artificial

Un nodo o neurona es el elemento básico de procesamiento de una red neuronal. La figura 2.1 muestra la estructura de una neurona artificial. Los x_i son las variables de entrada y cada una tiene un coeficiente w_i asociado denominado peso. La suma ponderada a de las entradas con sus respectivos pesos pasa por una función σ y el

resultado se denomina respuesta o salida $\hat{y}$. Esta función se denomina función de activación o función de transferencia.

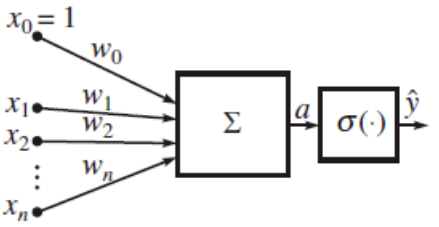


Figura 2.1: Estructura de una Neuronal Artificial Básica[4]

Una red neuronal básica se compone de capas y en cada una de ellas se encuentran nodos totalmente conectados con todos los nodos de la anterior capa y la siguiente. Estas capas se clasifican en: capa de entrada, una o varias capas ocultas y capa de salida. La capa de entrada contiene las variables predictoras, por lo que la cantidad de nodos en ella es la cantidad de variables de entrada. La capa de salida contiene las variables objetivo, por lo que la cantidad de nodos en ella es la cantidad de variables que se intentan predecir. Pero no hay una regla general para determinar la cantidad de capas ocultas ni la cantidad de nodos en ellas, básicamente se encuentran por tanteo. Este tipo de red es llamada **Perceptrón Multicapa** (MPL). La figura 2.2 muestra la estructura básica de una red neuronal.

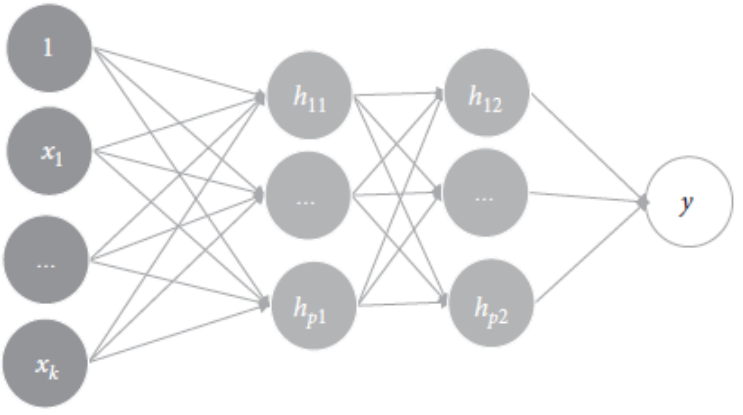


Figura 2.2: Estructura básica de una Red Neuronal[3]

Construir una red neuronal, consiste en encontrar los valores óptimos de los pesos. Para ello se utiliza el algoritmo o regla de aprendizaje. El principal algoritmo y del cual se desprenden varios más se llama **algoritmo de retropropagación**, también conocido como descenso del gradiente. Consiste en asignarle valores aleatorios a los pesos inicialmente, calcular las sumas ponderadas y pasarlas por las funciones de activación de todas las neuronas hasta llegar a un primer pronostico. Como se conocen los valores reales de las variables objetivo se calculan los errores y se propagan hacia atrás para encontrar la contribución del error en las capas ocultas. Luego se actualizan

los pesos en la dirección que se reduce el error, para esto se usa el método del descenso del gradiente. El proceso finaliza cuando el error llega a ser tan pequeño como se considere aceptable.

Existen problemas de pronóstico donde se requiere modelar datos secuenciales, es decir el dato actual depende de los anteriores. Note que este es el caso de las series de tiempo, donde existe autocorrelación. Para modelar este tipo de problemas donde se requiere la noción de memoria se crearon las **redes neuronales recurrentes** (RNN), ya que las redes básicas consideran independientes las variables de entrada. Lo que caracteriza las redes recurrentes es que almacena información sobre la historia de los estados anteriores del sistema, se alimenta a sí mismo. La figura 2.3 muestra una representación de modelos de redes recurrentes.

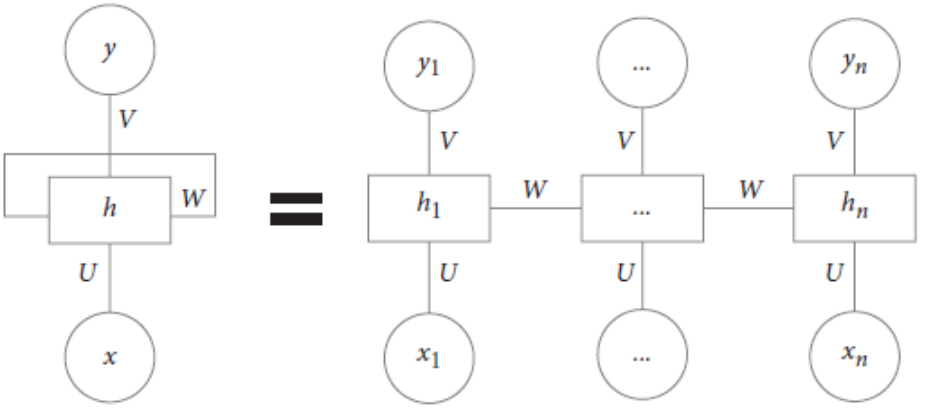


Figura 2.3: Red Neuronal Recurrente[3]

Sin embargo, las redes neuronales recurrentes básicas no funcionan muy bien para secuencias largas, es decir no tienen memoria a largo plazo. La razón es la desaparición de los gradientes. Así fue como surgieron los modelos de memoria a corto plazo LSTM. Los cuales contienen varias modificaciones de las redes recurrentes básicas pero debido a su complejidad no se profundizo en ellas en este documento.

2.3 Metodología estacional de Holt-Winters

A continuación se presenta un resumen acerca de los modelos de Holt-Winters extraído de Hyndman[12].

Esta familia de modelos es una generalización de los modelos de tendencias de Holt, que a su vez es una generalización de los modelos de suavizado exponencial. En el suavizado exponencial se hace un promedio ponderado de las observaciones de la serie asignándole mayor o menor peso a las observaciones más recientes dependiendo el parámetro de suavizado. El modelo de Holt mejoro esto agregando otra componente para modelar la tendencia de manera lineal y más tarde se mejoró este modelo para

amortiguar la tendencia. Después se le incluyo una nueva componente para captar la estacionalidad y se llamó modelo de Holt-Winters.

Es así como este tipo de modelos tiene la ecuación de pronóstico y tres ecuaciones de suavizado una para cada componente: nivel l_t, tendencia b_t y estacional s_t. Cada una de ellas tiene su respectivo parámetro de suavizado α, β y γ.

Por último, existen dos versiones del modelo, una aditiva y una multiplicativa. La elección depende de la variación de los datos, si la variación se mantiene estable es preferible usar el modelo aditivo. Por el contrario, si la variación crece es preferible el multiplicativo. En este caso mostramos la versión multiplicativa.

$$\hat{Y}_{t+h} = (l_t + hb_t)s_{t+h} \tag{2.12}$$

$$\begin{aligned} l_t &= \alpha(\frac{Y_t}{s_{t-m}}) + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1} \\ s_t &= \gamma(\frac{Y_t}{l_t}) + (1-\gamma)s_{t-m} \end{aligned}$$

Capítulo 3

Resultados principales

3.1 Análisis de la serie de tiempo

En este trabajo se analizó la serie del precio de los futuros de aceite de soja en la bolsa de valores de Chicago. Los datos se extrajeron de Yahoo Finanzas por medio de la función *getSymbols* del paquete *quantmod*. Dicha bolsa funciona de lunes a viernes, con algunas excepciones en días festivos. Esto complejiza el análisis debido a la periodicidad irregular. Existen varias técnicas para rellenar datos faltantes en series, pero por supuesto esto agrega cierto margen de error. Para evitar ese problema se analizó la serie con periodicidad semanal. La figura 3.1 muestra la gráfica de la serie desde el año 2016 hasta marzo del 2023. La figura 3.2 muestra la descomposición de la misma usando el modelo multiplicativo de los componentes debido a que gráficamente se puede apreciar que la variabilidad de los datos viene en aumento.

Figura 3.1: Serie del precio del aceite de soja

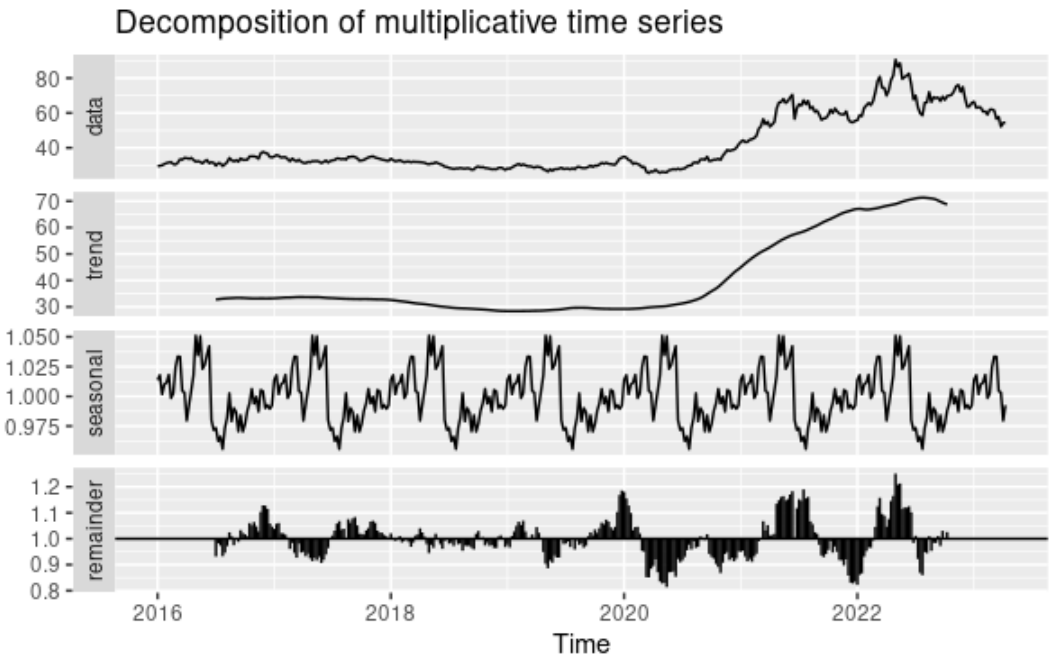


Figura 3.2: Descomposición multiplicativa de la serie

De la descomposición podemos notar que el precio se mantiene medianamente estable hasta el año 2020, pero a partir de ahí aparece una tendencia creciente, debido principalmente a la pandemia de COVID 19 y más tarde por la guerra de Ucrania. Esto representa un reto importante a la hora de pronosticar, pues el modelo que se elija debe ser capaz de identificar la dinámica habitual de este mercado, pero sin perder de vista la tendencia creciente de los últimos años.

Adicionalmente, la serie parece tener cierta estacionalidad en la cual el precio sube la primera mitad del año y cae la segunda mitad. El grafico de estacionalidad de la figura 3.3 nos permite comparar el precio año a año. Note que, los años 2021 y 2022 tienen cierta similitud en sus patrones, aunque el inicio del año 2023 no sigue el mismo patrón. En los años previos es difícil identificar una estacionalidad demasiado marcada.

3.2 Estimación de modelos

Para seleccionar el modelo que mejor pronostica la serie es tradición dividir la serie en dos. La primera parte de la serie se denomina serie de entrenamiento y la segunda serie de prueba o validación. Sobre el primer conjunto de datos cada modelo pronostica la misma cantidad de datos que tenga el segundo conjunto. Finalmente se calculan las diferentes medidas de error con cada pronostico y se determina cual fue más acertado. En nuestro caso la serie de entrenamiento va hasta la última semana del año 2022 y la serie de prueba son los datos del año 2023.

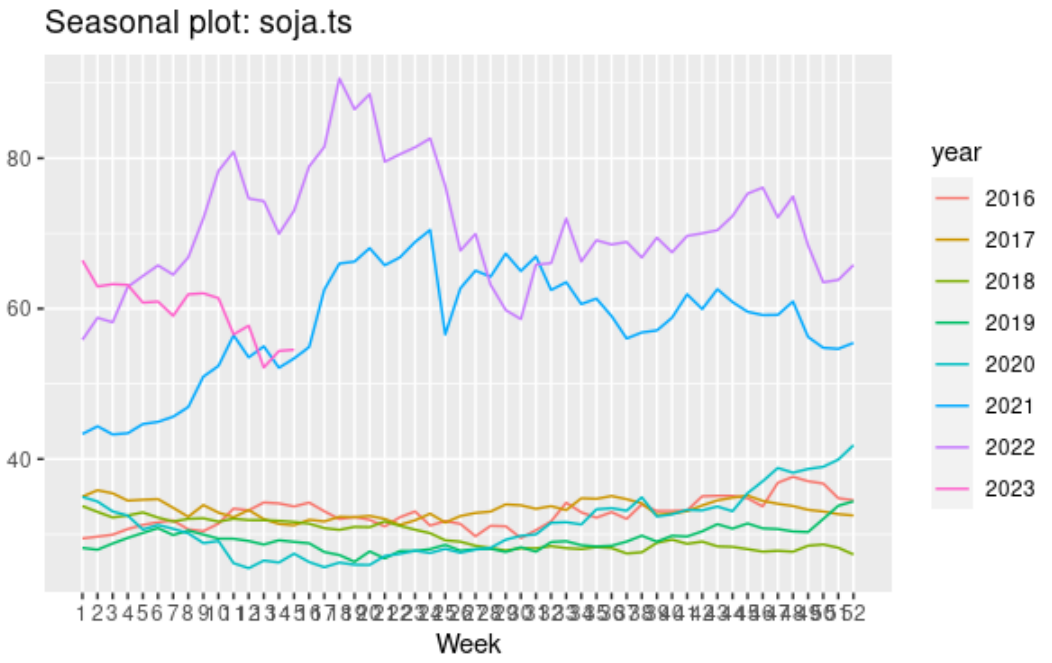


Figura 3.3: Gráfico de estacionalidad de la serie

Para los pronósticos se utilizó el lenguaje de programación R y el entorno RStudio. En todos los gráficos de pronósticos la serie de entrenamiento es de color negro, el ajuste del modelo sobre la serie de entrenamiento es de color rojo y el pronóstico es de color azul con sus respectivos intervalos de confianza. Además, se hizo zoom a partir del 2020 para mayor claridad.

3.2.1 Modelo ARIMA

Lo primero es determinar si la serie es o no estacionaria. De grafica de la serie se puede notar que esta tiene una tendencia creciente a partir del año 2020, con lo cual no es estacionaria. Esto se verifica al analizar la función de autocorrelación de la figura 3.4. Note que hay muchos retrasos significativos y como estos van cayendo lentamente a cero.

De hecho, el paquete *forecast* ofrece la función *ndiffs* que indica la cantidad de diferenciaciones necesarias para eliminar la tendencia. En este caso, indica que requiere una diferenciación $d = 1$. El paquete *forecast* también ofrece la función *nsdiffs* que indica la cantidad de diferenciaciones estacionales en caso de que se requieran. En este caso indica que no requiere $D = 0$.

Efectivamente, en la gráfica de la serie diferenciada de la figura 3.5 se puede notar que esta si varia alrededor de nivel fijo, prácticamente cero.

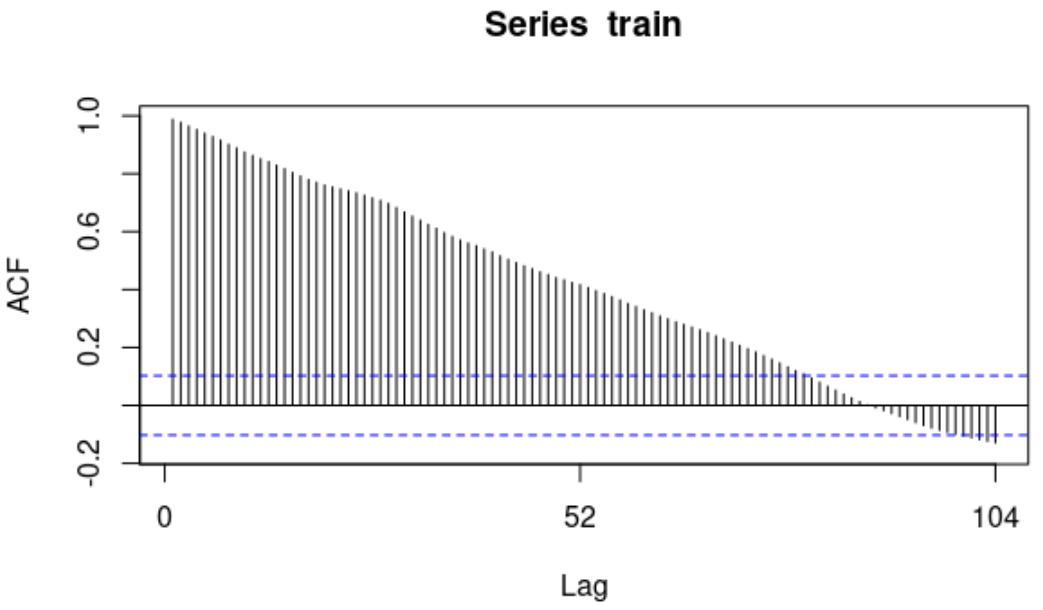


Figura 3.4: Función de autocorrelación de la serie

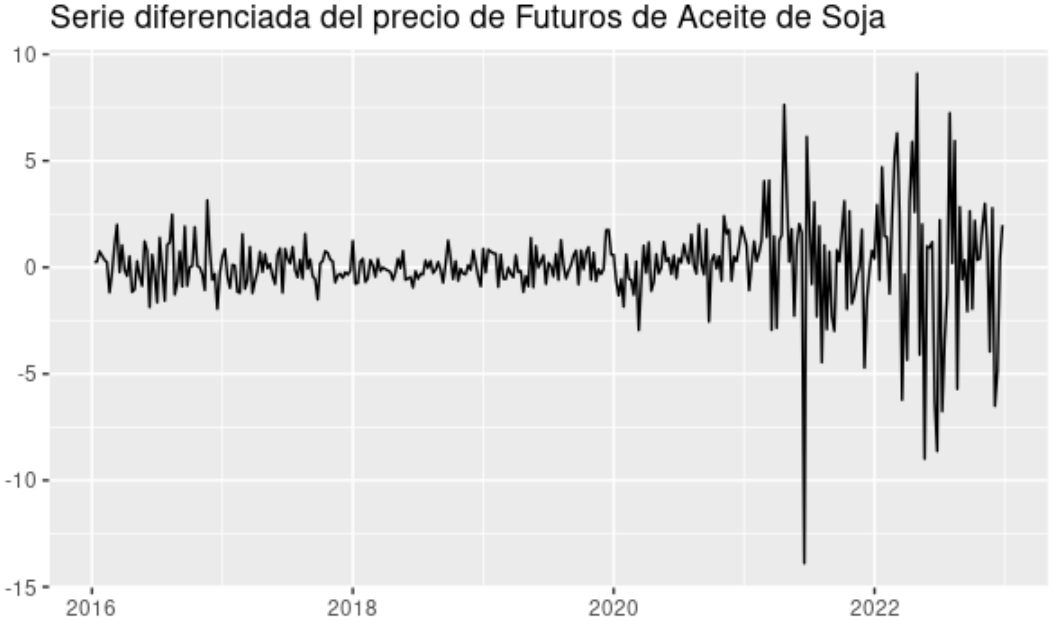


Figura 3.5: Serie diferenciada

Para validar si la serie es estacionaria se tiene la prueba Dickey–Fuller, donde la hipótesis nula es que la serie no es estacionaria. Y la prueba de estacionariedad KPSS, donde la hipótesis nula es que la serie es estacionaria. El paquete *tseries* ofrece ambas pruebas con las funciones *adf.test* y *kpss.test*. Los resultados de las pruebas se encuentran en las figuras 3.6 y 3.7.

```
        Augmented Dickey-Fuller Test

data:  diff(train)
Dickey-Fuller = -6.7249, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
```

Figura 3.6: Prueba Dickey–Fuller

```
        KPSS Test for Level Stationarity

data:  diff(train)
KPSS Level = 0.12196, Truncation lag parameter = 5, p-value = 0.1
```

Figura 3.7: Prueba KPSS

En la prueba Dickey–Fuller se rechaza la hipótesis nula porque el valor $p = 0{,}01$ es menor que $\alpha = 0{,}05$. En la prueba KPSS no se puede rechazar la hipótesis nula porque el valor $p = 0{,}1$ es mayor que $\alpha = 0{,}05$. En ambos casos la conclusión es que la serie es estacionaria.

Ahora bien, para identificar el orden de la parte autorregresiva y la parte de medias móviles tanto regulares como estacionales se analiza la función de autocorrelación y la función de autocorrelación parcial de la serie diferenciada, estas se encuentran en la figura 3.8 y 3.9.

En la figura 3.8, existe autocorrelación significativa en el rezago 52, pero no en el 104. Este comportamiento sugiere un modelo MA(1) para la parte estacional. Similarmente en la figura 3.9 existe autocorrelación parcial en el rezago 52, pero no en el 104. Este comportamiento sugiere un modelo AR(1) o un ARMA(1,1) para la parte estacional.

Por otra parte, en la figura 3.8 existe correlación significativa en los rezagos dos y tres, lo que sugiere un modelo MA(3). También, en la figura 3.9 existe autocorrelación parcial significativa en los rezagos dos y tres, lo que sugiere un modelo MA(3) o tal vez un ARMA(3,3).

De esta manera se tienen varios posibles modelos ARIMA, los cuales siguiendo la metodología Box-Jenkins se compararán con el criterio AIC para determinar cuál de ellos es mejor. Esta comparación es importante ya que los modelos sencillos, en el sentido de menos parámetros para estimar, son preferibles. No es recomendado quedarse solo con la combinación de todos los órdenes.

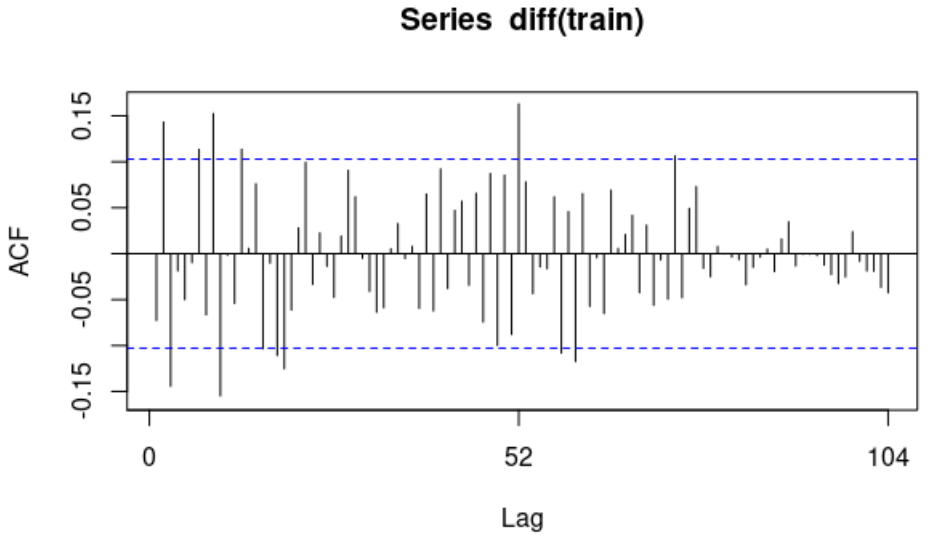


Figura 3.8: Funcion de autocorrelación de la serie diferenciada

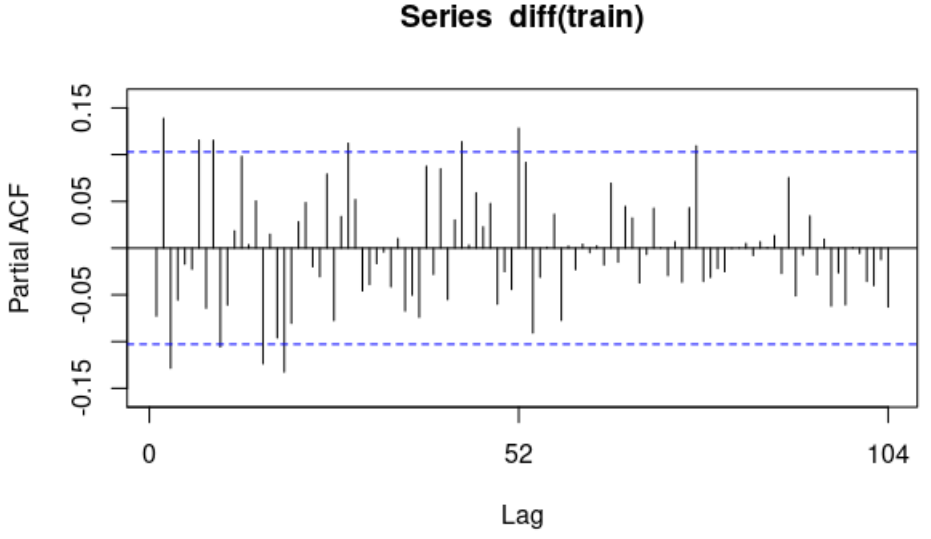


Figura 3.9: Función de autocorrelación parcial de la serie diferenciada

El paquete *forecast* contiene la función *Arima* que permite ajustar un modelo ARIMA a una serie según el orden que se le indique. En este caso se ajustaron nueve modelos ARIMA cubriendo las combinaciones que surgieron del análisis previo. Los modelos que se probaron están en la figura 3.10 y la comparación de sus AIC están en la figura 3.11.

```
a1=Arima(train,order=c(0,1,3),seasonal=c(1,0,0))
a2=Arima(train,order=c(0,1,3),seasonal=c(0,0,1))
a3=Arima(train,order=c(0,1,3),seasonal=c(1,0,1))

a4=Arima(train,order=c(3,1,0),seasonal=c(0,0,1))
a5=Arima(train,order=c(3,1,0),seasonal=c(1,0,0))
a6=Arima(train,order=c(3,1,0),seasonal=c(1,0,1))

a7=Arima(train,order=c(3,1,3),seasonal=c(1,0,0))
a8=Arima(train,order=c(3,1,3),seasonal=c(0,0,1))
a9=Arima(train,order=c(3,1,3),seasonal=c(1,0,1))
```

Figura 3.10: Modelos ARIMA probados

	df	AIC
a1	5	1534.074
a2	5	1531.809
a3	6	1533.677
a4	5	1532.050
a5	5	1534.221
a6	6	1533.892
a7	8	1527.494
a8	8	1524.778
a9	9	1526.720

Figura 3.11: Comparación del AIC

El modelo con el criterio AIC más bajo es el a8. Los parámetros ajustados del modelo se encuentran en la figura 3.12.

```
Series: train
ARIMA(3,1,3)(0,0,1)[52]

Coefficients:
         ar1      ar2      ar3      ma1     ma2     ma3    sma1
      0.5997  -0.2645  -0.5609  -0.6977  0.4627  0.3886  0.3407
s.e.  0.1625   0.1813   0.1504   0.1753  0.1909  0.1700  0.0653

sigma^2 = 3.742:  log likelihood = -754.39
AIC=1524.78   AICc=1525.18   BIC=1555.93
```

Figura 3.12: Modelo ARIMA propuesto

En este punto vale mencionar que existe una función *auto.arima* del paquete *forecast* que calcula el mejor modelo ARIMA, entre muchas combinaciones, según el criterio AIC con solo pasarle una serie como parámetro. En la figura 3.13 se presentan todos los modelos ARIMA que *auto.arima* comparo y el modelo que fue elegido. Los parámetros ajustados del modelo y sus órdenes se encuentran en la figura 3.14.

```
ARIMA(0,1,0)                : 1557.401
ARIMA(0,1,0)(0,0,1)[52]     : 1538.739
ARIMA(0,1,0)(1,0,0)[52]     : 1541.282
ARIMA(0,1,0)(1,0,1)[52]     : 1540.599
ARIMA(0,1,1)                : 1558.007
ARIMA(0,1,1)(0,0,1)[52]     : 1538.768
ARIMA(0,1,1)(1,0,0)[52]     : 1541.542
ARIMA(0,1,1)(1,0,1)[52]     : 1540.626
ARIMA(0,1,2)                : 1553.025
ARIMA(0,1,2)(0,0,1)[52]     : 1533.649
ARIMA(0,1,2)(1,0,0)[52]     : 1536.116
ARIMA(0,1,2)(1,0,1)[52]     : 1535.597
ARIMA(0,1,3)                : 1549.163
ARIMA(0,1,3)(0,0,1)[52]     : 1531.977
ARIMA(0,1,3)(1,0,0)[52]     : 1534.242
ARIMA(0,1,3)(1,0,1)[52]     : 1533.913
ARIMA(0,1,4)                : 1551.073
ARIMA(0,1,4)(0,0,1)[52]     : 1532.848
ARIMA(0,1,4)(1,0,0)[52]     : 1535.332
ARIMA(0,1,5)                : 1550.876
ARIMA(1,1,0)                : 1557.623
ARIMA(1,1,0)(0,0,1)[52]     : 1538.236
ARIMA(1,1,0)(1,0,0)[52]     : Inf
ARIMA(1,1,0)(1,0,1)[52]     : Inf
ARIMA(1,1,1)                : 1553.944
ARIMA(1,1,1)(0,0,1)[52]     : 1535.399
ARIMA(1,1,1)(1,0,0)[52]     : 1538.067
ARIMA(1,1,1)(1,0,1)[52]     : 1537.267
ARIMA(1,1,2)                : 1550.379
ARIMA(1,1,2)(0,0,1)[52]     : 1533.308
ARIMA(1,1,2)(1,0,0)[52]     : 1535.537
ARIMA(1,1,2)(1,0,1)[52]     : Inf
ARIMA(1,1,3)                : 1550.85
ARIMA(1,1,3)(0,0,1)[52]     : 1532.529
ARIMA(1,1,3)(1,0,0)[52]     : 1535.014
ARIMA(1,1,4)                : 1552.888
ARIMA(2,1,0)                : 1552.396
ARIMA(2,1,0)(0,0,1)[52]     : 1533.513
ARIMA(2,1,0)(1,0,0)[52]     : 1535.987
ARIMA(2,1,0)(1,0,1)[52]     : Inf
ARIMA(2,1,1)                : 1551.55
ARIMA(2,1,1)(0,0,1)[52]     : 1534.078
ARIMA(2,1,1)(1,0,0)[52]     : 1536.378
ARIMA(2,1,1)(1,0,1)[52]     : 1536.003
ARIMA(2,1,2)                : 1545.9
ARIMA(2,1,2)(0,0,1)[52]     : 1535.231
ARIMA(2,1,2)(1,0,0)[52]     : Inf
ARIMA(2,1,3)                : 1552.612
ARIMA(3,1,0)                : 1548.594
ARIMA(3,1,0)(0,0,1)[52]     : 1532.218
ARIMA(3,1,0)(1,0,0)[52]     : 1534.389
ARIMA(3,1,0)(1,0,1)[52]     : 1534.128
ARIMA(3,1,1)                : 1549.559
ARIMA(3,1,1)(0,0,1)[52]     : 1532.211
ARIMA(3,1,1)(1,0,0)[52]     : 1534.586
ARIMA(3,1,2)                : 1551.337
ARIMA(4,1,0)                : 1549.479
ARIMA(4,1,0)(0,0,1)[52]     : 1531.627
ARIMA(4,1,0)(1,0,0)[52]     : 1534.031
ARIMA(4,1,1)                : 1551.488
ARIMA(5,1,0)                : 1551.482

Best model: ARIMA(4,1,0)(0,0,1)[52]
```

Figura 3.13: Modelos ARIMA comparados por *auto.arima*

```
Series: train
ARIMA(4,1,0)(0,0,1)[52]

Coefficients:
          ar1     ar2      ar3      ar4    sma1
      -0.0651  0.1413  -0.1026  -0.0872  0.3193
s.e.   0.0525  0.0522   0.0530   0.0534  0.0669

sigma^2 = 3.842:  log likelihood = -759.7
AIC=1531.39   AICc=1531.63   BIC=1554.76
```

Figura 3.14: Modelo ARIMA elegido por *auto.arima*

Note que el modelo propuesto mediante el análisis tiene un criterio AIC menor que el propuesto por *auto.arima*. Con lo cual es mejor y se debe seguir trabajando con él.

El paquete *forecast* ofrece la función *checkresiduals* que permite verificar gráficamente los supuestos acerca de los residuos del modelo para saber si este adecuado. La figura 3.15 tiene el análisis de los residuos del modelo ARIMA.

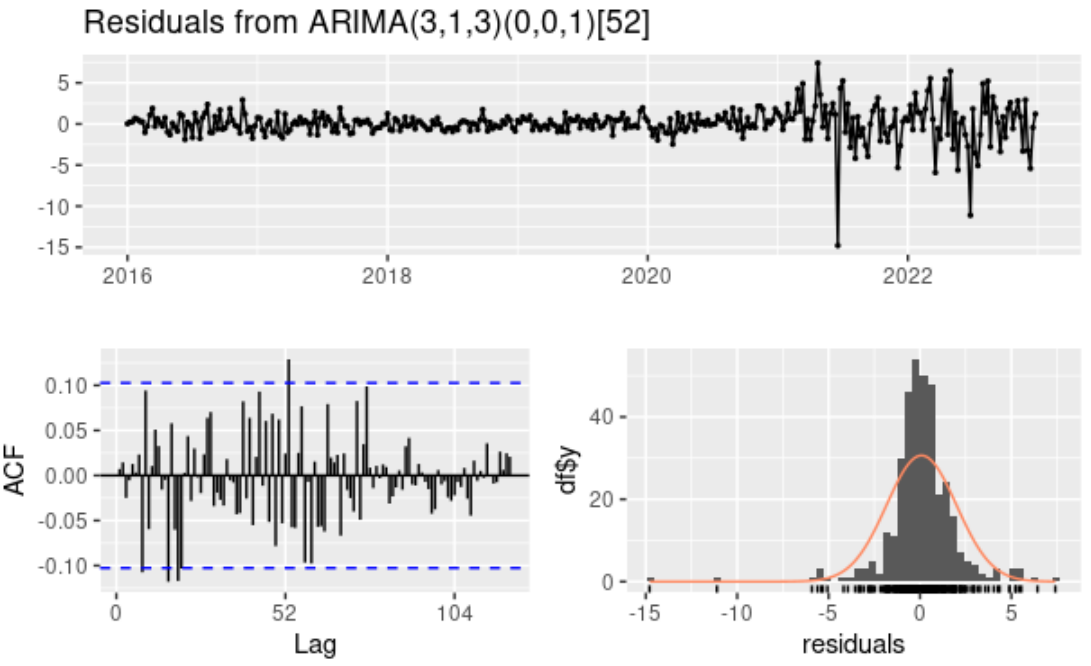


Figura 3.15: Análisis de residuos modelo ARIMA

Si el modelo aproxima correctamente los datos, los residuos deben ser aproximadamente aleatorios. Note que estos varían alrededor de un nivel fijo, su función de autocorrelación solo tiene unos pocos rezagos significativos y su distribución tiene un comportamiento parcialmente comparable con la distribución normal. Si bien los residuos no son completamente aleatorios, fue la mejor aproximación que conseguimos con modelos ARIMA.

Los pronósticos con el modelo ARIMA seleccionado se encuentran en el cuadro 3.1 y la figura 3.16.

Finalmente, note que todos los modelos ARIMA que se tuvieron en cuenta generaban pronósticos similares. La grafica de los pronósticos correspondientes a los 10 modelos ARIMA que se contemplaron están en la figura 3.17.

3.2.2 Modelo de Redes Neuronales

Como se mencionó en el marco teórico, las redes tipo MLP no son las más apropiadas para pronosticar series de tiempo, sin embargo, en el momento de la elaboración de este documento la mejor aproximación que se logró a la programación de redes neuronales fue a través de la función *mlp* del paquete *nnfor* que ajusta un modelo

Semana	Pronóstico	Lo.80	Hi.80	Lo.95	Hi.95
1	66.15	63.67	68.63	62.36	69.94
2	66.89	63.55	70.23	61.78	72.00
3	65.99	61.77	70.21	59.54	72.44
4	66.92	62.05	71.78	59.47	74.36
5	67.08	61.66	72.50	58.79	75.36
6	68.06	62.21	73.90	59.12	77.00
7	68.24	62.00	74.49	58.70	77.79
8	69.19	62.55	75.83	59.04	79.34
9	70.17	63.11	77.23	59.38	80.97
10	71.53	64.05	79.01	60.09	82.97
11	71.31	63.43	79.18	59.27	83.34
12	69.52	61.31	77.73	56.97	82.07
13	69.79	61.29	78.29	56.79	82.79
14	69.20	60.42	77.99	55.77	82.63
15	70.36	61.29	79.44	56.48	84.24

Cuadro 3.1: Pronósticos con el modelo ARIMA

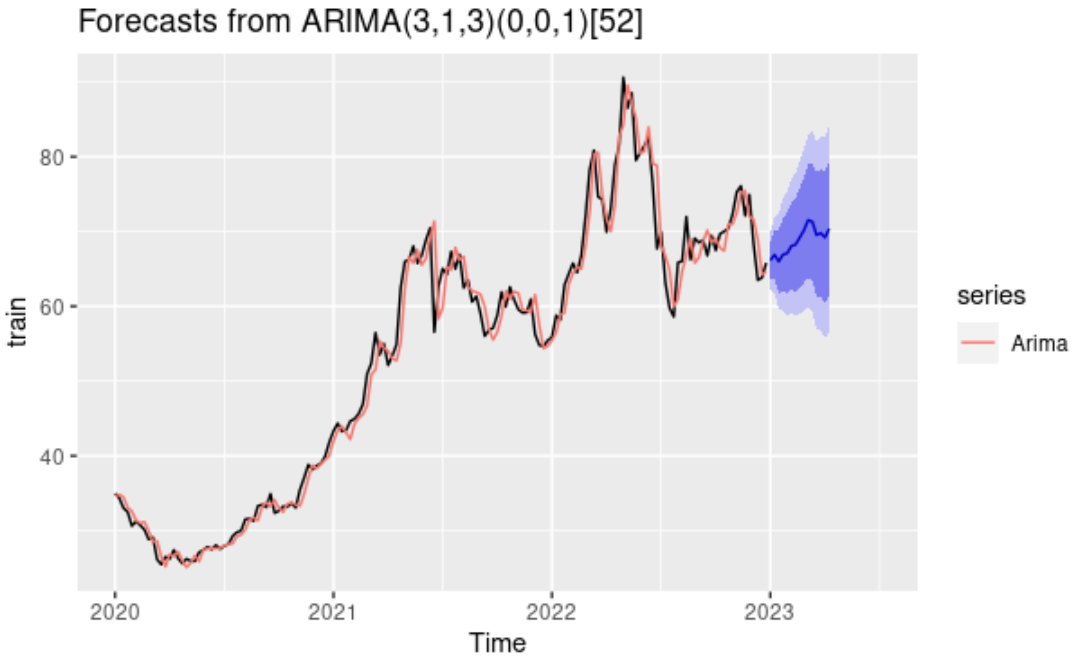


Figura 3.16: Pronóstico con el modelo ARIMA

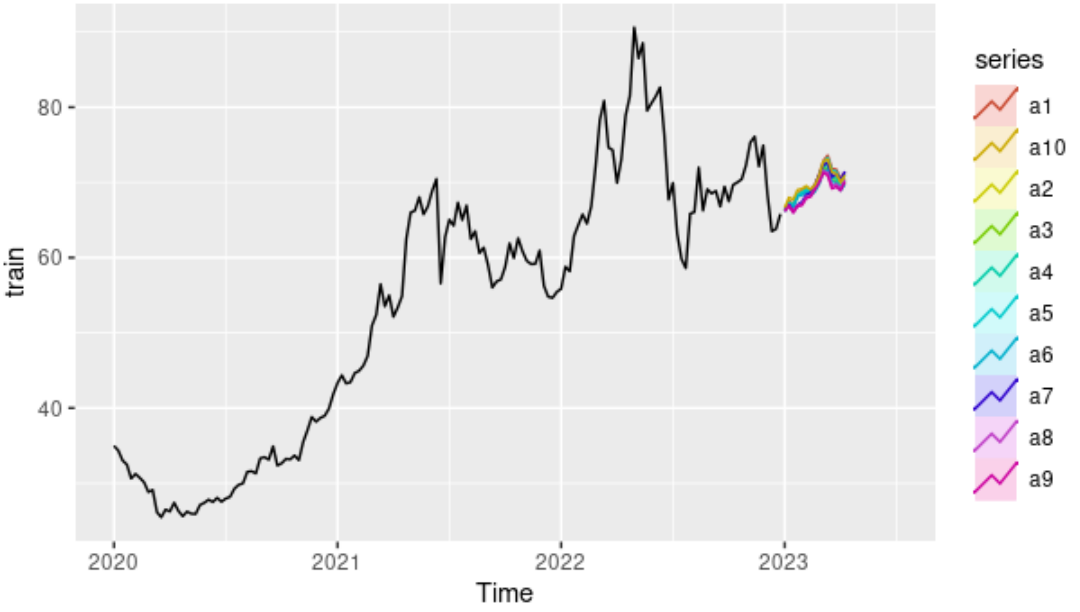


Figura 3.17: Pronósticos de los 10 modelos ARIMA contemplados

de redes neuronales tipo perceptrón multicapa al pronóstico de series de tiempo. El algoritmo de *mlp* determina cuantos rezagos autorregresivos deben ser usados como predictores. Por defecto trae una sola capa oculta con 5 nodos en ella y 20 nodos de entrada.

Una red generada para nuestra serie es la de la figura 3.18 y su arquitectura está en la figura 3.19. Los pronósticos generados por esta red se encuentran en la figura 3.20 y el cuadro 3.2.

```
MLP fit with 5 hidden nodes and 20 repetitions.
Series modelled in differences: D1.
Univariate lags: (3,7,9,16,18,19,20,25,27,28,29,35,37,38,41,44,46,48,49,51,52)
Forecast combined using the median operator.
MSE: 0.4152.
```

Figura 3.18: Red generada por *mlp*

El análisis de los residuos se encuentra en la figura 3.21. En ella se puede ver que estos no son aleatorios, a pesar de esto fue el mejor modelo de redes que se logró ajustar.

Para no quedarse solo con ese modelo puntual se generaron 10 redes y sus pronósticos se encuentran en la figura 3.22. Suponiendo que existe un patrón general se tomó el promedio de estos como representante de pronóstico de las redes neuronales y el conjunto de pronósticos sirve como una especie de intervalo de confianza. El grafico

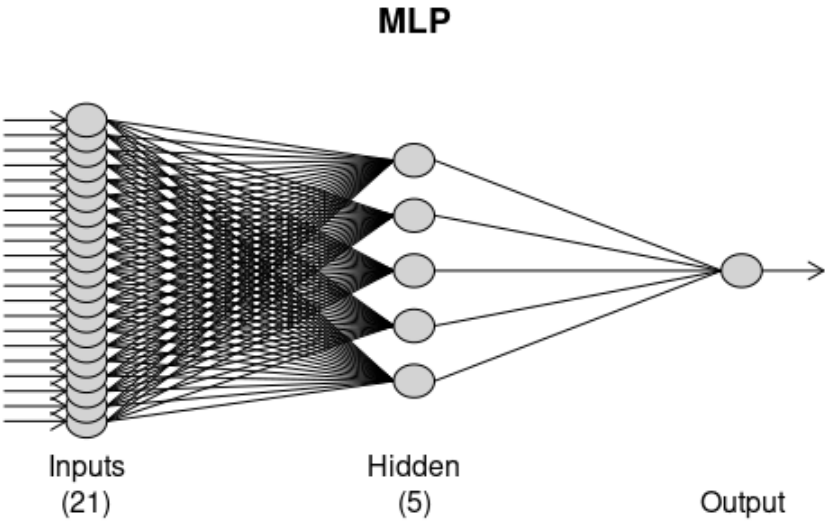


Figura 3.19: Arquitectura de la red generada por *mlp*

se encuentra en la figura 3.23 y en el cuadro 3.3.

3.2.3 Modelo Holt-Winters

Una vez obtenido los dos modelos anteriores se optó por incluir un método clásico que permitiera la comparación con los otros. Se eligieron los modelos de suavizado exponencial estacional de Holt-Winters. En este caso corresponde usar su versión multiplicativa ya que la variación de la serie va en aumento.

El paquete *stast* ofrece la función *HoltWinters* que ajusta un modelo Holt Winters a la serie que se le entrega. El modelo ajustado obtenido para la serie de estudio se encuentra en la figura 3.24. Los pronósticos a partir de él se encuentran en el cuadro 3.4 y el grafico 3.25.

El análisis de residuos se encuentra en la figura 3.26. En el se puede ver que no son completamente aleatorios, sin embargo fue el mejor modelo Holt-Winters que se logro obtener.

3.3 Comparación de modelos

En el cuadro 3.5 y en la figura 3.27 están plasmados los pronósticos con los tres modelos que se ajustaron.

Semana	Pronóstico
1	62.50
2	58.15
3	53.75
4	57.22
5	54.25
6	56.69
7	50.14
8	48.44
9	50.22
10	52.16
11	51.60
12	55.87
13	53.01
14	49.98
15	51.80

Cuadro 3.2: Pronósticos con la red generada por *mlp*

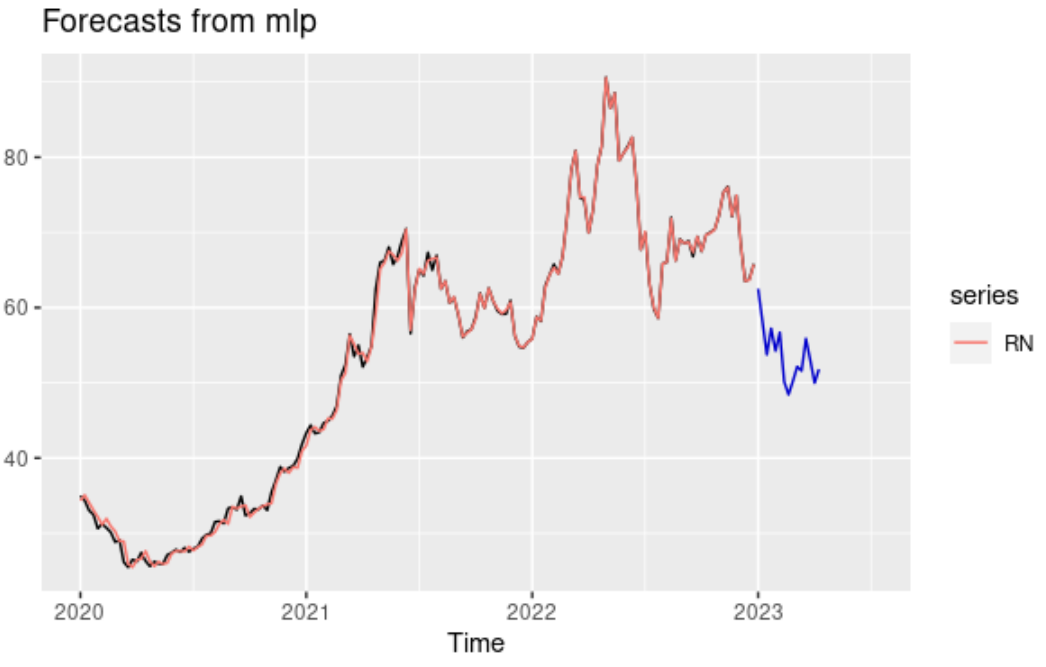


Figura 3.20: Pronósticos con la red generada por *mlp*

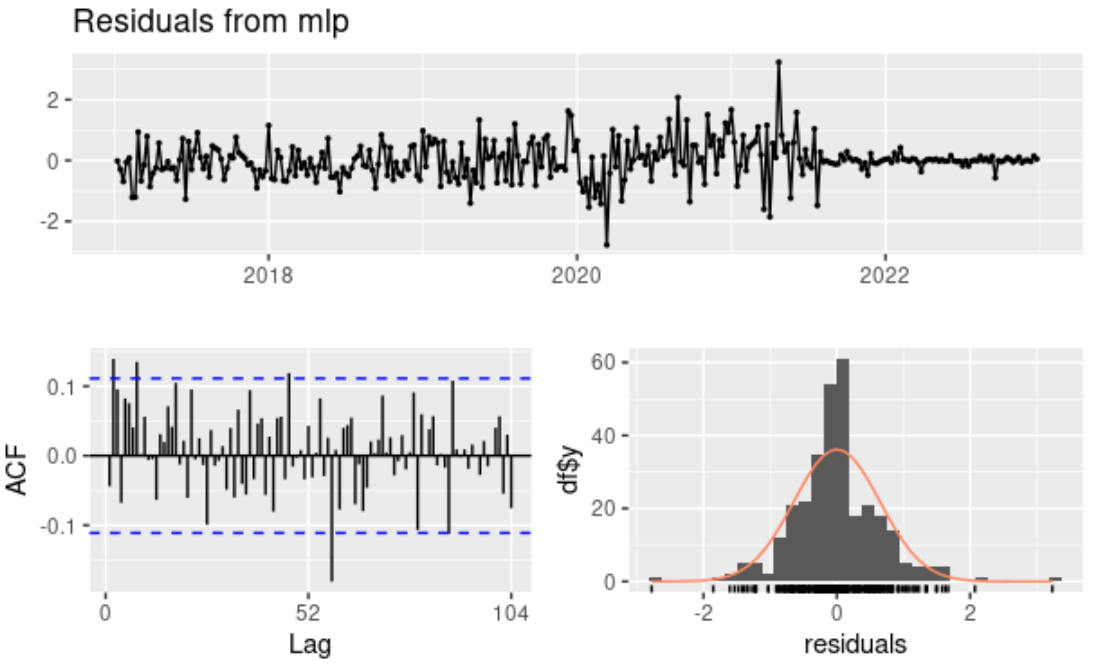


Figura 3.21: Análisis de residuos la red generada por *mlp*

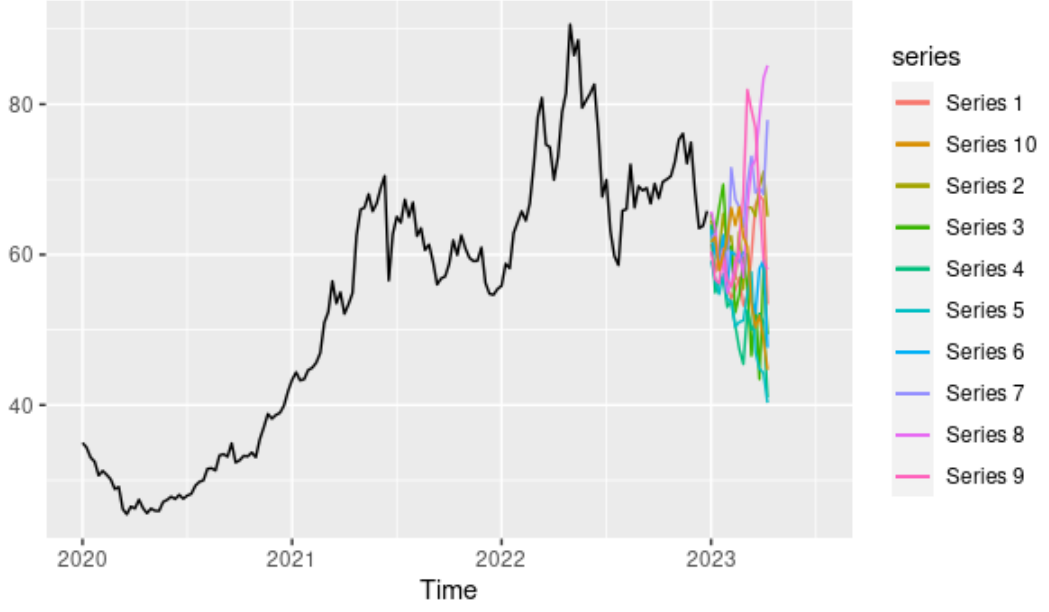


Figura 3.22: Pronósticos de 10 redes generadas por *mlp*

Semana	Pronóstico
1	62.80
2	59.83
3	58.81
4	61.31
5	57.26
6	59.53
7	57.30
8	58.72
9	57.62
10	62.27
11	61.15
12	60.32
13	60.20
14	61.08
15	56.25

Cuadro 3.3: Promedio de los pronósticos de las 10 redes

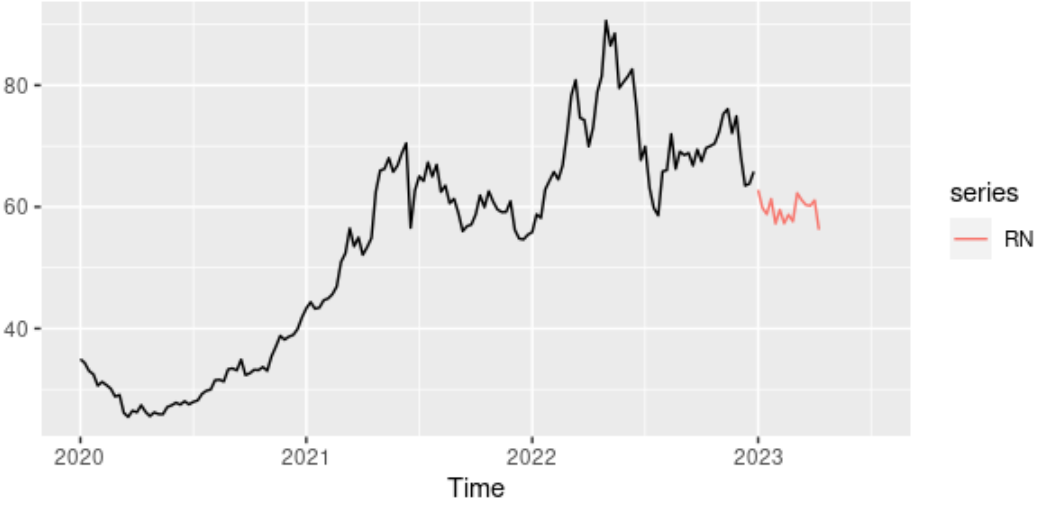


Figura 3.23: Promedio de los pronósticos de las 10 redes

```
Holt-Winters exponential smoothing with trend and multiplicative seasonal component.

Call:
HoltWinters(x = train, seasonal = "multiplicative")

Smoothing parameters:
 alpha: 0.8801211
 beta : 0
 gamma: 1

Coefficients:
           [,1]
a   61.45092715    s11  0.97684592    s23  0.95458478    s35  0.99423044    s47  1.06573578
b    0.01120482    s12  0.96061237    s24  0.97080545    s36  0.98374386    s48  1.11193262
s1   1.06735076    s13  0.96720967    s25  0.92818218    s37  0.96855290    s49  1.09927081
s2   1.07802074    s14  0.93681863    s26  0.95091093    s38  0.99596880    s50  1.10604910
s3   1.06323514    s15  0.94439236    s27  0.94759787    s39  0.99621487    s51  1.09075354
s4   1.06058322    s16  0.95147301    s28  0.92939670    s40  0.99953933    s52  1.07077315
s5   1.04630881    s17  0.95461477    s29  0.94297587    s41  1.01064342
s6   1.04550430    s18  0.97332768    s30  0.92010032    s42  1.03002130
s7   1.01531627    s19  0.95524024    s31  0.93257123    s43  1.05961434
s8   0.99464726    s20  0.97267209    s32  0.94064895    s44  1.05887225
s9   1.01102819    s21  0.94985071    s33  0.99621634    s45  1.06694370
s10  1.00280407    s22  0.95077113    s34  0.98211854    s46  1.04207666
```

Figura 3.24: Modelo Holt-Winters

Gráficamente no queda claro cual modelo fue mejor. Se analizó entonces el cuadro 3.6 de análisis de medidas de error.

Aquí se puede notar que, de las cinco medidas de error, el modelo Holt-Winters fue mejor en tres de ellas, mientras que el modelo de redes neuronales fue mejor en las otras dos. Se puede concluir que ambos son buenos para pronosticar la serie. Se elige el modelo Holt-Winters debido a su complejidad computacional, ya que el tiempo de ejecución es menor que el de las redes neuronales.

3.4 Pronóstico

El modelo de Holt-Wintes ajustado a toda la serie completa está en la figura 3.28. El pronóstico de las siguientes 15 semanas se encuentra en el cuadro 3.7 y en el gráfico 3.29.

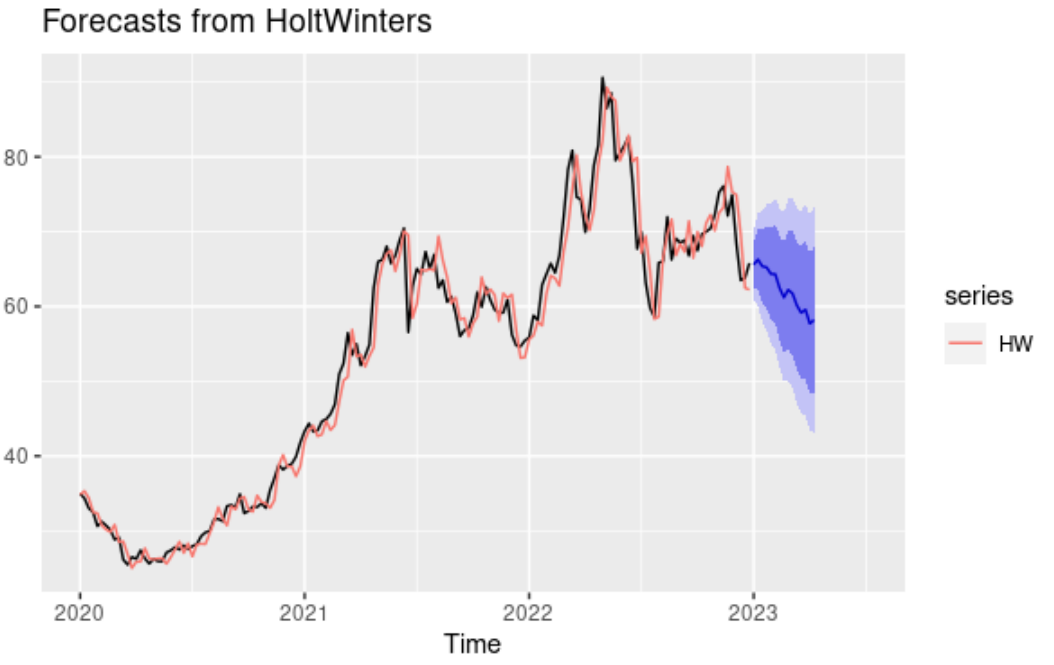


Figura 3.25: Pronoótico con el modelo Holt Winters

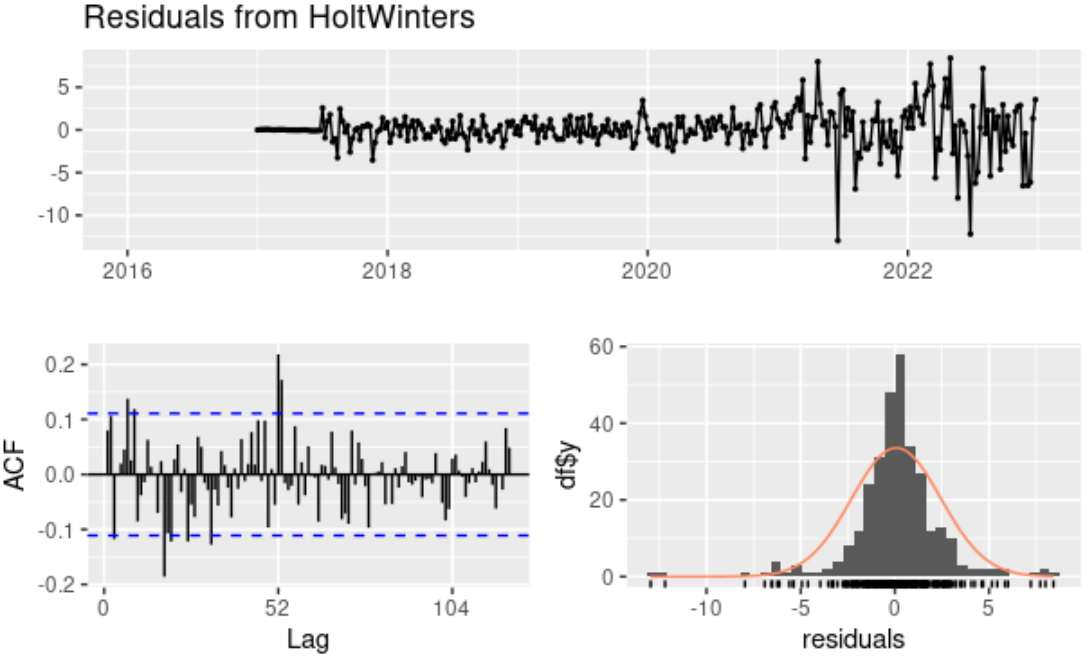


Figura 3.26: Análsis de residuos modelo Holt Winters

Semana	Pronóstico	Lo.80	Hi.80	Lo.95	Hi.95
1	65.60	62.56	68.64	60.95	70.25
2	66.27	62.20	70.34	60.05	72.49
3	65.37	60.54	70.20	57.99	72.76
4	65.22	59.71	70.73	56.79	73.65
5	64.36	58.29	70.42	55.08	73.63
6	64.32	57.69	70.94	54.18	74.45
7	62.47	55.49	69.45	51.80	73.14
8	61.21	53.86	68.56	49.98	72.45
9	62.23	54.30	70.16	50.10	74.36
10	61.74	53.43	70.04	49.03	74.44
11	60.15	51.62	68.68	47.10	73.20
12	59.16	50.35	67.97	45.69	72.63
13	59.58	50.31	68.84	45.41	73.74
14	57.72	48.34	67.09	43.38	72.05
15	58.19	48.38	68.01	43.18	73.21

Cuadro 3.4: Pronóstico con el modelo Holt-Winters

Semana	ARIMA	RN	HW	REAL
1	66.15	62.80	65.60	66.40
2	66.89	59.83	66.27	62.94
3	65.99	58.81	65.37	63.25
4	66.92	61.31	65.22	63.15
5	67.08	57.26	64.36	60.79
6	68.06	59.53	64.32	60.94
7	68.24	57.30	62.47	59.04
8	69.19	58.72	61.21	61.90
9	70.17	57.62	62.23	62.04
10	71.53	62.27	61.74	61.39
11	71.31	61.15	60.15	56.57
12	69.52	60.32	59.16	57.73
13	69.79	60.20	59.58	52.17
14	69.20	61.08	57.72	54.37
15	70.36	56.25	58.19	55.49

Cuadro 3.5: Comparación pronósticos vs datos reales

Medidas	ARIMA	RN	HW
ME	-8.82	0.25	-2.36
RMSE	10.14	3.93	3.11
MAE	8.85	3.39	2.56
MPE	-15.30	0.03	-4.11
MAPE	15.35	5.78	4.42

Cuadro 3.6: Errores

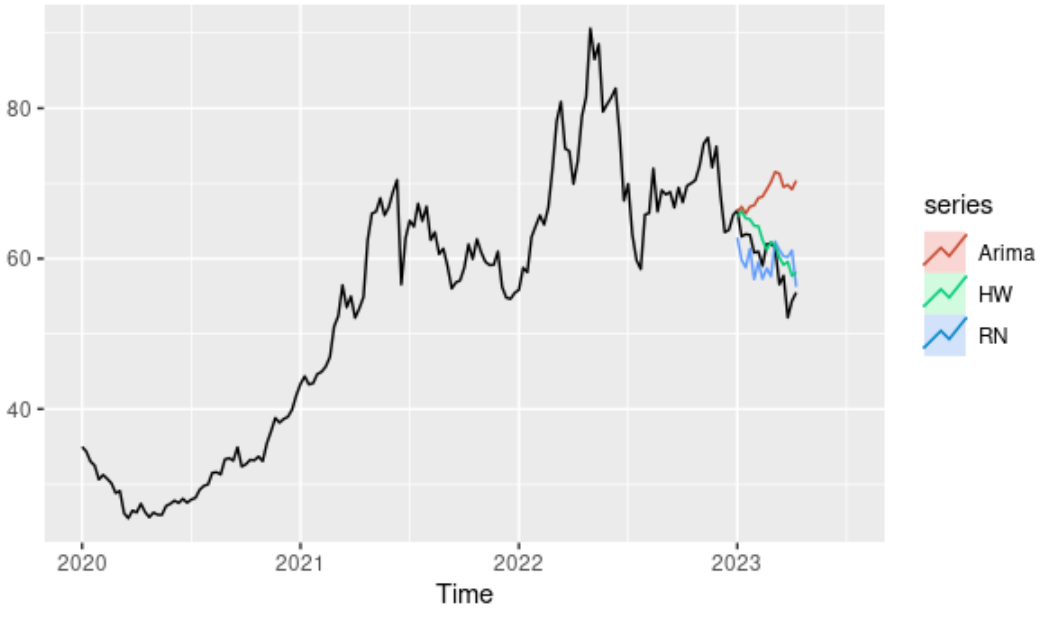


Figura 3.27: Comparacion pronósticos vs datos reales

```
Holt-Winters exponential smoothing with trend and multiplicative seasonal component.

Call:
HoltWinters(x = soja.ts, seasonal = "multiplicative")

Smoothing parameters:
 alpha: 0.8798101
 beta : 0
 gamma: 1

Coefficients:
           [,1]
a   58.62124572    s11  0.95084081    s23  0.99592566    s35  1.10600577    s47  1.00244936
b    0.01120482    s12  0.94765555    s24  0.99620257    s36  1.09082589    s48  0.97015750
s1   0.95149066    s13  0.92937753    s25  0.99953152    s37  1.07086284    s49  0.96398837
s2   0.95463718    s14  0.94296181    s26  1.01066802    s38  1.06893824    s50  0.95454299
s3   0.97335404    s15  0.92014382    s27  1.02997132    s39  1.06978478    s51  0.94340934
s4   0.95522100    s16  0.93262177    s28  1.05958641    s40  1.06460563    s52  0.94658517
s5   0.97267481    s17  0.94061876    s29  1.05885860    s41  1.06088246
s6   0.94983029    s18  0.99614751    s30  1.06696664    s42  1.04323513
s7   0.95080497    s19  0.98204934    s31  1.04213141    s43  1.04553764
s8   0.95460065    s20  0.99426641    s32  1.06566634    s44  1.01503264
s9   0.97078440    s21  0.98374738    s33  1.11185553    s45  1.00255904
s10  0.92811896    s22  0.96856156    s34  1.09918127    s46  1.01027901
```

Figura 3.28: Modelo Holt-Winters ajustado a la serie completa

Semana	Pronóstico	Lo.80	Hi.80	Lo.95	Hi.95
1	55.79	52.74	58.84	51.12	60.45
2	55.98	51.92	60.05	49.76	62.20
3	57.09	52.16	62.02	49.55	64.63
4	56.04	50.50	61.58	47.57	64.51
5	57.07	50.83	63.31	47.53	66.62
6	55.74	49.08	62.41	45.55	65.94
7	55.81	48.62	63.00	44.81	66.81
8	56.05	48.34	63.75	44.27	67.82
9	57.01	48.73	65.28	44.35	69.66
10	54.51	46.15	62.88	41.72	67.30
11	55.86	46.88	64.83	42.13	69.58
12	55.68	46.34	65.02	41.40	69.96
13	54.62	45.07	64.16	40.02	69.22
14	55.43	45.38	65.47	40.06	70.79
15	54.09	43.92	64.27	38.54	69.65

Cuadro 3.7: Pronóstico con el modelo Holt-Winters

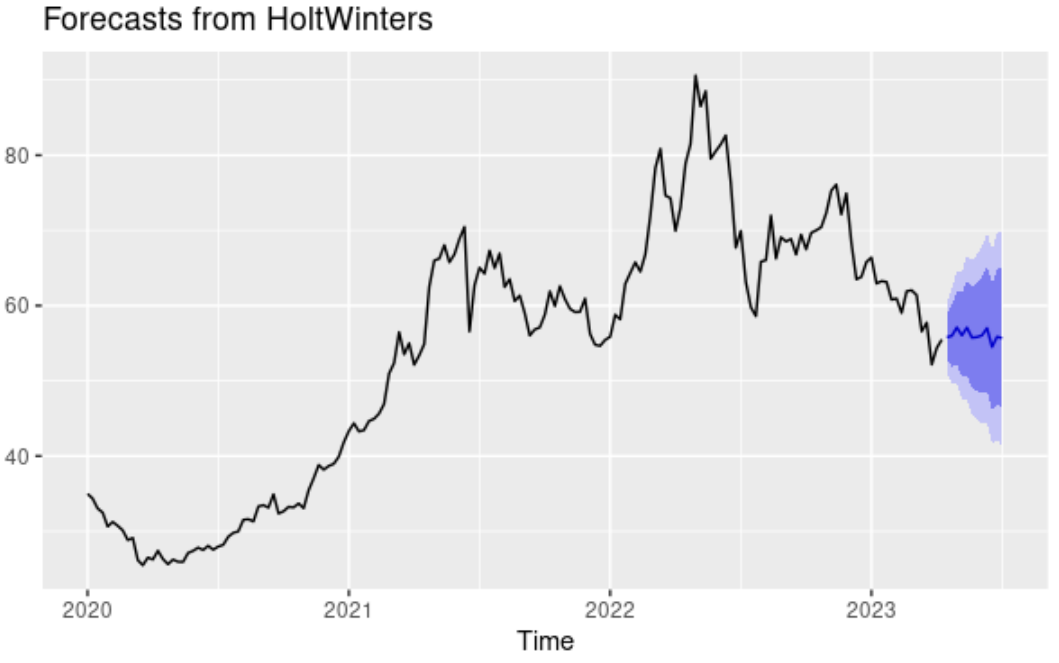


Figura 3.29: Pronóstico con el modelo Holt-Winters

Capítulo 4

Conclusiones

En este trabajo se investigo acerca de las técnicas de pronósticos de series de tiempo más utilizadas en la actualidad para contrastarlas con las técnicas clásicas. Se usaron redes neuronales, a pesar de que no se utilizó la arquitectura más apropiada debido a la falta de experiencia programando redes neuronales.

Para cada familia de modelos se elige un representante, siendo este el que mejor desempeño haya tenido en cada caso. Una vez identificados los tres modelos candidatos se realiza un análisis comparativo de medidas de error. Se concluye que tanto el modelo Holt-Winters como el de Redes Neuronales tuvieron pronósticos muy acertados. Para los pronósticos se optó por el modelo Holt-Winters debido a su sencilles y velocidad de ejecución.

Para la generación de los distintos modelos y los pronósticos se utilizó el lenguaje de programación R y el entorno RStudio ya que este es uno de los más utilizados en temas estadísticos.

Finalmente, se construyó el tablero de monitoreo y pronóstico [13] para facilitar la comunicación e interpretación de los resultados. El cual se alimenta de los nuevos datos a medida que estos se van generando y decide que modelo es mejor cada vez que se actualiza.

Ahora bien, para próximas investigaciones se propone el desarrollo de redes neuronales recurrentes del tipo LSTM (Long Short Term Memory), las cuales podrían mejorar la precisión de los pronósticos, aunque se requiere de un amplio manejo del tema para programarlas correctamente. También podría ser interesante intentar predecir el comportamiento de nuestra serie basándose en su correlación con otras series como la del precio del petróleo, la del precio del dólar estadounidense, entre otras. Además, sería interesante intentar replicar el estudio con otro tipo de series de precios de materias primas y determinar si los resultados son similares

Apéndice A

Códigos

A continuación anexamos el código utilizado para el desarrollo del proyecto.

A.1 Paquetes utilizados

```
library(quantmod) #para importar la serie
library(ggplot2)
library(forecast)
library(tseries) #pruebas DF y FNSS
library(xtable) #tablas en latex
library(nnfor) #redes neuronales
```

A.2 Importación de la serie desde Yahoo Finanzas

```
soja.df=getSymbols("ZL=F",
                   from="2016-01-01",
                   to="2023-04-01",
                   env=NULL,
                   periodicity ="weekly",
                   return.class="data.frame")[6]

soja.df$Fecha=as.Date(row.names(soja.df),format="%Y-%m-%d")
row.names(soja.df)=NULL
names(soja.df)[1]= "Precio"

sum(is.na(soja.df$Precio))
soja.df=na.omit(soja.df)
```

A.3 Análisis de la serie

```
ggplot(soja.df,aes(Fecha,Precio))+
  geom_line()+
   ggtitle("Precios de Futuros de Aceite de Soja")+
   xlab("")+
   ylab("")+
   scale_x_date(date_breaks = "1 year", date_labels = "%Y")

soja.ts=ts(soja.df$Precio,start = c(2016,1),frequency = 52)

forecast::autoplot(decompose(soja.ts,type = "multiplicative")
    )

ggseasonplot(soja.ts)


train=window(soja.ts,end=c(2022,52))
test=window(soja.ts,start=c(2023,1))
n=length(test)
```

A.4 Estimación de modelos

A.4.1 ARIMA

```
#ARIMA

Acf(train)
ndiffs(train)
nsdiffs(train)

forecast::autoplot(diff(train))+
   ggtitle("Serie diferenciada del precio de Futuros de Aceite
       de Soja")+
   xlab("")+
   ylab("")

adf=adf.test(diff(train))
adf$p.value

kps=kpss.test(diff(train))
kps$p.value

Acf(diff(train))
Pacf(diff(train))
```

```
20
21 a1=Arima(train,order=c(0,1,3),seasonal=c(1,0,0))
22 a2=Arima(train,order=c(0,1,3),seasonal=c(0,0,1))
23 a3=Arima(train,order=c(0,1,3),seasonal=c(1,0,1))
24
25 a4=Arima(train,order=c(3,1,0),seasonal=c(0,0,1))
26 a5=Arima(train,order=c(3,1,0),seasonal=c(1,0,0))
27 a6=Arima(train,order=c(3,1,0),seasonal=c(1,0,1))
28
29 a7=Arima(train,order=c(3,1,3),seasonal=c(1,0,0))
30 a8=Arima(train,order=c(3,1,3),seasonal=c(0,0,1))
31 a9=Arima(train,order=c(3,1,3),seasonal=c(1,0,1))
32
33 AIC(a1,a2,a3,a4,a5,a6,a7,a8,a9)
34 BIC(a1,a2,a3,a4,a5,a6,a7,a8,a9)
35
36
37 a10=auto.arima(train,
38                trace=TRUE,
39                approximation=FALSE,
40                stepwise=FALSE,
41                allowdrift = FALSE,
42                max.P = 1,
43                max.Q = 1)
44
45 checkresiduals(a8)
46
47
48 qqnorm(a8$residuals)
49 qqline(a8$residuals)
50
51 tsdiag(a8)
52 Box.test(a8$residuals,type = "Ljung-Box",lag = 80)
53
54 shapiro.test(a8$residuals)
55
56 jarque.bera.test(a8$residuals)
57
58 autoplot(a8)
59
60
61 m1=forecast::forecast(a8,h=n)
62 forecast::autoplot(m1)+
63   forecast::autolayer(fitted(m1),series="Arima")+
64   xlim(2020,2023.5)
```

```
65
66 p1=data.frame(m1)
67 rownames(p1)=1:n
68 print(xtable(p1))
69
70
71 forecast::autoplot(train)+
72   xlim(2020,2023.5)+
73   forecast::autolayer(forecast::forecast(a1,h=n),series="a1",
       PI=FALSE)+
74   forecast::autolayer(forecast::forecast(a2,h=n),series="a2",
       PI=FALSE)+
75   forecast::autolayer(forecast::forecast(a3,h=n),series="a3",
       PI=FALSE)+
76   forecast::autolayer(forecast::forecast(a4,h=n),series="a4",
       PI=FALSE)+
77   forecast::autolayer(forecast::forecast(a5,h=n),series="a5",
       PI=FALSE)+
78   forecast::autolayer(forecast::forecast(a6,h=n),series="a6",
       PI=FALSE)+
79   forecast::autolayer(forecast::forecast(a7,h=n),series="a7",
       PI=FALSE)+
80   forecast::autolayer(forecast::forecast(a8,h=n),series="a8",
       PI=FALSE)+
81   forecast::autolayer(forecast::forecast(a9,h=n),series="a9",
       PI=FALSE)+
82   forecast::autolayer(forecast::forecast(a10,h=n),series="a10
       ",PI=FALSE)
```

A.4.2 Redes neuronales

```
1  # REDES NEURONALES
2
3  rn=nnfor::mlp(train)
4  print(rn)
5  plot(rn)
6
7
8  forecast::autoplot(forecast::forecast(rn,h=n))+
9    forecast::autolayer(rn$fitted,series="RN")+
10   xlim(2020,2023.5)+
11   ylab("")
12
13 checkresiduals(forecast::forecast(rn,h=n))
```

```
14
15 p.rn=data.frame(forecast::forecast(rn,h=n))
16 rownames(p.rn)=1:n
17 print(xtable(p.rn))
18
19
20 sim <- ts(matrix(0, nrow=n, ncol=10), start=c(2023,1),
       frequency = 52)
21 for(i in 1:10)
22   sim[,i] <- forecast::forecast(nnfor::mlp(train),h=n)$mean
23
24 forecast::autoplot(train)+
25   forecast::autolayer(sim)+
26   xlim(2020,2023.5)+
27   ylab("")
28
29 m2=ts(rowMeans(sim),start = c(2023,1),frequency = 52)
30
31 forecast::autoplot(train) +
32   forecast::autolayer(m2, series = "RN")+
33   xlim(2020,2023.5)+
34   ylab("")
35
36 p2=data.frame(m2)
37 rownames(p2)=1:n
38 print(xtable(p2))
```

A.4.3 Holt-Winters

```
1 #HOLT WINTERS
2
3 hw=HoltWinters(train,seasonal = "multiplicative")
4 m3=forecast::forecast(hw,h=n)
5
6 forecast::autoplot(m3)+
7   forecast::autolayer(fitted(m3),series="HW")+
8   xlim(2020,2023.5)+
9   ylab("")
10
11 checkresiduals(m3)
12
13 p3=data.frame(m3)
14 rownames(p3)=1:n
15 print(xtable(p3))
```

A.5 Comparación de modelos

```
#COMPARATIVO

forecast::autoplot(soja.ts)+
  forecast::autolayer(m1, series="Arima",PI=FALSE)+
  forecast::autolayer(m2, series="RN",PI=FALSE)+
  forecast::autolayer(m3, series="HW",PI=FALSE)+
  xlim(2020,2023.5)+
  ylab("")

comp=data.frame(p1[1],p2[1],p3[1],data.frame(test))
rownames(comp)=1:n
print(xtable(comp))

ac1=data.frame(accuracy(m1,test))
ac2=data.frame(accuracy(m2,test))
ac3=data.frame(accuracy(m3,test))

errores=data.frame(
  "ARIMA"=t(accuracy(m1,test))[1:5,2],
  "RN"=t(accuracy(m2,test))[1:5],
  "HW"=t(accuracy(m3,test))[1:5,2])

print(xtable(errores))
```

A.6 Pronóstico

```
#PRONOSTICO

mod=HoltWinters(soja.ts,seasonal = "multiplicative")

pron=forecast::forecast(mod,h=n)

forecast::autoplot(pron)+
  xlim(2020,2023.5)+
  ylab("")

pron.df=data.frame(pron)
rownames(pron.df)=1:n
print(xtable(pron.df))
```

A.7 Dashboard

```
---
title: "Tablero de Pronostico"
output:
  flexdashboard::flex_dashboard:
  orientation: columns
  vertical_layout: fill
  source_code: embed
  theme: flatly
---

```{r setup, include=FALSE}

#install.packages("quantmod")
#install.packages("ggplot2")
#install.packages("forecast")
#install.packages("nnfor")

library(flexdashboard)
library(quantmod) #para importar la serie
library(ggplot2)
library(forecast)
library(nnfor) #redes neuronales

soja.df=getSymbols("ZL=F",
                   from="2016-01-01",
                   to="2023-04-01",
                   env=NULL,
                   periodicity ="weekly",
                   return.class="data.frame")

soja.df$Fecha=as.Date(row.names(soja.df),format="%Y-%m-%d")
row.names(soja.df)=NULL
names(soja.df)= c("Apertura","Maximo","Minimo","Cierre","
    Volumen","Ajustado","Fecha")

sum(is.na(soja.df))
soja.df=na.omit(soja.df)
dim(soja.df)
soja.df[dim(soja.df)[1],1]

soja.ts=ts(soja.df$Ajustado,start = c(2016,1),frequency = 52)

train=window(soja.ts,end=c(2022,52))
```
```

```
43 test=window(soja.ts,start=c(2023,1))
44 n=length(test)
45
46
47
48 ```
49
50 ZLK
51 ==========================================================================

52 Column {data-width=50}
53 --------------------------------------------------------------------------

54 ### Fecha
55 ```{r}
56 valueBox(value = soja.df[dim(soja.df)[1],]$Fecha,
57          caption = "Ultima actualizacion",
58          icon = "fa-line-chart",
59          color = "#3cb371")
60 ```
61
62
63 ### Apertura
64 ```{r}
65 valueBox(value = round(soja.df[dim(soja.df)[1],]$Apertura,2),
66          caption = "Apertura",
67          icon = "fa-line-chart",
68          color = "#3cb371")
69 ```
70
71
72 ### Cierre
73 ```{r}
74 valueBox(value = round(soja.df[dim(soja.df)[1],]$Cierre,2),
75          caption = "Cierre",
76          icon = "fa-line-chart",
77          color = "#3cb371")
78 ```
79
80
81 ### Minimo
82 ```{r}
83 valueBox(value = round(soja.df[dim(soja.df)[1],]$Minimo,2),
84          caption = "Minimo",
85          icon = "fa-line-chart",
```

```
            color = "#3cb371")
```
```


### Maximo
```{r}
valueBox(value = round(soja.df[dim(soja.df)[1],]$Maximo,2),
            caption = "Maximo",
            icon = "fa-line-chart",
            color = "#3cb371")
```
```

Volumen
```{r}
valueBox(value = paste(soja.df[dim(soja.df)[1],]$Volumen,"K")
    ,
            caption = "Volumen",
            icon = "fa-line-chart",
            color = "#3cb371")
```
```



Column {data-width=950}
-----------------------------------------------------------------------

```{r, fig.width=10, fig.height=6}
ggplot(soja.df,aes(Fecha,Ajustado))+
  geom_line(color = "#FC4E07")+
   ggtitle("Precios de Futuros de Aceite de Soja")+
   xlab("")+
   ylab("")+
   scale_x_date(date_breaks = "1 year", date_labels = "%Y")
```
```

Descomposicion
===

Column

```{r, fig.width=7, fig.height=6}
```

```
forecast::autoplot(decompose(soja.ts,type = "multiplicative")
    )
```

```
Column
-------------------------------------------------------------------------------

```{r,fig.width=7, fig.height=6}
ggseasonplot(soja.ts)
```


Comparacion
===

Column {data-width=700, .tabset}

ARIMA
```{r,fig.width=10, fig.height=6}
am=auto.arima(train,
                  stepwise=FALSE,
                  allowdrift = FALSE)
m1=forecast::forecast(am,h=n)

forecast::autoplot(m1)+
  forecast::autolayer(fitted(m1),series="ARIMA")+
  xlim(2020,2023.5)+
  ylab("")

```


RN
```{r,fig.width=10, fig.height=6}
rn=nnfor::mlp(train)
m2=forecast::forecast(rn,h=n)

forecast::autoplot(m2)+
  forecast::autolayer(rn$fitted,series="RN")+
  xlim(2020,2023.5)+
  ylab("")

```
```

````
167
168 ### HW
169 ```{r, fig.width=10, fig.height=6}
170 hw=HoltWinters(train,seasonal = "multiplicative")
171 m3=forecast::forecast(hw,h=n)
172
173 forecast::autoplot(m3)+
174   forecast::autolayer(fitted(m3),series="HW")+
175   xlim(2020,2023.5)+
176   ylab("")
177
178 ```
179
180 ### Comparacion
181 ```{r, fig.width=10, fig.height=6}
182 forecast::autoplot(soja.ts)+
183   forecast::autolayer(m1,series="Arima",PI=FALSE)+
184   forecast::autolayer(m2,series="RN",PI=FALSE)+
185   forecast::autolayer(m3,series="HW",PI=FALSE)+
186   xlim(2020,2023.5)+
187   ylab("")
188
189 ```
190
191
192 Column {data-width=300}
193 -----------------------------------------------------------------------

194 ### Ajustes
195 ```{r}
196 comp=round(data.frame(data.frame(m1)[1],data.frame(m2)[1],
      data.frame(m3)[1],data.frame(test)),2)
197 rownames(comp)=paste("Semana",1:n)
198 names(comp)=c("ARIMA","RN","HW","REAL")
199
200 knitr::kable(comp)
201 ```
202
203
204 ### Errores
205 ```{r}
206 errores=round(data.frame(
207   "ARIMA"=t(accuracy(m1,test))[1:5,2],
208   "RN"=t(accuracy(m2,test))[1:5,2],
209   "HW"=t(accuracy(m3,test))[1:5,2]),2)
````

```
210
211 knitr::kable(errores)
212 ```
213
214
215
216 Pronostico
217 ==========================================================================

218 Column {data-width=700}
219 --------------------------------------------------------------------------

220 ### Pronostico
221 ```{r, fig.width=10, fig.height=6}
222 mod=HoltWinters(soja.ts, seasonal = "multiplicative")
223 pron=forecast::forecast(mod,h=n)
224
225 forecast::autoplot(pron)+
226   forecast::autolayer(fitted(pron),series="HW")+
227   xlim(2020,2023.5)+
228   ylab("")
229
230 ```
231
232
233 Column {data-width=300}
234 --------------------------------------------------------------------------

235 ### Pronostico
236 ```{r}
237 pron.df=round(data.frame(pron),2)
238 rownames(pron.df)=paste("Semana",(n+1):(2*n))
239
240 knitr::kable(pron.df)
241 ```
```

Bibliografía

[1] Hanke, J. E., & Wichern, D. W. (2010). Pronósticos en los negocios 9na edición.

[2] Delgado, A. (1998). Inteligencia artificial y minirobots.

[3] Konasani, V. R., & Kadre, S. (2021). Machine learning and deep learning using python and tensorflow. McGraw-Hill Education.

[4] Gopal, M. (2019). Applied machine learning. McGraw-Hill Education.

[5] Lara-Benítez, P., Carranza-García, M., & Riquelme, J. C. (2021). An experimental review on deep learning architectures for time series forecasting. International journal of neural systems, 31(03), 2130001.

[6] Chen, Z., Goh, H. S., Sin, K. L., Lim, K., Chung, N. K. H., & Liew, X. Y. (2021). Automated agriculture commodity price prediction system with machine learning techniques. arXiv preprint arXiv:2106.12747.

[7] Athiyarath, S., Paul, M., & Krishnaswamy, S. (2020). A comparative study and analysis of time series forecasting techniques. SN Computer Science, 1(3), 175.

[8] Mahmoud, A., & Mohammed, A. (2021). A survey on deep learning for time-series forecasting. Machine learning and big data analytics paradigms: analysis, applications and challenges, 365-392.

[9] Bontempi, G., Ben Taieb, S., & Le Borgne, Y. A. (2013). Machine learning strategies for time series forecasting. Business Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, July 15-21, 2012, Tutorial Lectures 2, 62-77.

[10] Villada, F., Muñoz, N., & García-Quintero, E. (2016). Redes neuronales artificiales aplicadas a la predicción del precio del oro. Información tecnológica, 27(5), 143-150.

[11] Villada, F., Arroyave, D., & Villada, M. (2014). Pronóstico del precio del petróleo mediante redes neuronales artificiales. Información tecnológica, 25(3), 145-154.

[12] Hyndman, RJ & Athanasopoulos, G. (2018) Pronósticos: principios y práctica , 2.ª edición, OTextos: Melbourne, Australia. OTexts.com/fpp2.

[13] Tablero de pronóstico, Recuperado de: https://rpubs.com/juan031899/1028817, 2023