UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CIENCIAS

Predicción de la volatilidad de la tasa de cambio USD/MXN mediante modelos híbridos de series de tiempo heterocedásticos y redes neuronales

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Actuario

PRESENTA:

Jesús Alcántar Martínez

DIRECTORA DE TESIS

Dra. Lizbeth Naranjo Albarrán

Ciudad Universitaria, Cd. Mx., 2019

1. Datos del alumno
   Alcántar
   Martínez
   Jesús
   5534741150
   Universidad Nacional Autónoma de México
   Facultad de Ciencias
   Actuaría
   313172454

2. Datos del tutor
   Dra.
   Lizbeth
   Naranjo
   Albarrán

3. Datos del sinodal 1
   Dra.
   Ruth Selene
   Fuentes
   García

4. Datos del sinodal 2
   M. en C.
   Alma Rosa
   Bustamante
   García

5. Datos del sinodal 3
   Act.
   Miguel Ángel
   Chong
   Rodríguez

6. Datos del sinodal 4
   Act.
   Eduardo Selim
   Martínez
   Mayorga

7. Datos del trabajo escrito
   Predicción de la volatilidad de la tasa de cambio USD/MXN mediante modelos híbridos de series de tiempo heterocedásticos y redes neuronales
   99 p.
   2019

“El dolor es temporal;
los logros, permanentes...”

A mi padre, mis hermanos y, principalmente,
a mi madre.

Agradecimientos

Quiero agredecer principalmente a mis padres y hermanos, por todo el apoyo y cariño incondicional que me han proporcionado. A lo largo de mi vida, han estado siempre conmigo, motivándome a cumplir mis sueños y gracias a ellos es que he conseguido cumplir varias metas.

A mi madre, quien siempre ha estado conmigo cuando las cosas no andan bien. Siempre me ha apoyado, cada momento a su lado es muy especial para mí. Le estoy superagradecido.

A mi hermano José Luis no sólo es mi hermano, es también mi mejor amigo. Me ha ayudado muchísimo a crecer como persona y siempre ha estado para aconsejarme y apoyarme.

También quiero agradecer a cada uno de los sinodales: la Doctora Ruth, la Maestra Alma Rosa, el Actuario Miguel Ángel y el Actuario Eduardo Selim, por el tiempo dedicado en la revisión del presente trabajo, así como sus valiosos comentarios. Especialmente quiero agradecer a Eduardo Selim por sus recomendaciones, las cuales fueron de gran ayuda para mí.

A mi maestra de la primaria, Ana María, porque me impulsó para crecer tanto académicamente como de forma personal.

Me gustaría agradecer a la doctora Lizbeth Naranjo, quien fue mi tutora para la realización de la presente tesis. Con ella cursé las tres estadísticas correspondientes al plan de estudios de la carrera de Actuaría y la optativa de Estadística Bayesiana y a lo largo de este tiempo me ha brindado su apoyo, por ejemplo, para solicitar una beca para la escuela de verano de aprendizaje profundo organizada por el CViCom, la cual tuvo lugar en el instituto de matemáticas, Liz me escribió una carta de recomendación académica.

Agradezco a la Facultad de Ciencias y a los profesores, porque varios de ellos me inspiraron a dar lo mejor de mí cada día y el gusto por diversas ramas de la ciencia.

En general, agradezco a la UNAM, por el apoyo, la gran cantidad de recursos y de oportunidades que me brindó. Gracias a la UNAM es que pude realizar este trabajo, me dió una formación académica y pasé muy bellos momentos durante la misma. Gracias a la UNAM conocí a muchas personas a lo largo de mi vida que me fueron complementando como persona y que sin duda alguna me han motivado a ser mejor cada día.

Entre las personas que más marcaron mi vida se encuentran: mis mejores amigos de la preparatoria, María Fernanda (Fer) y Antonio (Toño), y algunos de mis mejores amigos de la facultad: Pame, Renato, Arturo y Andrea (*baby* Sham). Estoy muy agradecido de haber coincidido en esta vida con todos ellos.

Estoy muy agradecido con la vida, y cada una de las personas que he conocido, porque he aprendido algo de cada una de ellas. ¡Gracias!

Índice general

Índice de tablas

Índice de figuras

Introducción

En los últimos años se ha popularizado el *machine learning* o aprendizaje automático pero, ¿qué es *machine learning*? El *machine learning* es un conjunto de algoritmos que permiten la optimización de modelos matemáticos al realizar una tarea en específico. Por ejemplo, para un problema de regresión, la tarea sería la predicción de la variable respuesta a partir de los valores de las variables explicativas.

Dado un modelo estadístico, bajo el enfoque estadístico frecuentista o clásico se busca el vector de parámetros θ^* del modelo a ajustar que maximice la función de verosimilitud $\mathbb{P}(x|\theta)$, mientras que en los modelos bayesianos se obtiene una función de densidad de probabilidad *a posteriori* o final, $\mathbb{P}(\theta|x)$, y el estimador bayesiano de θ se obtiene al calcular $\mathbb{E}_\theta[\theta|x]$[1].

Cada modelo tiene supuestos por lo que depende del problema a resolver y de los datos si un modelo dado será eficiente. Cuando al ajustar los modelos propuestos, alguno de los supuestos no se cumple o la capacidad de generalización de éstos es escasa, una alternativa es el *machine learning*.

Para no profundizar, se tomará como ejemplo el enfoque clásico. En el enfoque clásico, en ocasiones es posible obtener explícitamente la estimación de los parámetros, así como los intervalos de confianza de las predicciones, ya que se hacen supuestos acerca de los datos, entre los cuales se encuentran la distribución de los datos, la cual es conocida y conmúnmente se supone además que las variables aleatorias son idénticamente distribuidas e independientes.

Existe otro enfoque: el de *machine learning* o aprendizaje automático. Como su nombre lo indica, permite que un sistema, a partir de un conjunto de datos, pueda aprender o generalizar a casos no previstos, sin necesidad de realizar una gran cantidad de supuestos como ocurre en muchos modelos estadísticos frecuentistas o bayesianos, por ejemplo, no se requiere conocer la distribución de las variables de entrada del modelo.

A su vez, los algoritmos de entrenamiento en *machine learning* se pueden optimizar y la complejidad de los modelos puede aumentar pero no precisamente por aumentar el número de parámetros a ajustar sino por incrementar la complejidad del diseño del modelo. Al hablar de la complejidad de un modelo se hace referencia tanto a la estructura

[1] Para mayor información acerca del enfoque bayesiano consulte [27].

del modelo, ya sea una regresión lineal, una regresión poisson, etc., como del número de parámetros a ajustar que tiene.

Aumentar la complejidad de un modelo no implica que el modelo resultante tenga un mejor desempeño. Sin embargo, al combinar al menos dos tipos de modelos distintos entre sí, de forma que uno de los modelos se encargue de realizar una tarea en específico y ya sea que el resto transforme los datos de forma que el modelo que realizará la tarea pueda entrenarse de mejor forma, o que proporcionen información adicional al modelo para mejorar las predicciones.

En el presente trabajo se proponen modelos compuestos por al menos dos modelos cuyos parámetros se ajustan bajo enfoques estadísticos distintos, en particular los enfoques frecuentistas y de *machine learning*, y comprobar si este tipo de modelos tienen un mayor desempeño al realizar una tarea en específico a comparación del que tienen cada uno de los modelos por separado, así como su viabilidad debido al costo computacional. A este tipo de modelos se le denominará modelo híbrido, ya que se requiere del ajuste de dos modelos bajo al menos dos enfoque estadísticos distintos. Los modelos ajustados bajo el enfoque frecuentista permitirán obtener información, la cual será utilizada de forma adicional en un modelo que se ajustará bajo el enfoque de *machine learning*.

Poder predecir el valor de la tasa de cambio entre dos divisas cuyo valor depende de la oferta y la demanda permitiría la existencia de arbitraje. Si bien esto no es posible, una alternativa es predecir su comportamiento, es decir, si aumentará o disminuirá en favor de alguna de las dos divisas en particular. Es claro que, a mayor incertidumbre, los riesgos son mayores.

A su vez, es importante considerar el sentimiento del mercado, ya que esto permitirá tener un panorama más amplio sobre si la tasa de cambio aumentará o disminuirá. Predecir momentos de altas y bajas volatilidades puede ser crucial para los inversionistas.

Para valuar opciones, modelos como el de Black-Scholes consideran la volatilidad de un activo denominado subyacente. Sin embargo, para el caso de la tasa de cambio dólar-peso, esta volatilidad no es observable, por lo que primero se debe estimar. Existen muchas metodologías de estimarla. Uno de los principales problemas del modelo clásico de Black-Scholes es el supuesto de que la volatilidad es constante, lo cual no es cierto ya que en ciertos periodos la tasa de cambio dólar-peso fluctúa de forma más agresiva y amplia que en otros. Por ello, se estimó la volatilidad para cada tiempo, luego se realizó un análisis estadístico de las volatilidades estimadas, se propusieron modelos y se prosiguió a ajustarlos, unos bajo el enfoque frecuentista y otros bajo el de *machine learning*.

El primer capítulo aborda el aprendizaje supervisado, una rama del *machine learning*. Se da una introducción del enfoque de *machine learning*. Se presentan metodologías que permiten estimar la eficiencia de un modelo al realizar una tarea en específico las cuales, a su vez, pueden ser utilizadas para verificar la eficiencia de cualquier modelo sin importar el enfoque bajo el cual fueron ajustados sus parámetros.

En el segundo capítulo se da un bosquejo del tema de redes neuronales. La finalidad

es brindar un panorama de los modelos de redes neuronales.

En el tercer capítulo se abarca el tema de las redes neuronales recurrentes. Se presentan los modelos LSTM y GRU, dos tipos de redes que solucionan algunos problemas que presentan las redes recurrentes tradicionales.

En el cuarto capítulo se explican las series de tiempo de heterocedasticidad condicional. Los modelos expuestos son los modelos GARCH y EGARCH.

En el quinto capítulo se da una breve descripción de la tasa de cambio entre dos divisas. Se parte de un modelo de Black-Sholes clásico y se exponen algunos de sus inconvenientes. Se proponen diversas soluciones partiendo del hecho de que la volatilidad no es observable ni es constante, por lo que se debe estimar. También se proponen soluciones para el caso en el que la tendencia no es constante. En algunos meses, la tendencia de la tasa de cambio es alcista y en otros es bajista.

En el sexto capítulo se realizan ajustes de modelos. Se comienza con un análisis de los datos históricos de la tasa de cambio del periodo comprendido del 3 de enero de 2000 al 03 de mayo de 2019. Se estima la volatilidad para cada tiempo, se proponen modelos y se realiza una selección para cada tipo de modelo y después se selecciona aquél que realiza de forma más eficiente las predicciones. Se verifica la eficiencia de los modelos híbridos y se comparan con el resto de los modelos seleccionados individuales.

Posteriormente, se exponen las conclusiones. Se explica si los modelos de redes neuronales presentan un mejor rendimiento que los de series de tiempo heterocedásticos bajo el enfoque frecuentista, y si los modelos híbridos propuestos mejoran las predicciones realizadas por cada uno de los modelos de forma individual.

Capítulo 1

Aprendizaje Supervisado

1.1. Introducción

El *machine learning* o aprendizaje automático consiste en un conjunto de algoritmos que tienen por objetivo la optimización o eficiencia de modelos matemáticos al realizar alguna tarea. A este proceso de optimización se le conoce como aprendizaje o entrenamiento. Cuando se habla de entrenar un modelo se alude al ajuste de los parámetros del modelo mediante algún algoritmo de optimización. En otras palabras, *machine learning* permite que las máquinas "aprendan", es decir, que tengan la capacidad de generalizar a partir de un conjunto de ejemplos o datos observados a casos no previstos.

> *"A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**"*. [22]

La experiencia **E** corresponde al conjunto de los datos observados, la tarea **T** es lo que se desea hacer, mientras que el rendimiento **P** es una medida que permite conocer si el sistema está aprendiendo o no.

El aprendizaje automático se divide principalmente en: supervisado, no supervisado y por refuerzo.

En el aprendizaje supervisado, el conjunto de datos está conformado por las parejas de la forma (x, y) donde $x \in X$ corresponde a las variables explicativas (*input*) y $y \in Y$ al valor asociado a x (*target* o *label*). Entre los problemas que se pueden modelar mediante este tipo de aprendizaje se encuentran los problemas de regresión y los de clasificación, por ejemplo la clasificación de correos electrónicos como spam o no spam, la detección de caracteres en imágenes y la predicción de la temperatura diaria mínima en una determinada región. Este tipo de aprendizaje se abordará en el presente trabajo.

En el aprendizaje no supervisado, el conjunto de datos está conformado únicamente por el conjunto X. Este tipo de aprendizaje resulta útil cuando se desea agrupar los datos en k conjuntos, ya que se buscan patrones o similitudes entre los datos.

El aprendizaje por refuerzo está inspirado en la psicología conductista. Cuando la tarea se realiza de forma correcta, el modelo recibe una recompensa. Este tipo de apendizaje es empleado en los automóviles de conducción autónoma.

En el aprendizaje supervisado, el conjunto $\mathbf{E}$ está conformado por parejas (x,y), donde $x \in X$ y $y \in Y$. Al no contarse con la totalidad de elementos posibles (x,y), el conjunto de datos o la experiencia $\mathbf{E}$ corresponde a una muestra de tamaño n. El conjunto de datos utilizado para ajustar los parámetros del modelo se conoce como conjunto de entrenamiento. Tanto X como Y son variables aleatorias. Se denotará como $\mathbb{P}_{X,Y}(x,y)$ a la función de densidad de probabilidad (f.d.p.) conjunta del vector aleatorio (X,Y) en el punto (x,y), es decir, $\mathbb{P}_{X,Y}(x,y) = \mathbb{P}[X = x, Y = y]$.

El presente trabajo se centrará en los problemas de regresión, siendo la tarea $\mathbf{T}$ la predicción del valor de Y dado X. Sin embargo, se harán algunas menciones para los problemas de clasificación para complementar el contenido.

Sea $\mathscr{F} = \{f : X \to Y\}$ un espacio de funciones y sea Θ es el espacio paramétrico de $\mathscr{F}$ (los parámetros de las funciones $f \in \mathscr{F}$). Uno de los principales supuestos en el aprendizaje supervisado es la existencia de alguna función $f \in \mathscr{F}$ tal que, para cada pareja $(x,y) \in \mathbf{E}$, $f(x) = y$, la cual suele ser difícil de hallar, por lo que se proponen modelos estadísticos. Los parámetros de los modelos propuestos se deben estimar. Para cada pareja (X,Y), a cada una de las componentes de X se les conoce como *features* o atributos. En una regresión lineal múltiple, los *features* corresponden a las variables explicativas, mientras que el *label* a la variable explicada.

En la estadística frecuentista, se busca el estimador máximo verosímil $\hat{\theta}^*$ de θ. En el enfoque de *machine learning*, estos parámetros se ajustan mediante algún algoritmo de aprendizaje. Se denotará como $\hat{f}(x) \equiv f_\theta(x)$ al modelo ajustado con el vector de parámetros $\theta \in \Theta$, de modo que $\hat{y}_i = \hat{f}(x_i)$ es el valor estimado de y correspondiente a x_i.

La relación entre los valores observados, x, y los valores estimados, $\hat{f}(x)$, se puede establecer como $\hat{y} = \hat{f}(x) + \varepsilon$, donde ε recibe el nombre de residual. Cuando se hable de parámetros, se aludirá a los parámetros del modelo a ajustar $\hat{f}(x)$, a menos que se especifique lo contrario.

Uno de los problemas que se pueden presentar es que $\hat{f}$ se ajuste perfectamente a los datos de la experiencia $\mathbf{E}$ pero ante nuevos casos pero posibles (x^*,y^*) no sea eficiente al realizar la tarea $\mathbf{T}$, lo cual indica que el aprendizaje fue escaso. A esta capacidad de un modelo $\hat{f}$ de poder realizar de forma eficiente la tarea $\mathbf{T}$ ante casos no previstos, es decir, no observados previamente (x^*,y^*) se le denominará generalización.

1.2. Partición de la Muestra

Bajo el enfoque de *machine learning*, con el objetivo de poder comprobar que un modelo realiza de forma eficiente la tarea **T**, la muestra o conjunto de datos se debe dividir en tres subconjuntos: conjunto de entrenamiento, conjunto de validación y conjunto de prueba. Considerando esta forma de particionar el conjunto de datos, el conjunto **E** corresponde al conjunto de entrenamiento —el conjunto **E** corresponde a la experiencia—.

El conjunto de entrenamiento se utilizará para entrenar el modelo, mientras que el conjunto de validación permitirá comprobar la eficiencia del modelo al generalizar y, con base en los resultados obtenidos, reajustar los parámetros de aprendizaje como la tasa de aprendizaje, los parámetros del algoritmo de aprendizaje o el número de épocas, además de poder optar por cambiar el algoritmo de aprendizaje, el empleo de algún tipo de regularización si no se había considerado o inclusive el rediseño del modelo, etc.

El conjunto de prueba permitirá verificar si generaliza de forma eficiente ante datos no previstos, es decir, el conjunto de prueba se emplea para la prueba "de hierro". Una buena elección para el tamaño destinado al conjunto de entrenamiento es de al menos el 50 % de la muestra y para el de validación al menos el 25 % del total de la muestra.

1.3. Función de Pérdida

Como se mencionó al comienzo del actual capítulo, en el aprendizaje supervisado se busca aproximar f mediante algúna función $\hat{f}$, lo que equivale a minimizar los errores del modelo ajustado $\hat{f}$ al realizar la tarea **T** ante nuevos datos posibles (x^*, y^*), los cuales no pertenecen al conjunto de datos con el cuál se ajustó el modelo, es decir, minimizar los errores ε_i entre los valores predichos y los observados pero, ¿cómo se puede medir ese error?

Una función de pérdida $\ell : Y \times \hat{f}(X) \to \mathbb{R}$ es una función que mide el error de predicción de y asociado a $\hat{f}$ para cada valor x. Cuando el problema es de regresión, entre las funciones más comunes se encuentran:

$$\begin{aligned} \ell(y, \hat{f}(x)) &= (y - \hat{f}(x))^2, \\ \ell(y, \hat{f}(x)) &= |y - \hat{f}(x)|, \end{aligned} \tag{1.1}$$

siendo $\ell(y, \hat{f}(x)) = (y - \hat{f}(x))^2$ las más utilizada de las anteriores en problemas de regresión.

En los problemas de clasificación con k categorías, $Y = (Y_1, ..., Y_k)$ es un vector aleatorio que sigue una distribución multinomial $(N = 1, p_1, ..., p_k)$, donde $p_i = \mathbb{P}(y_j = 1, y_{j \neq i} = 0), j = 1, 2, ..., k$. $Y_1, ..., Y_n$ es una muestra de variables aleatorias idénticamente distribuidas de independientes. La función que se desea aproximar es $f(x) = (f(x)_1, ..., f(x)_k)$ donde $f(x)_j = p_j = \mathbb{P}(y_j = 1, y_{1 \leq l \neq i \leq k} = 0 | X = x), j = 1, 2, ..., k$. Así, $\hat{f}$ debe ser tal que, para cada una de las k categorías, $\hat{f}(x)_j = p_j$. Luego, $\hat{y}_i$ sigue una

distribución multinomial $(N = 1, p_1 = \hat{f}(x_i)_1 ..., p_k = \hat{f}(x_i)_k)$. La función de pérdida apropiada es:

$$\ell(y, \hat{f}(x)) \quad = \quad -\sum_{j=1}^{k} y_j \log \hat{f}(x)_j. \tag{1.2}$$

En la siguiente sección se abordará con mayor detalle por qué, para problemas de regresión, la función de pérdida (1.1) es la más empleada, mientras que para problemas de clasificación, la función de pérdida (1.2) es la más apropiada.

1.4. Función de Riesgo

El error de aproximación de $\hat{f}$ a f se conoce como error de generalización o riesgo, el cual se define como:

$$R(\hat{f}) := \mathbb{E}_{X,Y}\left[\ell(Y, \hat{f}(X))\right] = \int_{X \times Y} \ell(y, \hat{f}) \mathbb{P}(x, y) d(X \times Y).$$

Debido a que la f.d.p del vector (X, Y) es desconocida y que $\mathbf{E}$ no incluye todos los valores posibles, la función de riesgo se estima mediante la función de riesgo empírica:

$$R_{emp}(\hat{f}) := \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \hat{f}(x_i)),$$

donde n es la cardinalidad de $\mathbf{E}$. Por la ley de los grandes números, R_{emp} converge a R casi seguramente (c.s.), lo cual se puede expresar como $R_{emp} \to R$ c.s.

La función de riesgo empírica[1] se conoce comúnmente como función de costos, denotada por $L \equiv R_{emp}$. La función de costos más utlizada es el error cuadrático medio (RMSE) debido a que considera la media de los errores cuadráticos. [15] El RMSE se define como:

$$\begin{aligned} MSE(\hat{Y}) &= \frac{1}{n}\left(Y - \hat{Y}\right)'\left(Y - \hat{Y}\right) \\ &= \frac{1}{n} \sum_{i=1}^{n} \left\| y_i - \hat{y}_i \right\|^2. \end{aligned}$$

Otras funciones de costos son:

Error Absoluto Medio (MAE) : $\dfrac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|.$

Error Porcentual Absoluto Medio (MAPE) : $\dfrac{1}{n} \sum_{i=1}^{n} \left| \dfrac{y_i - \hat{y}_i}{y_i} \right|.$

[1] Muchos autores se refieren a la función de riesgo empírica como función de pérdida. Para evitar confusiones, se respetará la nomenclatura que se ha dado.

En los problemas de clasificación de k categorías, se recomienda utilizar la función de costos entropía cruzada (*cross-entropy*):

$$R_{emp}(\hat{f}) = -\sum_{i=1}^{n}\sum_{j=1}^{k} y_{ij}\log\hat{y}(x_i)_j$$

donde $0<\hat{y}_i<1$ y $\sum_{i=1}^{k}\hat{y}_i=1$.

Uno de los criteriores para seleccionar la función de riesgo empírica se basa en aquella que permita maximizar la función de verosimilitud $\mathbb{P}(E|\theta)$.

En los problemas de regresión, bajo el supuesto de que $y_1,...,y_n$ son variables aleatorias idénticamente distribuidas e independientes y $\varepsilon_i \sim N(0,\sigma^2)$, $\sigma^2\in\mathbb{R}^+$:

$$\begin{aligned}\mathbb{P}(E|\theta) &= \prod_{i=1}^{n}\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-(y_i-\hat{f}(x_i))^2}{2\sigma^2}}\\ \log\mathbb{P}(E|\theta) &= \frac{n}{2}\log(2\pi\sigma^2)-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(y_i-\hat{f}(x_i)\right)^2.\end{aligned}$$

Maximizar $\mathbb{P}(E|\theta)$ equivale a minimizar $\sum_{i=1}^{n}\left(y_i-\hat{f}(x_i)\right)^2$, lo cual es el *MSE*.

Otro ejemplo se encuentra en los problemas de clasificación de k categorías. Como se mencionó en la sección 1.3, $\hat{y}_i$ sigue una distribución multinomial de parámetros $(N=1, p_1=\hat{f}(x_i)_1,...,p_k=\hat{f}(x_i)_k)$. La función de verosimilitud es:

$$\begin{aligned}\mathbb{P}(E|\theta) &= \prod_{i=1}^{n}\prod_{j=1}^{k}\hat{f}(x_i)_j^{y_{ij}}\\ \log\mathbb{P}(E|\theta) &= \sum_{i=1}^{n}\sum_{j=1}^{k} y_{ij}\log\hat{f}(x_i)_j,\end{aligned}$$

donde y_{ij} corresponde a la j-ésima componente de y_i.

Maximizar $\mathbb{P}(E|\theta)$ equivale a minimizar $-\sum_{i=1}^{n}\sum_{j=1}^{k} y_{ij}\log\hat{f}(x_i)_j$, lo cual es la entropía cruzada.

Por tanto, para problemas de regresión se recomienda emplear la función de costos *MSE*; para problemas de clasificación, la entropia cruzada. En los problemas de regresión no es necesario considerar como función de costos al *MSE*, ya que el aprendizaje depende de los datos y de los algoritmos empleados para el aprendizaje del modelo, por lo que la elección de alguna función de costos no garantiza que el modelo ajustado tenga mayor capacidad de generalización que otro cuyos parámetros se ajustaron empleado una función de costos distinta o que el tiempo de ajuste sea menor.

1.5. Regularización

Al entrenar un modelo, los parámetros del modelo pueden tender a infinito, provocando que el ajuste de los parámetros no converja. Para evitar esto se penaliza a los

parámetros del modelo, lo que se conoce como regularización. Sea $Reg(\theta)$ la función que penaliza a los parámetros del modelo. La función de costos con la regularización $Reg(\theta)$ es:

$$\tilde{L}(\hat{y}) = L(\hat{y}) + Reg(\theta). \tag{1.3}$$

Entre las formas más conocidas de regularización se encuentran la regularización L_1 y la regularización L_2. De forma general, la regularización L_p considera en la ecuación (1.3) que el término $Reg(\theta)$ es de la forma $Reg(\theta) = \lambda \|\theta\|_p^p$, de modo que la función a minimizar es:

$$\tilde{L}(\hat{y}) = L(\hat{y}) + \lambda ||\theta||_p^p,$$

donde $p \in \mathbb{Z}^+$, θ es el vector de parámetros, $||\cdot||_p$ es la norma-p y λ es la tasa de regularización. $L(\hat{Y})$ hace referencia al costo del modelo, mientras que $\lambda ||\theta||_p^p$ penaliza a los parámetros del modelo.

La finalidad de la regularización L_1 es restar importancia a los atributos dispersos, aproximando sus pesos (coeficientes) a cero; la regularización L_2 se utiliza para que los pesos de los atributos atípicos sean cercanos a cero.

Cuando se considera algún tipo de regularización, la función que interesa minimizar es (1.5). Al hablar de minimización de errores o de la función de riesgo, queda implícito si se considera algún tipo de regularización.

1.6. Algoritmos de Aprendizaje

Dependiendo de la complejidad del modelo será o no posible hallar de forma analítica los parámetros $\hat{\theta}^*$ tales que se minimice la función de riesgo. Existen diversos algoritmos de optimización en *machine learning*, los cuales se conocen como algoritmos de aprendizaje. No existe un algoritmo que sea más eficiente al resto, ya que cada uno cuenta con sus ventajas y sus desventajas. La elección del algoritmo depende mucho de los datos, así como el tipo de modelo. Dado un conjunto de datos, un algoritmo de optimización puede ser computacionalmente menos costoso que otro, pero no serlo si los datos cambian.

Dada una función de costos L, se busca elegir los parámetros de manera que L alcance un mínimo local. Uno de los problemas al ajustar los parámetros de un modelo es que éste podría ajustarse adecuadamente a los datos pero no realizar de forma eficiente la tarea **T**. Por ello, los errores de generalización son más relevantes que los de entrenamiento. Se referirá al error de generalización utilizando el conjunto de validación como error de validación, y al utilizar los conjuntos de entrenamiento y de prueba, se referirá como error de entrenamiento y error de prueba, respectivamente.

En esencia, se busca $\hat{f}$ tal que $\mathbb{E}\left[R(\hat{f})\right]$ se minimice, lo que implica minimizar $\mathbb{E}\left[R_{emp}\right]$. Por ende, se busca $\hat{\theta}$ tal que $\mathbb{E}\left[L(\hat{f}(X))|X\right]$ se minimice. El cálculo de esta esperanza puede resultar relativamente sencillo como en el caso de una regresión lineal. Sin embargo, cuando resulta muy complicado calcularla, una alternativa es emplear métodos de aproximación numérica.

Cuando, por el nivel de complejidad de un modelo, resulta difícil hallar de forma analítica $\hat{\theta}^*$ tal que $\mathbb{E}\left[L(\hat{f}(X))|X\right]$ se minimice, los métodos de aproximación numérica o iterativos son una altenativa.

Existen varios algoritmos de aprendizaje, entre los que se encuentran los algoritmos de primer orden y los de segundo orden, los cuales son iterativos. Los algoritmos de primer orden emplean el vector gradiente de la función de costos, mientras que los de segundo orden[2] utilizan la matriz Hessiana dicha función. Entre los algoritmos de primer orden más utilizados se encuentran: *Stochastic Gradient Descent* (*SGD*) y *Adaptive Moment Estimation* (*Adam*). Los algoritmos de aprendizaje basados en el descenso del gradiente son los más conocidos.

En los algoritmos de aprendizaje de primer orden es fundamental que la función de costos L sea diferenciable excepto quizás en un conjunto numerable de puntos, mientras que en los de segundo orden se requiere que la función de costos L sea doblemente diferenciable excepto quizás en un conjunto numerable de puntos.

En los algoritmos de aprendizaje iterativos, se le denomina época cuando el algoritmo de optimización ha utilizado todo el conjunto de datos con el cual se ajustan los parámetros del modelo. Limitar el número de épocas para el ajuste de un modelo es muy importante para evitar que el algoritmo entre en un posible bucle infinito.

Al ser algoritmos iterativos, se requiere de valores iniciales para los parámetros a ajustar. Éstos pueden inicializarse con valores predefinidos, por ejemplo, los parámetros iniciales pueden ser todos cero o todos uno. Existen diversos algoritmos siendo uno de los más simples la simulación de variables aleatorias uniformes $(0,1)$.

1.6.1. Descenso de Gradiente

Los siguientes algoritmos de aprendizaje de primer orden se basan en el descenso de gradiente de la función de costos hacia un mínimo local.

Batch Gradient Descent

Este algoritmo utiliza todo el conjunto de datos con el que se entrena el modelo. Se conoce como *batch gradient descent*. Para aproximarse hacia un valor mínimo local de ℓ existen varias direcciones, siendo la dirección de mayor descenso la contraria a la del vector gradiente (Figura 1.1).

La margnitud de los pasos y el número de iteraciones influyen en si se logra la convergencia y la rapidez con que sucede. A esta margnitud se le denomina tasa de aprendizaje, de manera que se sigue la dirección del vector $-\eta\nabla L$, donde η es la tasa de aprendizaje. Los parámetros se actualizan de forma iterativa:

$$\theta^{(i)} = \theta^{(i-1)} - \eta\frac{\partial L}{\partial \theta},$$

[2]Para mayor información de los algoritmos de segundo orden consulte [8].

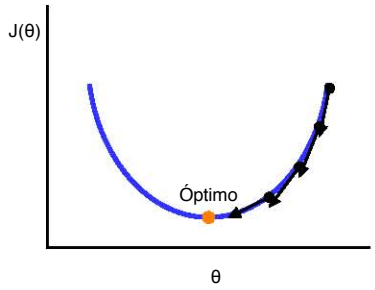


Figura 1.1: Descenso de gradiente. $J(\theta)$ es la función de costos.

siendo $\theta^{(0)}$ el vector de parámetros iniciales.

Descenso de Gradiente Estocástico

El algoritmo de descenso de gradiente es computacionalmente costoso. Una alternativa es seleccionar de forma aleatoria, para cada iteración, un elemento (x,y) del conjunto de datos y llevar a cabo el algoritmo de descenso de gradiente. Este algoritmo recibe el nombre de descenso de gradiente estocástico o *stochastic gradient descent* (*SGD*) [23].

Al considerarse únicamente un elemento (x,y) del conjunto de datos seleccionado de forma aleatoria en cada iteración, el algoritmo de descenso de gradiente estocástico puede resultar más eficiente que el algoritmo de descenso de gradiente. Debido a que el *SGD* considera un valor x de forma aleatoria del conjunto de datos en cada una de las épocas, existe una mayor presencia de ruido en el proceso de convergencia de la función de costos a comparación del algoritmo *batch gradient descent*.

Descenso de Gradiente Estocástico de Minilote

El algortimo de descenso de grandiente estocástico de minilote o *mini-batch gradient descent* es un intermedio entre los algoritmos anteriores. En cada época, se particiona el conjunto de datos, cada subconjunto de un tamaño dado mayor que 1 y menor que el tamaño del conjunto de entrenamiento, y se selecciona un dato de forma aleatoria de cada subconjunto, evitando así disminuir el ruido excesivo del *SGD* y acelerando el algoritmo del *mini-batch gradient descent*. En la Figura 1.2 se puede apreciar de manera gráfica el ruido que se puede presentar en la convergencia al emplear los tres algoritmos mencionados.

1.6.2. Algoritmos de Optimización del Descenso de Gradiente

Con la finalidad de optimizar el proceso de aprendizaje, existen diversos algoritmos de optimización basados en el descenso de gradiente. No existe un algoritmo que sea

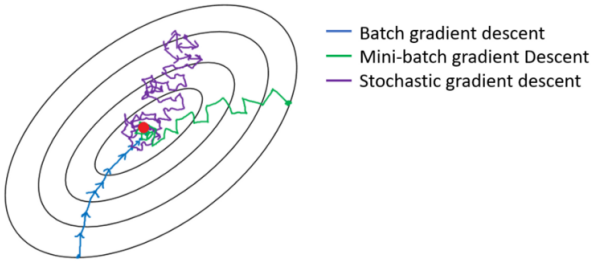


Figura 1.2: Comparación entre *batch gradient descent*, *stochastic gradient descent* y *mini-batch gradient descent*.

Fuente: `https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3`

el más eficiente ya que depende mucho del conjunto de datos, así como del problema en cuestión.

A continuación se presentan dos algoritmos de optimización basados en el descenso de gradiente: *Adagrad* y *Adam*.

Adagrad

El nombre *Adagrad* proviene de *Adaptive Gradient*. Este algoritmo requiere de gradientes de primer orden.[3]. El proceso iterativo es el siguiente:

$$\begin{aligned} g_t &= \nabla_\theta L(\theta_{t-1}), \\ G_t &= \sum_{i=1}^{t} g_t g_t', \\ \theta_t &= \theta_{t-1} - \frac{\lambda}{\sqrt{G_t + \varepsilon}} \cdot g_t, \end{aligned}$$

donde ε es una constante que permite evitar la división entre cero.

Adam

El nombre *Adam* proviene de *Adaptive Moment Estimation*. Este algoritmo requiere de gradientes de primer orden.[4]. El proceso iterativo es el siguiente:

[3]Para mayor información consulte [10].
[4]Para mayor información consulte [19].

$$
\begin{aligned}
g_t &= \nabla_\theta L(\theta_{t-1}), \\
m_t &= \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t, \\
v_t &= \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t \circ g_t, \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\
\hat{v}_t &= \frac{v_t}{1-\beta_2^t}, \\
\theta_t &= \theta_{t-1} - \lambda \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon},
\end{aligned}
$$

donde $\circ$ es el producto elemento a elemento, ε es un valor muy pequeño que se utiliza para evitar división entre cero, λ es la tasa de aprendizaje, m_t y v_t son los vectores correspondientes al primer y segundo momento estimados al tiempo t, respectivamente, y β_1 y β_2 son tasas de decaimiento para el primer y segundo momento estimados, en ese orden. m_0, v_0 son vectores iniciales nulos. Otras versiones del algoritmo Adam se pueden consultar en [26].

1.6.3. *Early Stopping*

En los algoritmos de aprendizaje iterativos basados en el gradiente, para encontrar un punto óptimo $\hat{\theta}^*$ a partir del cual el error de validación deja de disminuir y comienza a aumentar se puede emplear el método *early stopping*, mediante el cual se puede evitar el *overfitting*.

Early stoppping consiste en monitorear en cada época el valor de la función de costos empleando el conjunto de validación. En los problemas de regresión, la función a monitorear es la función de costos[5]. Dadas $\varepsilon \geq 0$ y $d \in \mathbb{Z}$, en cada época se compara el error de validación obtenido en la época con el error de validación mínimo obtenido en el ajuste del modelo (el mínimo de todas las épocas). Si después de d épocas no hubo alguna mejora significativa respecto al mejor valor obtenido durante todo el ajuste, es decir, el valor de validación no fue inferior al registrado por al menos una cantidad $\varepsilon > 0$, se detiene el ajuste del modelo.

1.6.4. *Learning Rate Scheduler*

Elegir un valor adecuado para la tasa de aprendizaje puede ser un problema de ensayo y error. Dependiendo del valor de ésta, el tiempo de entrenamiento puede ser menor e inclusive el algoritmo puede no converger, por lo que que se debe probar con diversos valores para la tasa de aprendizaje, ya sea aumentándola o disminuyéndola.

[5] En los problemas de clasificación, además, se puede monitorear una función de precisión, la cual representa el porcentaje de éxitos obtenidos por el modelo ajustado $\hat{f}$.

El decaimiento de la tasa de aprendizaje (*learning rate scheduler*) permite que la tasa de aprendizaje disminuya su valor a través del tiempo, es decir, al transcurrir las épocas, lo cual permite tener una mayor libertad en la elección de la tasa de aprendizaje y un mejor entrenamiento del modelo. Existen muchos tipos de decaimiento para la tasa de aprendizaje, siendo uno de los más comunes el decaimiento exponencial (*exponential decay scheduler*):

$$\lambda_t = \lambda_0 \delta^t,$$

donde δ es la tasa de decaimiento, λ_0 corresponde a la tasa de aprendizaje inicial (la tasa de aprendizaje elegida antes de comenzar el ajuste del modelo) y λ_t es la tasa de aprendizaje que se utiliza en la época $t-1$. Otros ejemplos de decaimiento de la tasa de aprendizaje se pueden encontrar en [17].

1.7. Compensación Sesgo-Varianza

El sesgo (o *bias*) y la varianza son otros dos problemas a tratar. Cuando se tiene alta varianza y bajo sesgo se incurre en un sobreajuste (*overfitting*). En cambio, baja varianza y alto sesgo implican bajoajuste (*underfitting*) [4].

Supóngase que la función de costos es el RMSE. Sea θ una v.a. y sea $\hat{\theta}$ un estimador de θ. Entonces:

$$\begin{aligned} MSE(\hat{\theta}) &= \mathbb{E}\left[(\hat{\theta}-\theta)^2\right] \\ &= \mathbb{E}\left[(\hat{\theta}-\theta+\mathbb{E}\left[\hat{\theta}\right]-\mathbb{E}\left[\hat{\theta}\right])^2\right] \\ &= \mathbb{E}\left[(\hat{\theta}-\mathbb{E}\left[\hat{\theta}\right])^2\right]+\mathbb{E}\left[(\theta-\mathbb{E}\left[\hat{\theta}\right])^2\right]-2\mathbb{E}\left[(\hat{\theta}-\mathbb{E}\left[\hat{\theta}\right])(\theta-\mathbb{E}\left[\hat{\theta}\right])\right] \\ &= Var(\hat{\theta})+Bias(\hat{\theta},\theta)^2-2\mathbb{E}\left[(\hat{\theta}-\mathbb{E}\left[\hat{\theta}\right])(\theta-\mathbb{E}\left[\hat{\theta}\right])\right]. \end{aligned}$$

Por otro lado,

$$\mathbb{E}\left[(\hat{\theta}-\mathbb{E}\left[\hat{\theta}\right])(\theta-\mathbb{E}\left[\hat{\theta}\right])\right] = \mathbb{E}\left[\hat{\theta}\theta+\mathbb{E}\left[\hat{\theta}\right]^2-\theta\mathbb{E}\left[\hat{\theta}\right]-\hat{\theta}\mathbb{E}\left[\hat{\theta}\right]\right] = 0.$$

$$\therefore MSE(\hat{\theta}) = Var(\hat{\theta})+Bias(\hat{\theta},\theta)^2. \tag{1.4}$$

Sea $\hat{f}(x)$ tal que $f(x)=\hat{f}(x)+\varepsilon, \varepsilon$ independiente de f, $\mathbb{E}[\varepsilon]=0$ y $Var(\varepsilon)=\sigma^2$. De la relación (1.4):

$$\begin{aligned} MSE(\hat{f}+\varepsilon) &= Var(\hat{f}+\varepsilon)+Bias(\hat{f}+\varepsilon,f)^2 \\ &= Var(\hat{f})+Var(\varepsilon)+Bias(\hat{f}+\varepsilon,f)^2 \\ &= Var(\hat{f})+\sigma^2+Bias(\hat{f}+\varepsilon,f)^2 \end{aligned}$$

$$\therefore MSE(\hat{f}+\varepsilon) = Var(\hat{f})+\sigma^2+Bias(\hat{f}+\varepsilon,f)^2$$

De la igualdad anterior se concluye que, si la función de riesgo empírica es el *MSE* entonces, a mayor $Bias(\hat{f})$, mayor riesgo de *underfitting*. Si $\hat{f}$ es insesgado entonces se podría incurrir en *overfitting*.

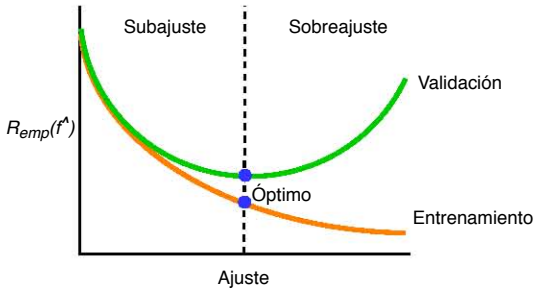


Figura 1.3: Subajuste, sobreajuste y ajuste óptimo.

Se debe ajustar el modelo de forma equilibrada, es decir minimizar su varianza y su sesgo hasta cierto umbral de manera que la condición de paro sea cuando la función de riesgo empírica comience a converger. Este equilibrio recibe el nombre de compensación sesgo-varianza o *bias-variance tradeoff* (Figura 1.3).

1.8. Preprocesamiento de los Datos

Antes de comenzar a diseñar un modelo es fundamental realizar un análisis exploratorio de los datos. ¿Cuál variable se desea predecir? ¿Cuáles atributos tienen relación con la variable a explicar? Información *a priori* como la relevancia que tiene cada variable —algunas variables pueden ser más importantes que otras y algunas podrían no son relevantes—, el tipo de distribución que siguen y los valores posibles que pueden tomar pueden ser de gran utilidad en el ajuste del modelo. Con base en el análisis exploratorio, se pueden proponer los modelos que permitan realizar la tarea **T**.

Una vez definidas las variables que serán los *features* (X) y las que serán los *labels* (Y), se debe identificar su tipo de dato, es decir, si son variables categóricas, continuas, booleanas, cadenas de texto, entre otros.

Los datos deben tener alguna estructura que permita que sean procesados de una forma más sencilla. Es una fase muy importante porque de ésta dependen los resultados que se puedan obtener al entrenar un modelo. Por ejemplo, si se cuenta con un atributo llamado calle, el cual es una cadena y una variable categórica, se debe crear un vector cuyo número de entradas sea igual a la cantidad de calles que se considerarán para el modelo. La componente del vector que corresponde a la calle será 1 y el resto serán 0, es decir, cada componente de la variable calle tiene una distribución binomial cuya esperanza es alguna $p_i \in \mathbb{R}, i = 1, ..., k$, donde k es el número de componentes de la variable calle vectorizada, por lo que la variable calle sigue una distribución multinomial.

Con el objetivo de evitar problemas durante el ajuste de un modelo es fundamental:

- ⋆ Tratar los datos faltantes, ya sea eliminándolos o mediante algún método de imputación de datos.

- ⋆ Quitar los datos duplicados.

Dependiendo de la tarea **T**, puede ser o no conveniente realizar algunas de las siguientes sugerencias, las cuales podrían mejorar el proceso de aprendizaje:

- ⋆ Si se considera que los *outliers* o datos atípicos no son relevantes para la tarea **T**, se deben tratar.
- ⋆ Aligerar las colas pesadas. Una opción es aplicar transformaciones como el logaritmo.
- ⋆ Categorizar una variable. Un variable que representa la temperatura máxima diaria de un país se puede dividir en rangos de manera que se cuente con las categorías frío, cálido y caluroso.
- ⋆ Escalar Las variables numéricas a un rango determinado. Según el modelo propuesto, el aprendizaje podría ser más rápido si las variables se encuentran en determinado rango.
- ⋆ Se recomienda que los *features* no estén fuertemente correlacionados, es decir, que la correlación entre los atributos no sea cercana a 1 para evitar problemas de multicolinealidad. Al tener variables muy correlacionadas se tendrá información duplicada puesto que la información de una estará contenida en algunas de las otras variables, implicando que el modelo tenga un nivel de complejidad innecesario.
- ⋆ En el caso de las variables categórcias, cada una de las posibles categorías debe estar varias veces en el conjunto de datos con el fin de que el modelo pueda aprender de este valor. Si alguna de las categorías aparece pocas veces (una o dos, por ejemplo), el modelo podría no ser capaz de generalizar para esos valores.
- ⋆ No mezclar datos observados con datos sin sentido. Por ejemplo, si se tiene una variable que representa la edad de una persona, no es válido que la variable tome el valor -1 para indicar que no se registró la edad. Una alternativa para esos casos es crear una variable auxiliar binaria que indique si se registró o no la edad.

Los datos deben cumplir los supuestos de los modelos propuestos. Si la variable Y es secuencial, es decir, que Y_t corresponda a un proceso estocástico, un modelo de regresión lineal no sería adecuado.

Existen errores en los cuales resulta complicado verificar su veracidad: etiquetas incorrectas o valores de atributos incorrectos. A mayor veracidad de los datos, mayor eficiencia al generalizar tendrá el modelo.

Respecto a la recomendación de que las variables continuas se encuentren en un determinado rango, de acuerdo con el algoritmo de entrenamiento y del modelo propuesto, puede resultar conveniente que los datos se encuentren en un rango de $(-1,1)$ o $(0,1)$. Sean $T_i : X \to \mathbb{R}$ funciones biyectivas, $i = 1,...,p,p+1$ y p es el número de

atributos, donde T_i es la transformación aplicada al atributo i si $i \leq p$ y T_{p+1} es la transformación asociada a la variable explicada y. Si la variable explicada se utiliza como atributo (como sucede en las series de tiempo), $i = 1, ..., p$ y T_y es la transformación asociada a la variable y. De esta forma, el conjunto de los datos de entrada del modelo será $T(X) = (T_1, ..., T_p)(X)$ y $T^{-1}(\hat{Y})$ la salida. Entre las transformaciones más conocidas se encuentran:

$$T(x) = \frac{x - \bar{x}}{\widehat{sd}(X)}. \tag{1.5}$$

$$T(x) = \frac{x - \text{mín}\{x_i\}}{\text{máx}\{x_i\} - \text{mín}\{x_i\}}. \tag{1.6}$$

$$T(x) = \frac{x}{\text{máx}\{|x_i|\}}. \tag{1.7}$$

A la transformación indicada en la ecuación (1.6) se le conoce como normalización. La estandarización (ecuación (1.5)) es otra de las formas de transformar los datos, la cual permite que diferentes atributos se encuentren en la misma escala.

1.9. Selección del Modelo

Antes de declarar un modelo como eficiente al realizar la tarea **T**, se debe verificar su capacidad de generalización. En ocasiones se cuenta con más de un modelo propuesto, o únicamente con uno. Bajo el enfoque de *machine learning*, es recomendable proponer uno y, con base en los resultados obtenidos en el proceso de validación, decidir si conservar el modelo, mejorar el proceso de aprendizaje o proponer algún otro modelo, ya que entrenar un modelo bajo el enfoque de *machine learning* puede ser computacionalmente costoso.

1.9.1. Validación de Modelos

Para el proceso de validación, cuando se cuenta con más de un modelo, se pueden emplear diversos criterios o metodologías tales como elegir el modelo que tenga los menores errores de generalización, ya sea el de menor valor en la función de costos utilizando el conjunto de validación o empleando otras métricas.

Si es viable obtener la verosimilitud de cada uno de los modelos propuestos, los criterios de información basados en la función de verosimilitud como el *Akaike Information Criterion* (AIC), *Bayesian Information Criterion* (BIC) y el *Deviance Information Criterion* (DIC), de los cuales los últimos dos a su vez consideran el número de parámetros del modelo, pueden ser considerados durante el proceso de validación del modelo. El AIC, BIC y el DIC se definen como:

$$\begin{aligned} \text{AIC} &:= -2\log(\mathbb{P}(X|\theta)) + 2p, \\ \text{BIC} &:= -2\log(\mathbb{P}(X|\theta)) + p\log(n), \\ \text{DIC} &:= -2\log(\mathbb{P}(X|\theta)), \end{aligned}$$

donde p es el número de parámetros, n es el tamaño es la muestra y $\mathbb{P}(x|\theta)$ es la función de verosimilitud bajo el parámetro θ.

Cuando el problema es de regresión, se puede emplear el criterio de la R^2, el cual consiste en elegir aquél modelo cuya R^2 sea más cercana a 1. La medida R^2 se define como:

$$R^2 := \frac{MSS}{TSS} = 1 - \frac{RSS}{TSS},$$

donde

$$\begin{aligned} RSS &= \sum_{i=1}^{N} \left(y_i - \hat{f}(x_i)\right)^2, \\ TSS &= \sum_{i=1}^{N} (y_i - \bar{y})^2, \\ MSS &= \sum_{i=1}^{N} \left(\hat{f}(x_i) - \bar{y}\right)^2, \\ \bar{y} &= \sum_{i=1}^{N} y_i. \end{aligned}$$

Si se cuenta únicamente con un modelo ajustado, se puede considerar únicamente el valor de la función de costos empleando el conjunto de entrenamiento y el de validación, tomando en mayor consideración el de validación. Si los puntajes obtenidos mediante la función de costos o alguna otra métrica empleando el conjunto de validación son aceptables (depende del problema y de la tarea **T**), entonces el modelo se elige para posteriormente probar su eficiencia mediante el conjunto de prueba, lo cual se explicará en la siguiente sección.

Emplear el conjunto de validación es una forma de estimar el riesgo $R\left(\hat{f}\right)$. Al elegir un modelo ajustado de varios basándose en el valor de la función de costos empleando el conjunto de validación es equivalente a seleccionar aquél que minimiza $R\left(\hat{f}\right)$. Si la función de costos se seleccionó a raíz de maximizar la función de verosimilitud (como el MSE en los problemas de regresión), entonces a su vez esto equivale a elegir el modelo cuya función de verosimilitud es la mayor respecto al total de modelos ajustados.

Existen varios criterios de validación, su uso es libre y no existe restricción alguna sobre el número de éstos, siempre y cuando el problema a resolver lo permita (el criterio de la R^2 no se puede aplicar si el problema es de clasificación).

1.9.2. Prueba del Modelo

Tras el proceso de validación de un modelo, se prueba la capacidad de generalización de éste mediante el conjunto de prueba. De forma opcional, se pueden considerar, además de la función de costos, otras métricas distintas a dicha función. Si los resultados no son los deseados, se debe rediseñar, ya sea el proceso de aprendizaje del modelo

o al modelo en sí. Emplear el conjunto de prueba es una forma más eficiente de estimar el riesgo $R\left(\hat{f}\right)$, ya que en este conjunto no se utiliza hasta que el modelo ajustado ha pasado el proceso de validación.

Los conjuntos de validación y de prueba se "desgastan" con el uso repetido ya que, a mayor uso de estos conjuntos, el modelo se ajustará mejor, de manera que se tendrá sobreajuste, lo cual se puede interpretar como que el modelo se aprendió bastante bien el comportamiento de los datos presentados que le es difícil generalizar de forma eficiente. Se recomienda, si es posible, recopilar más datos para reemplazarlos en el conjunto de prueba y en el de validación o ajustar un modelo desde cero. De aquí la importancia de emplear el conjunto de prueba como la "prueba de hierro".

Si bien esta metodología para estimar el riesgo de los modelos $R\left(\hat{f}\right)$ es muy útil, existen algunos otros métodos más robustos de estimarlo, los cuales implementan algunos cambios a la metodología inicial. Cuando los datos no son secuenciales, uno de los más conocidos es la Validación Cruzada de K-Iteraciones (*K-Fold Cross-Validation*), sobre la cual se puede encontrar mayor información en [20]. Al realizar este tipo de validación, se recomienda reservar una partición del conjunto de datos para el conjunto de prueba, ya que éste permite conocer si el modelo realmente es eficiente o no. Sin embargo, cuando los datos son secuenciales, es decir, $\{y_t\}$ es un proceso estocástico, el modelo se puede ajustar modificando el tamaño de la partición del conjunto de entrenamiento. A esta metolodogía se le denominará validación en secuencia.

La metodología es la siguiente: se particiona el conjunto de datos en tres: entrenamiento, validación y prueba. La proporción del conjunto de entrenamiento respecto al conjunto de datos es dinámica, es decir, se tienen diversos casos, cada uno con una proporción distinta para el conjunto de entrenamiento, por ejemplo el $50\,\%, 60\,\%, 70\,\%$ y el $80\,\%$ del conjunto de datos.

Sea $\mathscr{P}$ un conjunto de particiones del conjunto de datos $\mathbf{E}$ tales que el tamaño del conjunto de entrenamiento es distinto entre cada una de las particiones y que además respeten el orden o temporalidad de los datos, es decir, los conjuntos no se forman de forma aleatoria. Los tamaños para el conjunto de entrenamiento deben ser distintos entre cada una de las particiones $\pi \in \mathscr{P}$. Si los conjuntos de entrenamiento, validación y de prueba constituyen el $60\,\%$, el $30\,\%$ y el $10\,\%$ de los datos, los primeros $60\,\%$ datos conforman el conjunto de entrenamiento, los siguientes $30\,\%$ datos corresponden al conjunto de validación y el $10\,\%$ restante al conjunto de prueba. Sean $\mathbf{E}_i \in \mathscr{P}, X_i$ y Y_i el conjunto de datos, de *inputs* y de *targets*, respectivamente, cuya partición es tal que la proporción del conjunto de entrenamiento correspondiente a la i-ésima proporción del conjunto de entrenamiento predefinida.

Al ajuste del modelo bajo los distintos tamaños del conjunto de entrenamiento se le denominará escenario. Así, dos escenarios implican que, dado un modelo, el ajuste de éste se realizó dos veces para cada una de las proporciones predefinidas para el conjunto de entrenamiento. Se selecciona el modelo que mejor desempeño obtenga, es decir, cuya función de costos sea menor con el conjunto de validación.

Si el ajuste de un modelo dado no es reproducible, es decir, la probabilidad de obtener los mismos parámetros estimados al ajustar el modelo bajo las mismas condiciones

(algoritmo de aprendizaje, método de asignación de los parámetros, entre otros) es cero, el algoritmo anterior puede no ser eficiente para estimar el riesgo asociado a dicho modelo. Para mermar la posible variabilidad debida a la no reproducibilidad del modelo ajustado, se pueden considerar m escenarios. De esta forma, la función de riesgo estimada del modelo corresponde a la media de las funciones de riesgo estimadas bajo la metodología anterior.

En resumen, se entrenan m veces los modelos con cada uno de los conjuntos de entrenamiento. Posteriormente, se prueba la eficiencia de cada uno de los modelos ajustados al realizar la tarea **T** con su respectivo conjunto de validación. El riesgo del modelo se estima promediando los riesgos estimados del modelo de cada escenario. De forma adicional, se puede estimar el riesgo del modelo empleando los conjuntos, ya sea de entrenamiento o de prueba, en lugar del conjunto de validación. Al emplear el conjunto de entrenamiento, se puede realizar un análisis sobre su eficiencia bajo los datos de entrenamiento, mientras que al utilizar el conjunto de prueba, se prueba el modelo bajo datos no previstos.

Cuando se emplea este método para comparar la eficiencia entre dos o más modelos, (aludiendo al diseño de éstos), se elige el modelo (el diseño del modelo) que mejor desempeño haya obtenido bajo el conjunto de validación.

Sea m el número de escenarios. Sea $\mathscr{P}$ un conjunto de particiones de E tales que el tamaño del conjunto de entrenamiento es diferente entre cada una de las particiones. Sea D_π el conjunto de datos de validación correspondiente a la partición $\pi \in \mathscr{P}$. Sea $\hat{f}_{\theta_{i,\pi}}$ el modelo ajustado correspondiente al escenario j cuyos parámetros fueron estimados empleando el conjunto de entrenamiento según la partición π. Sea $\hat{f}^\star$ el modelo a ajustar. Sea $R^{(j)}_{emp}\left(\hat{f}^\star, \mathscr{P}\right)$ la función de riesgo empírica del modelo a ajustar $\hat{f}^\star$, asociada a $\mathscr{P}$, del escenario j, $j = 1,2,...,m$. La función de riesgo $R\left(\hat{f}^\star\right)$ se puede estimar mediante la función $R^{(j)}_{emp}$ de la forma siguiente:

$$
\begin{aligned}
R_{emp}\left(\hat{f}^\star, \mathscr{P}\right) &= \frac{1}{m}\sum_{j=1}^{m} R^{(j)}_{emp}\left(\hat{f}^\star, \mathscr{P}\right), \qquad (1.8)\\
R^{(j)}_{emp}\left(\hat{f}^\star, \mathscr{P}\right) &= \frac{1}{|\mathscr{P}|}\sum_{\pi\in\mathscr{P}}\frac{1}{|D_\pi|}\sum_{\xi\in D_\pi}\ell\left(\xi_2, \hat{f}_{\hat{\theta}_{j,\pi}}\right), \xi = (\xi_1,\xi_2), j = 1,2,...,m
\end{aligned}
$$

Cuando los datos son secuenciales y los modelos ajustados no son reproducibles, la ecuación (1.8) permite estimar $\mathbb{E}_\theta\left[\mathbb{E}_{X,Y}\left[\ell\left(Y, \hat{f}_\theta(X)\right)|\theta\right]\right] = \mathbb{E}_{X,Y}\left[\ell\left(Y, \hat{f}^\star(X)\right)\right] = R(\hat{f}^\star)$ de forma más eficiente, que la metodología simple de entrenamiento-validación-prueba.

La validación en secuencia puede ser computacionalmente muy costosa por lo que, podría ser una opción más viable emplear la metodología simple de entrenamiento-validación-prueba.

Capítulo 2

Redes Neuronales

En el presente capítulo se presentará un bosquejo de la teoría de redes neuronales. Se mencionarán algunos de los tipos de modelos existentes. El objetivo es proporcionar un panorama que permita comprender las redes neuronales.

2.1. Introducción

Las redes neuronales son modelos matemáticos que, de cierta manera, están inspirados en el funcionamiento del cerebro. Se han tenido grandes avances en las últimas décadas debido a la necesidad de procesar una gran cantidad de datos y al incremento del poder computacional. Las redes neuronales están conformadas por funciones que pueden o no ser lineales, lo cual supone una gran ventaja ya que los datos no necesariamente pueden describirse mediante alguna regresión lineal.

Entre las principales ventajas de las redes neuronales respecto a otros modelos se encuentran:

- ⋆ Suelen tener mayor capacidad de generalización.
- ⋆ Se puede emplear cómputo paralelo en el entrenamiento de una red neuronal.
- ⋆ Las transformaciones en el modelo pueden ser lineales o no lineales.
- ⋆ El diseño de las redes es muy versátil, por lo que permiten resolver diversos problemas.

Una de las principales desventajas de las redes neuronales es que son computacionalmente más costosas que otros modelos.

2.2. Neurona, la Unidad Básica

Una neurona consiste en una función valuada en una combinación lineal de los valores de los atributos de un *input* $x \in X$. El vector de parámetros es $\theta = W$, donde W es el vector de los parámetros de la transformación lineal.

2.2.1. Función de Activación

Una función de activación $a : \mathbb{R}^n \to \mathbb{R}$ procesa los datos recibidos por la neurona. Las funciones de activación más comunes son:

$$\begin{aligned} \text{Sigmoide o Logística:} \quad & a(x) = \sigma(x) = \frac{1}{1+e^{-x}}. \\ \text{ReLU (Unidad Lineal Rectificada):} \quad & a(x) = \text{máx}\{x, 0\}. \\ \text{Tangente Hiperbólica:} \quad & a(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 2\sigma(2x) - 1. \end{aligned}$$

La función sigmoide (Figura 2.1) comúnmente se representa por σ (de sigmoide). Es la más utilizada debido a que:

⋆ Es diferenciable para cualquier $x \in \mathbb{R}$. Su derivada no se anula en ningún punto:

$$\frac{d\sigma(x)}{dx} = \sigma(x) \cdot (1 - \sigma(x)).$$

⋆ Se encuentra entre 0 y 1, por lo que se puede asociar al cálculo de probabilidades y, con ello, a la predicción de categorías en base a las probabilidades obtenidas. Al ser una función no lineal, el modelo entrenado es no lineal.

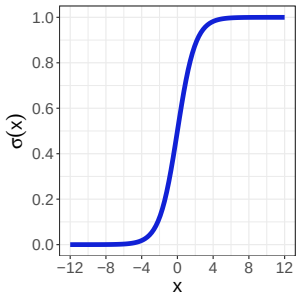


Figura 2.1: Función sigmoide.

2.3. Red Neuronal

Una red neuronal está conformada por una o varias neuronas, entre las cuales existe alguna relación. Existen varios modelos de redes neuronales. Los más conocidos son el perceptrón y el perceptrón multicapa. Varios modelos de redes neuronales son variaciones del perceptrón multicapa[1]. La arquitectura de una red neuronal define las relaciones entre las neuronas y la forma en que están posicionadas [30].

Debido a la posible existencia de ciclos en el grafo dirigido de la red neuronal, no es sencillo de expresar de forma general la expresión matemática de una red neuronal. Sin embargo, una forma de expresar la salida de la neurona i es:

$$neurona_i = a_i \left(\sum_{j=1}^{n} neurona_j w_{ji} + b_i \right),$$

donde $neurona_i$ es la salida de la i-ésima neurona, a_i es la función de activación de la i-ésima neurona w_{ij} es el peso corresponde de la neurona j a la i, b_i es el sesgo del *input* de la neurona i y n es el número de neuronas de la red. $w_{ii} \neq 0$ para alguna $i = 1, ..., n$.

2.3.1. Capas de una Red Neuronal

La capa de entrada está conformada por nodos, donde cada nodo representa cada uno de los *features*. Estos nodos no son nueronas, únicamente corresponden a los datos de entrada de la red. La última capa es la capa de salida, la cual devuelve la salida de la red neuronal. Las capas que se encuentran entre la capa de entrada y la de salida se denominan capas ocultas.

Cuando los datos han sido procesados, es decir que han llegado a la capa de salida, se emplea una función de activación que permite que los datos tengan la forma deseada, por ejemplo que se obtenga un vector de valores entre 0 y 1 con K componentes. La elección de esta función depende del problema que se quiere resolver. Para problemas de regresión se suele emplear la función $f(z) = z'W$, donde z corresponde al vector de las salidas de la última capa de la red previa a la capa de salida.

Para los problemas de clasificación de K categorías se suele emplear en la capa de salida la función softmax $a : \mathbb{R}^K \to [0,1]^K$, la cual permite asignar probabilidades para cada una de las K categorías. Se define como:

$$a(z_i) = \frac{e^{z_i}}{\sum_{i=1}^{k} e^{z_i}},$$

donde z_i corresponde a la i-ésima transformación lineal de los valores de salida de la capa anterior, mientras que $a(z_i)$ se asocia a la probabilidad estimada de la i-ésima categoría, $i = 1, ..., k$.

[1] Existe otros modelos de redes neuronales tales como las redes convolucionales, las recurrentes y las generativas adversariales. Para mayor información consulte [31].

2.3.2. Redes *Feedforward*

Las redes neuronales *feedforward* (*FNN*) o de propagación hacia delante constan de una función compuesta de otras funciones. Los datos de entrada "viajan" hacia la capa de salida sin formar ciclos, es decir "de forma directa". El perceptrón simple y perceptrón multicapa son ejemplos de redes *feedforward*. En este tipo de redes, es fundamental que los datos no sean secuenciales, es decir, que no exista una dependencia respecto al tiempo. Si $\{y_t\}$ es un proceso estocástico, las redes *FNN* no son adecuadas.

Perceptrón

Una red neuronal perceptrón consta de dos capas: la capa de entrada y la capa de salida. La red perceptrón consiste en una sola neurona, siendo esta neurona la capa de salida.

Supóngase que se tienen *p features*. Sea x un vector de entradas. Sea W el vector de pesos. La neurona recibe una combinación lineal de las entradas, $x'W+b$, donde b es el vector de sesgo o *bias*. La finalidad del sesgo es permitir que el modelo no esté centrado en el cero.

Dada una función de activación $a(x)$, el modelo perceptron se representa de forma matemática como:

$$\hat{f}(x) = a(x'W+b).$$

Perceptrón Multicapa

El perceptrón multicapa o *Multi-Layer Perceptron* (*MLP*), está conformado por una capa de entrada, $m-1$ capas ocultas, $m \in \mathbb{Z}^+$, y una capa de salida. Las capas ocultas se conectan con su capa antecesora y la siguiente.

Cada capa oculta tiene su función de activación que no necesariamente debe ser la misma que el resto de las capas. La capa de salida aplica la función de salida y devuelve el valor de los *labels* correspondientes a los datos de entrada. En la Figura 2.2 se representa de forma gráfica la arquitectura de un perceptrón multicapa.

De forma matemática, un perceptrón multicapa se representa como:

$$\hat{f}(x) = a_m\left(a_{m-1}\left(\cdots\left(\left(a_1\left(x'W_1+b_1\right)'W_2+b_2\right)'\cdots\right)'W_{m-1}+b_{m-1}\right)'W_m+b_m\right),$$

donde:

a_i es la función de activación de la i-ésima capa.

W_i es el vector de pesos de la i-ésima capa.

b_i es el sesgo de la i-ésima capa.

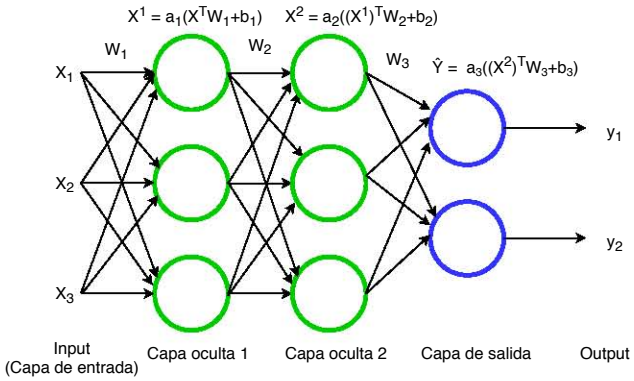


Figura 2.2: Grafo dirigido de un perceptrón multicapa. a_1, a_2 corresponden a alguna función de activación para las capas ocultas 1 y 2, mientras que a_3 es una función de activación para la capa de salida.

Gracias a que la función de activación de cada neurona puede no ser lineal y a las conexiones entre distintas neuronas, mediante las redes neuronales es posible aproximar una infinidad de funciones.

2.4. Algoritmo de Aprendizaje

Una de las formas más empleadas en el aprendizaje de las redes neuronales es calculando los errores de generalización del modelo y, posteriormente, ajustar los pesos con base en éstos. Este algoritmo se conoce como *backpropagation*.

2.4.1. *Backpropagation*

De forma inicial, se le asignan valores a los parámetros de la red (pesos y sesgos). Posteriormente, dado un conjunto de datos, se calculan los errores cometidos por la red mediante una función de costos previamente seleccionada. Se elige algún algoritmo de aprendizaje de *machine learning*, por ejemplo, *SGD*. Luego, se recalculan los errores de predicción y se emplea nuevamente el algoritmo de aprendizaje y así sucesivamente hasta cumplir algún criterio de paro.[2]

[2]El desarrollo detallado del algoritmo de *backpropagation* se puede encontrar en [28].

2.5. Optimización del Aprendizaje de Redes Neuronales

En ocasiones, el ajuste de modelos bajo el enfoque de *machine learning* puede ser lento y el modelo ajustado puede no tener una eficiencia deseada. Existen diversos algoritmos de optimización de los algoritmos de aprendizaje.

Debido a la gran cantidad de datos que se procesan y a la posible escalabilidad de los modelos de redes neuronales, es decir, aumentar el número de neuronas y las conexiones entre éstas, entrenar una red neuronal puede ser computacionalmente costoso. Existen formas de disminuir el tiempo de entrenamiento como el cómputo paralelo y el empleo de tarjetas gráficas (*GPU*). Se considera que una red es de aprendizaje profundo cuando está constituida por varias capas, por ejemplo 100.

A diferencia de otros modelos, las redes neuronales pueden ser diseñadas de diversas formas, aumentando así su complejidad en el sentido de la forma en que están conectadas las neuronas de la red, no en el de aumentar la cantidad de parámetros a ajustar. Existen otros modelos de redes neuronales, entre las que se encuentran las redes neuronales convolucionales (*CNN*))[3].

2.5.1. Técnicas de Optimización del Aprendizaje

Existen diversas técnicas que permiten mejorar el aprendizaje. A diferencia del enfoque tradicional de *machine learning*, en el *deep learning* se implementan diversas técnicas que permiten optimizar el proceso de aprendizaje, algunas de las cuales incrementan la complejidad del modelo. Estas técnicas de optimización están destinadas principalmente a las redes neuronales, entre las cuales se encuentra el *dropout*[14].

Dropout consiste en anular, de forma aleatoria, la salida de un determinado porcentaje de neuronas para cada capa oculta, evitando así el sobreajuste.

Otra de las técnicas es *batch normalization*[4], en el cual antes de que las salidas z de la capa anterior ingresen a la siguiente capa, es decir, antes de ser ponderados con el respectivo vector de pesos, se estandarizan de la forma:

$$T(z) = \frac{z - \bar{z}}{\hat{sd}(z)}.$$

El implementar todas las técnicas no necesariamente hará que el aprendizaje de un modelo sea superior a si se ajusta el mismo modelo bajo únicamenta una de las técnicas empleadas, por lo que se recomienda emplear aquellas que, de acuerdo al problema en cuestión, podrían mejorar el aprendizaje. Otra técnicas de optimización del aprendizaje de los modelos de redes neuronales se pueden encontrar en [23].

[3] Los modelos de *CNN* no se abordarán. Para obtener mayor información acerca de este tipo de redes se recomienda consultar [31].

[4] Para mayor información consulte [18].

Capítulo 3

Redes Neuronales Recurrentes

Como se mencionó en el capítulo 2, uno de los inconvenientes de las redes *feed-forward* es que los datos no deben ser secuenciales. Cuando existe cierta dependencia a través del tiempo, las redes *FNN* se vuelven obsoletas. Las redes recurrentes se utilizan cuando los datos son secuenciales. La salida de la neurona de una capa puede estar conectada a sí misma o a una capa anterior. En las redes recurrentes se tienen ciclos, lo cual es ideal cuando $\{y_t\}$ es un proceso estocástico. En el presente capítulo se mostrará con mayor detalle el tema de las redes neuronales recurrentes y se expondrán tanto un modelo sencillo de red neuronal recurrente como algunos otros modelos más complejos.

3.1. Introducción

Sea $\{x_t\}$ un proceso estocástico a tiempo discreto. Una *RNN* se puede expresar como:

$$g(x_t) = \hat{f}\left(x_t' W + g\left(x_{t-1}\right)' U + b\right),$$

donde

$\hat{f}$ es la función correspondiente a la red neuronal.

x_t es el *input* del tiempo t.

W y U son vectores de pesos.

$g(x_t)$ corresponde a la salida de la red al tiempo t.

Las *RNN* tienen múltiples aplicaciones:

⋆ Predicción de series temporales.

- ⋆ Reconocimiento de voz.
- ⋆ Predictores de texto.
- ⋆ Traductores de idiomas.

3.2. Arquitecturas

Existen tres principales tipos de arquitecturas de *RNN*: uno a uno, uno a muchos, muchos a uno y muchos a muchos. La primera palabra de los nombres de las arquitecturas anteriores hace referencia a la cantidad de tiempos requeridos de las variables explicativas ($\{x_t\}$) mientras que el último alude al número de tiempos de la variable a explicar ($\{y_t\}$). Por ejemplo, sea $\hat{f}$ el modelo correspondiente a una red recurrente. La arquitectura uno a uno es de la forma $y_t = \hat{f}(x_t)$; la de muchos a uno, $y_t = f(f(...f(f(x_{t-m}), x_{t-m+1}), ..., x_{t-1}), x_t)$.

3.3. Algoritmo de Aprendizaje

Debido a la dependencia temporal, los errores deben calcularse para cada uno de los tiempos. En el caso del algoritmo *backpropagation*, al calcular los errores para cada uno de los tiempos se conoce como *Backpropagation Through Time*. [9]

3.4. Modelos

Existen varios modelos de redes recurrentes, entre los que se encuentran:

- ⋆ *Time-Delay Neural Network* (TDNN)
- ⋆ Elman
- ⋆ Hopfield
- ⋆ *Long Short-Term Memory* (LSTM)
- ⋆ *Gated Recurrent Unit* (GRU)

Los modelos Elman, Hopfield y Time-Delay Neural Network no se abordarán. Si se desea mayor información acerca de estos modelos se recomienda consultar [8] y [31].

A continuación se presentan algunos modelos de redes neuronales recurrentes.

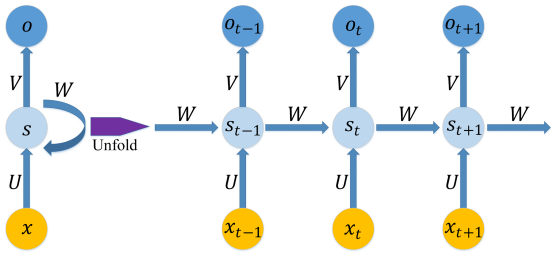


Figura 3.1: Arquitectura de una red neuronal recurrente sensilla. Cada valor de salida se almacena en la memoria de la red y se utiliza en tiempos posteriores. A la izquierda se muestra el modelo mientras que a la derecha se muestra el flujo con mayor detalle.

Fuente: `https://doi.org/10.1371/journal.pone.0180944.g004`

3.4.1. Red Neuronal Recurrente Simple

Una red neuronal recurrente simple (Figura 3.1) se puede expresar de forma matemática como:

$$h_t = \phi(x_t' U + h_{t-1}' W + b_1), \tag{3.1}$$
$$o_t = g(h_t' V + b_2), \tag{3.2}$$

donde:

U, V, W son vectores de pesos.

x_t es la entrada al tiempo t.

o_t es la salida de la red al tiempo t.

h_t es la salida de la neurona (estado) al tiempo t.

ϕ es la función de activación de un estado a otro.

g es la función de activación para la capa de salida.

b_1, b_2 son vectores de sesgo (*bias*).

Vanishing Gradient

En las redes neuronales recurrentes sencillas se presentan dos principales problemas: *vanishing gradient* y *exploding gradient*. *vanishing gradient* se refiere a que los gradientes tienden a cero por lo que, conforme aumentan la longitud de las secuencias, el aprendizaje tiende a ser nulo. Por otro lado, *exploding gradient* se refiere a que los gradientes tienden a infinito por lo que, a medida que aumenta la longitud de las secuencias, los parámetros tienden a infinito.

Supóngase que se tiene una red neuronal sencilla (ecuaciones (3.1) y (3.2)) y que el algoritmo de aprendizaje es *backpropagation*. Sea L una función de costos. Debido a la dependencia temporal existente, la función de costos del modelo se calcula para cada tiempo, por lo que se considerará el costo asociado a la función al tiempo t, el cual se denotará por L_t. Así,

$$\begin{aligned}
\frac{\partial L_t}{\partial W} &= \sum_{k=1}^{t} \frac{\partial L_t}{\partial o_t}\frac{\partial o_t}{\partial h_t}\frac{\partial h_t}{\partial h_k}\frac{\partial h_k}{\partial W} = \frac{\partial L_t}{\partial o_t}\frac{\partial o_t}{\partial h_t}\sum_{k=1}^{t}\frac{\partial h_t}{\partial h_k}\frac{\partial h_k}{\partial W}, \\
\frac{\partial h_t}{\partial h_k} &= \prod_{i=k+1}^{t}\frac{\partial h_i}{\partial h_{i-1}} \\
&= \prod_{i=k+1}^{t} diag\left(\frac{\partial}{\partial h_{i-1}}\phi\left(U'x_i + W'h_{i-1} + b_1\right)\right)' W. \qquad (3.3)
\end{aligned}$$

Por otro lado,

$$\left\|\frac{\partial h_i}{\partial h_{i-1}}\right\| \leq \|W\| \left\| diag\left(\frac{\partial}{\partial h_{i-1}}\phi\left(x_i'U + h_{i-1}'W + b_1\right)\right)\right\|. \qquad (3.4)$$

De la desigualdad anterior y la igualdad (3.3):

$$\left\|\frac{\partial h_t}{\partial h_k}\right\| \leq \|W\|^{t-k} \prod_{i=k+1}^{t} \left\| diag\left(\frac{\partial}{\partial h_{i-1}}\phi\left(x_i'U + h_{i-1}'W + b_1\right)\right)\right\|. \qquad (3.5)$$

Si existe $\gamma \in \mathbb{R}$ tal que

$$\left\| diag\left(\frac{\partial}{\partial h_{i-1}}\phi\left(x_i'U + h_{i-1}'W + b_1\right)\right)\right\| < \gamma$$

y $\|W\| < 1/\gamma$, entonces

$$\left\|\frac{\partial h_t}{\partial h_k}\right\| \xrightarrow{t\to\infty} 0 \implies \left\|\frac{\partial L_t}{\partial W}\right\| \xrightarrow{t\to\infty} 0. \qquad (3.6)$$

De la expresión (3.6) se concluye que, si el aprendizaje de una red neuronal recurrente simple (ecuaciones (3.1) y (3.2)) es *backpropagation* y el algoritmo de aprendizaje se basa en el descenso de gradiente, el gradiente tiende a cero, lo que implica que, a medida que el tiempo transcurre, el ajuste de los parámetros tiende a ser nulo. A esto se le conoce como *vanishing gradient*[1].

Para solucionar el *exploding gradient* basta con aplicar un umbral a los gradientes, es decir, acotarlos tanto inferior como superiormente. El verdadero problema de las redes neuronales recurrentes simples es el *vanishing gradient*. Existen algunos modelos de redes recurrentes que no presentan el problema del *vanishing gradient* tales como las redes LSTM y las redes GRU, los cuales corresponden al *deep learning*.

[1] Para mayor información consulte [25]

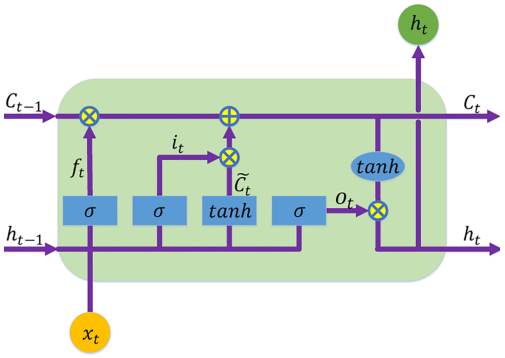


Figura 3.2: Arquitectura de una red LSTM.

Fuente: `https://doi.org/10.1371/journal.pone.0180944.g006`

3.4.2. *Long Short-Term Memory*

El modelo *Long Short-Term Memory* (LSTM) fue propuesto por Sepp Hochreiter y Jürgen Schmidhuber (1997).[16] Consta de una celda de memoria y tres puertas: la puerta de entrada, la de olvido y la de salida. Este modelo se caracteriza por tener memoria debido a que es capaz de recordar y olvidar datos históricos. Consta de una capa de entrada y una celda de memoria que juega el rol de las capas ocultas (Figura 3.2). A comparación del perceptrón multicapa, las redes neuronales recurrentes LSTM están conformadas por celdas de memoria en lugar de neuronas.

La celda de memoria está conformada por tres puertas: de entrada, de olvido y de salida. La puerta de olvido y la de salida están compuestas por la función de activación sigmoide, mientras que la puerta de entrada consta de dos funciones: la función sigmoide y la tangente hiperbólica (tanh). La puerta de entrada selecciona la información que ha de utilizarse, ignorando aquella que es irrelevante; la puerta de olvido decide la información histórica que ha de olvidarse; la puerta de salida se encarga de seleccionar la información que se utilizará como salida. La función tanh es infinitamente diferenciable, por lo que no se desvanece, además de que su rango es $(-1,1)$, de modo que valores cercanos a 1 indican que la información debe recordarse, mientras que valores cercanos a -1 implican que se debe olvidar [2].

De forma matemática, el modelo se expresa como:

$$\begin{aligned} i_t &= \sigma(x_t' U_i + h_{t-1}' V_i + b_i), \\ f_t &= \sigma(x_t' U_f + h_{t-1}' V_f + b_f), \\ g_t &= \tanh(x_t' U_g + h_{t-1}' V_g + b_g), \\ o_t &= \sigma(x_t' U_o + h_{t-1}' V_o + b_o), \\ c_t &= f_t' c_{t-1} g_t' i_t, \\ h_t &= \tanh(c_t)' o_t, \end{aligned} \tag{3.7}$$

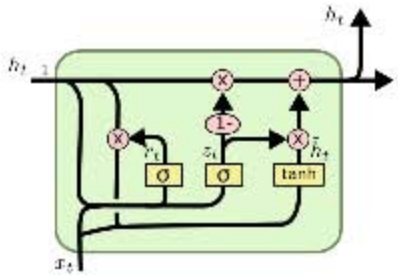


Figura 3.3: Arquitectura de una red GRU.

Fuente:
`www.data-blogger.com/2017/08/27/gru-implementation-tensorflow/`

donde c_t, f_t, i_t, o_t y h_t son los vectores de la memoria de la celda, la puerta de olvido, la puerta de entrada, la puerta de salida y de la historia al tiempo t, respectivamente; y b_i, b_f, b_g y b_o son vectores de sesgo; g_t se conoce como estado de memoria candidato o simplemente como candidato y a h_t como estado oculto.

Derivando (3.7) respecto a c_{t-1}:

$$\frac{\partial c_t}{\partial c_{t-1}} = diag(f_t).$$

De la igualdad anterior se puede ver que la arquitectura LSTM no tiene el problema de *vanishing gradient*, por lo que se puede entrenar mediante *backpropagation*.

3.4.3. *Gated Recurrent Unit*

El modelo *Gated Recurrent Unit* (GRU) es una versión simplificada del LSTM. Fue propuesto por Kyunghyun Cho et al. (2014). Consta de una puerta de actualización de la memoria almacenada y de una puerta de salida (Figura 3.3). [6] Se representa de forma matemática como:

$$\begin{aligned} r_t &= \sigma(x_t' W_i + h_{t-1}' V_i + b_r), \\ u_t &= \sigma(x_t' W_f + h_{t-1}' V_f + b_u), \\ \tilde{h}_t &= \tanh(r_t' W_r + h_{t-1}' W_h + x_t' U + b_h), \\ h_t &= h_{t-1}'(1-u_t) + \tilde{h}_t' u_t, \end{aligned}$$

donde r_t, u_t y h_t corresponden a las puertas de reinicio, actualización y salida, respectivamente.

No es dificil comprobar que el modelo GRU no tiene el problema de *vanishing gradient*.

Como sucede en las redes recurrentes LSTM, las redes GRU están conformadas por celdas de memoria en lugar de neuronas.

El modelo GRU es más sencillo y computacionalmente menos costoso que el modelo LSTM. Sin embargo, se tiene menos control sobre los datos. Como se comentó en capítulos pasados, que un modelo sea más complejo que otro no implica que sea mejor al momento de generalizar. Para un problema dado puede resultar más eficiente el modelo GRU y para otro el modelo LSTM.

Capítulo 4

Series de Tiempo

Existen varios modelos de series de tiempo, entre los que se encuentran los procesos autorregresivos (AR) o los de medias móviles (MA). Uno de los supuestos de estos modelos es que los residuales $\{\varepsilon_t\}$ tienen varianza constante, es decir, $Var[\varepsilon_t|\mathscr{F}_{t-1}] = \sigma$, donde σ es una constante, lo cual recibe el nombre de homocedasticidad. Sin embargo, algunas series de tiempo no cumplen ese supuestos, ya que esa varianza no es constante, lo cual se denomina heterocedasticidad.

En el presente capítulo se presentarán algunos modelos de series de tiempo heterocedásticos. Para entrar en contexto, se recordará el concepto de estacionariedad y algunos otros conceptos fundamentales.

Sea $\{X_t\}$ un proceso estocástico. En el caso continuo, se denotará como $\mathscr{F}_t, t \geq 0$, a la σ-álgebra generada por el proceso hasta tiempo t sin incluir t, es decir, $\mathscr{F}_t = \sigma(\{X_s\}_{s<t})$, mientras que en el caso discreto se denotará como $\mathscr{F}_{t-1} = \sigma(X_0, X_1, ..., X_{t-1})$. Se considerará que los procesos estocásticos son a tiempo discreto.

4.1. Conceptos Básicos

Sea $\{X_t\}$ un proceso estocástico adaptado a la filtración $\mathscr{F}_{t-1}, t \geq 0$. A continuación se definen tres funciones:

Función de Autocovarianza:	$\gamma_X(h)$	:=	$Cov(X_t, X_{t+h})$.
Función de Autocorrelación (ACF):	$\rho_X(h)$	:=	$Corr(X_t, X_{t+k}) = \frac{\gamma_X(h)}{\gamma_X(0)}$.
Función de Autocorrelación Parcial (PACF):	ϕ_{kk}	:=	$Corr(X_t, X_{t+k}\|X_{t+1}, ..., X_{t+k-1})$.

Definición 4.1.1 (Estrictamente estacionario) *Se dice que el proceso $\{X_t\}$ es estric-*

tamente estacionario si los vectores $(X_1,...,X_k)'$ *y* $(X_{1+\tau},...,X_{k+\tau})'$ *tienen la misma distribución conjunta para todo* $k,\tau \in \mathbb{Z}$.

Definición 4.1.2 (Estacionariedad de segundo orden) *Se dice que el proceso* $\{X_t\}$ *es estacionario de segundo orden o débilmente estacionario si:*

i) $X_t \in L^2 \quad \forall t \in \mathbb{Z}$.

ii) $\mathbb{E}[X_t] = m \quad \forall t \in \mathbb{Z}$.

iii) $Cov(X_t, X_{t+h}) = \gamma_X(h) \quad \forall t,h \in \mathbb{Z}$.

Definición 4.1.3 (Ruido blanco débil) *Sea* $\{\varepsilon_t\}$ *un proceso estocástico se denomina ruido blanco débil si, para alguna constante* σ^2*:*

i) $\mathbb{E}[\varepsilon_t] = 0 \quad \forall t \in \mathbb{Z}$.

ii) $Var[\varepsilon_t] = \sigma^2 \quad \forall t \in \mathbb{Z}$.

iii) $Cov[\varepsilon_t, \varepsilon_{t+h}] = 0 \quad \forall t,h \in \mathbb{Z}, h \neq 0$.

Definición 4.1.4 (Ruido blanco fuerte) *Sea* $\{\varepsilon_t\}$ *un proceso estocástico se denomina ruido blanco fuerte si, para alguna constante* σ^2*:*

i) $\mathbb{E}[\varepsilon_t] = 0 \quad \forall t \in \mathbb{Z}$.

ii) $Var[\varepsilon_t] = \sigma^2 \quad \forall t \in \mathbb{Z}$.

iii) ε_t *y* ε_{t+h} *son independientes* $\forall t,h \in \mathbb{Z}, h \neq 0$.

Dada una serie de tiempo $\{X_t\}$, el valor predicho al tiempo t por un modelo se denota como $\hat{x}_t$. El residual al tiempo t se define como $\varepsilon_t := x_t - \hat{x}_t$. Cuando $Var(\varepsilon_t|\mathscr{F}_t) = \sigma^2$ es constante se conoce como homocedasticidad. El caso contrario recibe el nombre de heterocedasticidad. En ocasiones, se supone que los residuales son un proceso de ruido blanco.

Las gráficas o correlogramas del ACF y del PACF son de gran ayuda para identificar el tipo de modelo de serie de tiempo correspondiente a una serie de tiempo dada. El correlograma del ACF permite ver si existe correlación respecto a tiempos pasados o *lags*, mientras que el correlograma el PACF muestra la correlación que existe respecto a tiempos pasados dada la información entre ambos tiempos.

Se denotará por B^p al operador de retraso de grado p, es decir, $B^p X_t = X_{t-p}$.

4.2. Procesos Heterocedásticos

En las series de tiempo se pueden presentar diversos problemas como la heterocedasticidad y la presencia de colas pesadas en la distribución. Además, factores positivos podrían afectar de forma distinta a la serie de tiempo que los negativos, ya sea en mayor o menor medida.

A continuación se presentan algunos modelos de series de tiempo heterocedásticos.

4.2.1. Procesos ARCH

Los modelos de heterocedasticidad condicional autorregresivos (*ARCH*) fueron presentados por Engle (1982) [11].

Definición 4.2.1 (Proceso ARCH(q)) *Un proceso estocástico $\{\varepsilon_t\}$ es un proceso ARCH(q) si $\varepsilon_t|\mathscr{F}_{t-1} \in L^2$ y satisface:*

i) $\mathbb{E}[\varepsilon_t|\mathscr{F}_{t-1}] = 0 \quad \forall t \in \mathbb{Z}$.

ii) Existen constantes $\omega > 0, \alpha_i \geq 0$, $i = 1,...,q$, tales que

$$\sigma_t^2 = Var[\varepsilon_t|\mathscr{F}_{t-1}] = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 \quad \forall t \in \mathbb{Z}. \tag{4.1}$$

Definición 4.2.2 (Proceso ARCH(q) fuerte) *Sea $\{\eta_t\}$ un proceso de ruido blanco tal que $Var[\eta_t] = 1$. Un proceso estocástico $\{\varepsilon_t\}$ es un proceso ARCH(q) fuerte si:*

i) $\varepsilon_t = \sigma_t \eta_t \quad \forall t \in \mathbb{Z}$.

ii) Existen constantes $\omega > 0, \alpha_i \geq 0$, $i = 1,...,q$ tales que

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 \quad \forall t \in \mathbb{Z}.$$

Sea $\{\varepsilon_t\}$ un proceso ARCH(q) débil. Para hallar la condición que debe cumplir el proceso para ser débilmente estacionario se utilizará la siguiente observación:

$$\mathbb{E}\left[\varepsilon_t^2\right] = \mathbb{E}\left[\mathbb{E}\left[\varepsilon_t^2|\mathscr{F}_s, s < t\right]\right] = \mathbb{E}\left[\sigma_t^2\right],$$

lo cual no depende del tiempo.

Es inmediato ver que $Cov[\varepsilon_t, \varepsilon_{t+h}] = 0$ si $h > 0$. Cuando $h = 0$,

$$Var\left[\varepsilon_t^2\right] = \mathbb{E}\left[\varepsilon_t^2\right] = \omega + \sum_{i=1}^{q} \alpha_i \mathbb{E}\left[\varepsilon_{t-i}^2\right] = \omega + \mathbb{E}\left[\varepsilon_t^2\right] \sum_{i=1}^{q} \alpha_i.$$

Así,

$$\omega = \mathbb{E}\left[\varepsilon_t^2\right]\left(1 - \sum_{i=1}^{q} \alpha_i\right).$$

Por lo tanto, un proceso ARCH(q) es débilmente estacionario si

$$\sum_{i=1}^{q} \alpha_i < 1.$$

La ecuación (4.1) se puede expresar como

$$\sigma_t^2 = \omega + \alpha(B)\varepsilon_t^2,$$

donde

$$\alpha(B) = \sum_{i=1}^{q} \alpha_i B^i.$$

Los parámetros se pueden ajustar por máxima verosimilitud. En la siguiente sección se ejemplificará, ya que los procesos mostrados a continuación son más generales que los ARCH.

4.2.2. Procesos GARCH

El proceso GARCH(p,q) fue propuesto por Bollerslev (1986). Es una generalización del proceso ARCH(q) [5].

Definición 4.2.3 (Proceso GARCH(p,q)) *Un proceso estocástico $\{\varepsilon_t\}$ es un proceso GARCH(p,q) si $\varepsilon_t|\mathscr{F}_{t-1} \in L^2$ y satisface:*

i) $\mathbb{E}[\varepsilon_t|\mathscr{F}_{t-1}] = 0 \quad \forall t \in \mathbb{Z}$.

ii) Existen constantes $\omega \geq 0$ y $\alpha_i, i = 1,...,q, \beta_j > 0, j = 1,...,p$, tales que

$$\sigma_t^2 = Var[\varepsilon_t|\mathscr{F}_{t-1}] = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 \quad \forall t \in \mathbb{Z}. \tag{4.2}$$

Definición 4.2.4 (Proceso GARCH(p,q) fuerte) *Sea $\{\eta_t\}$ un proceso de ruido blanco tal que $Var[\eta_t] = 1$. Un proceso estocástico $\{\varepsilon_t\}$ es un proceso GARCH(p,q) fuerte si:*

i) $\varepsilon_t = \sigma_t \eta_t \quad \forall t \in \mathbb{Z}$.

ii) Existen constantes $\omega \geq 0$ y $\alpha_i, i = 1,...,q, \beta_j > 0, j = 1,...,p$, tales que

$$\sigma_t^2 = \omega + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 \quad \forall t \in \mathbb{Z}.$$

Sea $\{\varepsilon_t\}$ un proceso GARCH(p,q) débil. La condición que debe cumplir el proceso para ser débilmente estacionario se obtienen de forma análoga al caso del proceso ARCH. Es inmediato ver que $Cov[\varepsilon_t, \varepsilon_{t+h}] = 0$ si $h > 0$. Cuando $h = 0$,

$$\begin{aligned} Var\left[\varepsilon_t^2\right] &= \mathbb{E}\left[\varepsilon_t^2\right] \\ &= \omega + \sum_{i=1}^{p} \beta_i \mathbb{E}\left[\sigma_{t-i}^2\right] + \sum_{i=1}^{q} \alpha_i \mathbb{E}\left[\varepsilon_{t-i}^2\right] \\ &= \omega + \mathbb{E}\left[\varepsilon_t^2\right]\left(\sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i\right). \end{aligned}$$

Así,

$$\omega = \mathbb{E}\left[\varepsilon_t^2\right]\left(1 - \sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i\right).$$

Por lo tanto, un proceso GARCH(p,q) es débilmente estacionario si

$$\sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i < 1.$$

El proceso ARCH(p) se obtiene cuando $q = 0$. Si $\sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i = 1$, se denomina proceso GARCH Integrado o IGARCH(p,q).[1]

La ecuación (4.2) se puede expresar como

$$\sigma_t^2 = \omega + \alpha(B)\varepsilon_t^2 + \beta(B)\sigma_t^2, \tag{4.3}$$

donde

$$\alpha(B) = \sum_{i=1}^{q} \alpha_i B^i \text{ y } \beta(B) = \sum_{i=1}^{p} \beta_i B^i.$$

La innovación del proceso $\{\varepsilon_t^2\}$ es el proceso estocástico $\{v_t\}$, donde $v_t = \varepsilon_t^2 - \sigma_t^2$. Sustituyendo v_t en la ecuación (4.2) se obtiene

$$\varepsilon_t^2 = \omega + \sum_{i=1}^{m} (\alpha_i + \beta_i)\varepsilon_{t-i}^2 + v_t - \sum_{i=1}^{p} \beta_i v_{t-i}^2,$$

la cual es la estructura de un proceso ARMA(m,q), donde $m = max\{p,q\}$. Si las raíces de $1 - \beta(B)$ se encuentran fuera del círculo unitario, de la ecuación (4.3) se obtiene:

$$\begin{aligned} \sigma_t^2 &= \omega + \beta(B)\sigma_t^2 + \alpha(B)\varepsilon_t^2 \\ &= \omega(1 - \beta(B))^{-1} + (1 - \beta(B))^{-1}\alpha(B)\varepsilon_t^2 \\ &= \omega(1 - \beta(B))^{-1} + \sum_{i=1}^{\infty} \delta_i \varepsilon_{t-i}^2, \end{aligned} \tag{4.4}$$

[1] Para mayor información se recomienda consultar [13].

donde

$$\delta_i = \alpha_i \mathbb{I}(i)_{i\leq q} + \sum_{j=1}^{r} \beta_j \delta_{j-i},$$

y $r = min\{p, i-1\}$. La ecuación (4.4) es la representación de un AR(∞), por lo que un proceso GARCH se puede ver como un ARCH(∞).

Estimación de Parámetros

Existen diversas formas de estimar los parámetros. Las más conocidas son por momentos y por máxima verosimilitud. La función de verosimilitud es:

$$f(\varepsilon_t|\mathscr{F}_{t-1}) = \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(-\frac{1}{2\sigma_t^2}\varepsilon_t^2\right).$$

Por otro lado,

$$f(\varepsilon_t, ..., \varepsilon_1|\varepsilon_0) = \prod_{i=1}^{t} f(\varepsilon_i|\mathscr{F}_{t-1}),$$

entonces,

$$\begin{aligned} \log f(\varepsilon_t, ..., \varepsilon_1|\varepsilon_0) &= \sum_{i=1}^{t} \log f(\varepsilon_i, \varepsilon_{i-1}, ..., \varepsilon_1|\varepsilon_0) \\ &= -\frac{t}{2}\log 2\pi - \frac{1}{2}\sum_{i=1}^{t}\log\sigma_i^2 - \frac{1}{2}\sum_{i=1}^{t}\frac{\varepsilon_i^2}{\sigma_i^2}. \end{aligned} \tag{4.5}$$

Los estimadores máximo verosímiles de α_i, β_i son aquellos que maximizan (4.5).

4.2.3. Procesos EGARCH

En los procesos anteriores no existe asimetría en la varianza a causa de efectos tanto positivos como negativos. El proceso EGARCH (*Exponential GARCH*) fue propuesto por Nelson (1991).

Definición 4.2.5 (Proceso EGARCH(p,q)) *Sea $\{\eta_t\}$ un proceso de ruido blanco tal que $Var[\eta_t] = 1$. Un proceso estocástico $\{\varepsilon_t\}$ es un proceso EGARCH(p,q) si satisface:*

i) $\varepsilon_t = \sigma_t \eta_t \quad \forall t \in \mathbb{Z}$.

ii) Existen constantes $\omega \geq 0$, θ, $\gamma \in \mathbb{R}$ y $\alpha_i, i = 1, ..., q$, $\beta_j > 0, j = 1, ..., p$, tales que

$$\log\sigma_t^2 = \omega + \sum_{i=1}^{q}\alpha_i g(\eta_{t-i}) + \sum_{i=1}^{p}\beta_i \log\sigma_{t-i}^2 \quad \forall t \in \mathbb{Z}, \tag{4.6}$$

donde

$$g(\eta_{t-i}) = \theta\eta_{t-i} + \gamma(|\eta_{t-i}| - \mathbb{E}[|\eta_{t-i}|]). \tag{4.7}$$

g permite cambios asimétricos dependiendo del signo de $\{\eta_{t-i}\}_{i>0}$, por lo que la presencia de efectos negativos tiene efectos distintos a la de los positivos. Si $g \neq 0$ casi seguramente y los polinomios de retraso $\alpha(B) = \sum_{i=1}^{q} \alpha_i B^i$ y $\beta(B) = \sum_{i=1}^{p} \beta_i B^i$ no tienen raíces comunes, entonces un proceso EGARCH(p,q) es estrictamente estacionario si y sólo si las raíces de $\beta(B)$ se encuentran fuera del círculo unitario.[2]

De las ecuaciones (4.6) y (4.7) se obtiene:

$$\begin{aligned}
\log(\sigma_t^2) &= \omega + \sum_{i=1}^{p} \alpha_i \log \sigma_{t-i}^2 + \sum_{i=1}^{q} \beta_i g(\eta_{t-i}) \\
&= \omega + \sum_{i=1}^{p} \alpha_i \log \sigma_{t-i}^2 + \sum_{i=1}^{q} \beta_i \left(\theta \eta_{t-i} + \gamma(|\eta_{t-i}| - \mathbb{E}[|\eta_{t-i}|])\right) \\
&= \omega + \sum_{i=1}^{p} \alpha_i \log \sigma_{t-i}^2 + \sum_{i=1}^{q} \beta_i \left(\theta \frac{\varepsilon_{t-i}}{\sigma_{t-i}} + \gamma\left(\left|\frac{\varepsilon_{t-i}}{\sigma_{t-1}}\right| - \mathbb{E}\left[\frac{\varepsilon_{t-i}}{\sigma_{t-1}}\right]\right)\right),
\end{aligned}$$

de manera que, cuando $0 < \eta_t < \infty$, $g(\eta_t)$ es lineal en η_t con pendiente $(\theta + \gamma)$; cuando $-\infty < \eta_t < 0$, $g(\eta_t)$ es lineal en η_t con pendiente $(\theta - \gamma)$.

Si $\eta_t \sim N(0,1)$, entonces

$$\log(\sigma_t^2) = \omega + \sum_{i=1}^{p} \alpha_i \log \sigma_{t-i}^2 + \sum_{i=1}^{q} \beta_i \left(\theta \frac{\varepsilon_{t-i}}{\sigma_{t-i}} + \gamma\left(\left|\frac{\varepsilon_{t-i}}{\sigma_{t-1}}\right| - \sqrt{\frac{2}{\pi}}\right)\right).$$

La estimación de los parámetros puede ser por máxima verosimilitud.

[2] La demostración se puede encontrar en [13].

Capítulo 5

Tasa de Cambio

La tasa de cambio o paridad establece la equivalencia de unidades de una moneda con respecto a otra. En otras palabras, expresa la cantidad de unidades de una moneda que se requieren para comprar otra. Supóngase que la tasa de cambio dólar-peso (USD/MXN) es de 10.00, por lo que se requieren 10 pesos (10 MXN) para comprar 1 dólar (1 USD).

5.1. Régimen Cambiario

La tasa de cambio de cada país está determinada por el régimen cambiario, que es la forma en que se administra el valor de una moneda respecto a otras. Los tres tipos de regímenes cambiarios más conocidos son: fijo, de libre flotación o flexible y de bandas cambiarias [1].

En el régimen de libre flotación, la tasa de cambio se determina a través de la oferta y la demanda. La tasa de cambio dólar-peso se calcula bajo este régimen.

En el régimen fijo, se establece un nivel de la tasa de cambio, y la autoridad monetaria se compromete, interviniendo en el mercado comprando o vendiendo divisas, para garantizar que esta tasa de cambio se mantenga en ese nivel.

El régimen de bandas cambiarias es un intermedio entre los anteriores. La autoridad monetaria fija una banda en la cual deja que la tasa de cambio se mueva libremente. Cuando la tasa de cambio alcanza los extremos de la banda, la autoridad interviene vendiendo o comprando divisas para mantenerlo en la banda.

5.2. Tipos de Cambio

Debido a que la tasa de cambio dólar-peso cambia a través el tiempo, es necesario contar con alguna referencia cuyo valor sea fijo durante cierto periodo de tiempo, ya que de otro modo, las obligaciones pagaderas en México denotadas en dólares podrían liquidarse en un instante de tiempo que beneficie al deudor, dentro de un periodo pactado. Por ello, existe dos tipos de cambio: el tipo de cambio FIX y el tipo de cambio spot.

El tipo de cambio FIX es calculado por la Oficina de Cambios Nacionales del Banco de México con base al promedio de las cotizaciones del mercado de cambios al mayoreo y que son obtenidas de medios electrónicos con representatividad en el mercado de cambios. Estas cotizaciones se obtienen tres veces al día, entre las 9:00 am y las 12:00 pm. Es publicado en la página electrónica del Banco de México a partir de las 12:00 pm y el día hábil siguiente en el Diario Oficial de la Federación (DOF) y es utilizado para solventar obligaciones denominadas en dólares liquidables en la República Mexicana al día siguiente de la publicación en el DOF.[1]

Por otro lado, el tipo de cambio spot alude al tipo de cambio actual por lo que, bajo un régimen cuya tasa de cambio se ve afectada por la oferta y la demanda, su valor cambia a través del tiempo. Eventos imprevistos pueden provocar que el mercado se mueva en cuestión de segundos de forma ”brusca”, afectando a la tasa de cambio, entre otras variables económicas.

5.3. Modelo de Black-Scholes

Antes de continuar es imprescindible definir algunos conceptos. Un derivado es un activo cuyo valor depende de otro activo denominado subyacente. Entre los derivados más consolidados en el mercado se encuentran los *forwards* y las opciones, éstas últimas pueden ser del tipo *call* o *put*.

Un *forward* es un contrato entre dos partes: la parte corta y la parte larga. La parte corta tiene la obligación de comprar cierta cantidad (o cierto número) de unidades de subyacente a un precio pactado o strike K en la fecha de maduración T, mientras que la parte larga tiene la obligación de vender la cantidad (o número) pactada de subyacente al precio K en el tiempo T. Las opciones se pueden clasificar de acuerdo al momento en que pueden ser ejercidas.

Una opción europea de tipo *call* con precio de ejercicio K fecha de maduración T es un contrato que le otorga a su poseedor llamado parte larga el derecho mas no la obligación de comprar una unidad de subyacente al tiempo y precio pactados, T y K, respectivamente. En el caso de una opción de tipo *put*, se le otorga el derecho mas no la obligación a la parte larga de vender una unidad de subyacente al tiempo y precio pactados. La parte que tiene la obligación de vender o comprar la unidad de subyacente recibe el nombre de parte corta.

[1] Para mayor información acerca de la tasa de cambio FIX consulte [12].

Una opción americana puede ser ejercida en cualquier momento t, donde $0 \leq t \leq T$.

Debido a la existencia de incertidumbre en el valor de la tasa de cambio y, al ser desconocida la medida de probabilidad del mundo real, suena tentador apostar por una subida en la tasa de cambio entre las dos divisas que están involucradas a favor de alguna de éstas.

El rendimiento de un activo se obtiene dividiendo su *payoff*, X_{t+1}, por su precio, p_t, por lo que el rendimiento está dado por:

$$R_{t+1} = \frac{X_{t+1}}{p_t}.$$

Los rendimientos expresados de esta forma reflejan la proporción del valor a tiempo $t+1$ con respecto a su precio en t, por lo que es común considerar la tasa $r_{t+1} = R_{t+1} - 1$, la cual representa la proporción de la diferencia entre el *payoff* de un activo al tiempo $t+1$ y su precio al tiempo t con respecto a su precio al tiempo t.

Considérese una composición continua lognormal en la tasa de interés y un crecimiento lognormal. El proceso de los rendimientos logarítmicos $\{r_t\}$ se define como

$$r_t := \log(R_t).$$

Se denotará como **P** a la medida de probabilidad del mundo real. Dados W_t un **P**-movimiento Browniano, y μ y σ variables determinísticas, el modelo de Black-Scholes (básico) está conformado por las ecuaciones:

$$\begin{aligned} S_t &= S_0 \exp(\mu t + \sigma W_t), \\ B_t &= \exp(rt), \end{aligned}$$

donde r es la tasa de interés libre de riesgo, B_t es el precio del bono a tasa r del tiempo 0 al tiempo t y S_t es el precio del *stock* al tiempo t. μ y σ representan la tendencia y la volatilidad del *stock*, respectivamente. El crecimiento del valor del subyacente a tasa r puede complicar los cálculos, por lo que es conveniente considerar el proceso $Z_t = B_t^{-1} S_t$. Para simplificar los cálculos, se supondrá $r = 0$. Es fácil ver que S_t satisface la ecuación diferencial estocástica (*SDE*):

$$d \log S_t = \mu dt + \sigma dW_t. \tag{5.1}$$

Para resolver la ecuación diferencial estocástica (5.1) se utilizará el siguiente lema:

Lema 5.3.1 (Lema de Itô) *Sea $\{X_t\}$ un proceso estocástico que satisface la ecuación diferencial estocástica $dX_t = \mu_t dt + \sigma_t dW_t$, y f una función determinística cuya segunda derivada es continua, entonces $Y_t := f(X_t)$ satisface la ecuación diferencial estocástica:*

$$dY_t = \left(\mu_t f'(X_t) + \frac{1}{2} \sigma_t^2 f''(X_t) \right) dt + \sigma_t f'(X_t) dW_t$$

Considérese $f(x) = e^x$. De la *SDE* (5.1) y por el lema de Itô:

$$\begin{aligned} df(\log S_t) = dS_t &= \left(\mu + \frac{1}{2}\sigma^2\right) S_t dt + \sigma S_t dW_t \\ &= \left(\mu_t + \frac{1}{2}\sigma_t^2\right) dt + \sigma_t dW_t \\ &= \tilde{\mu}_t dt + \sigma_t dW_t, \end{aligned} \tag{5.2}$$

donde $\mu_t = \mu S_t$, $\sigma_t = \sigma S_t$ y $\tilde{\mu}_t = \mu_t + \frac{\sigma_t^2}{2}$. Se busca una medida $\mathbf{Q}$ tal que la tendencia de S_t sea ν_t. La ecuación (5.2) se puede reescribir como:

$$dS_t = \nu_t + \sigma_t \left(\left(\frac{\tilde{\mu}_t - \nu_t}{\sigma_t} \right) dt + dW_t \right). \tag{5.3}$$

Cuando se calcula el precio de un derivado, trabajar con alguna medida tal que el proceso $Z_t = B_t^{-1} S_t$ sea $\mathbf{Q}-$martingala facilita los cálculos. El teorema de Girsanov[21] garantiza la existencia de esta medida bajo ciertas condiciones.[2]

Teorema 5.3.2 (Teorema de Girsanov) *Sean W_t un $\boldsymbol{P}-$movimiento Browniano y $\mathscr{F}_t$ la información hasta tiempo t. Dado un proceso $\{\gamma_t\}$ adaptado a la filtración $\mathscr{F}_t$ que satisface la condición*

$$\mathbb{E}_{\boldsymbol{P}} \left[\exp\left\{ -\frac{1}{2} \int_0^T \gamma_t^2 dt \right\} \right] < \infty \tag{5.4}$$

Entonces:

i) $\boldsymbol{Q}$ *es equivalente a* $\boldsymbol{P}$.

ii) $\dfrac{d\boldsymbol{Q}}{d\boldsymbol{P}} = exp\left\{ -\displaystyle\int_0^T \gamma_t dW_t - \frac{1}{2} \int_0^T \gamma_t^2 dt \right\}$.

iii) $\widetilde{W}_t = W_t + \int_0^t \gamma_s ds$ *es un* $\boldsymbol{Q}$*-movimiento Browniano.*

Sea $\gamma_t = \frac{\tilde{\mu}_t - \nu_t}{\sigma_t}$. Por el teorema de Girsanov, la ecuación (5.3) es equivalente a

$$dS_t = \nu_t dt + \sigma_t \widetilde{W}_t,$$

donde $\widetilde{W}_t = W_t + \int_0^t \gamma_s ds$ es un $\mathbf{Q}$-movimiento Browniano. Es fácil verificar que

$$S_t = S_0 \exp\left\{ \left(v - \frac{1}{2}\sigma^2 \right) t + \sigma \widetilde{W}_t \right\},$$

y que además, si $v_t = 0$, S_t es martingala bajo $\mathbf{Q}$. De forma inmediata, si $h > 0$, se tienen los siguientes resultados:

i) $r_{t-h,t} = \log \dfrac{S_t}{S_{t-h}} = \mu h + \sigma W_h$.

[2] La demostración del teorema se puede encontrar en [7].

ii) Si $\gamma_t = \tilde{\mu}_t/\sigma_t$, $\tilde{\mu}_t = \mu_t + \frac{\sigma_t^2}{2}$ satisface la condición (5.4), entonces:

$$r_{t-h,t} = \log \frac{S_t}{S_{t-h}} = \left(v - \frac{1}{2}\sigma^2\right) h + \sigma \widetilde{W}_h.$$

iii) Bajo **P**: $r_{t-h,t} \sim N(\mu h, \sigma^2 h)$.

iv) Bajo **Q**: $r_{t-h,t} \sim N((v - \frac{1}{2}\sigma^2)h, \sigma^2 h)$.

Si se desea profundizar en los temas expuestos hasta el momento de la actual sección, se recomienda consultar [3].

Supóngase que el subyacente es la tasa de cambio entre dos divisas. Sea $\{C_t\}$ el proceso estocástico del valor de la tasa de cambio entre las dos divisas a tiempo t. El modelo de Black-Scholes para el mercado cambiario se obtiene considerando como el subyacente a tiempo t, a la tasa de cambio a tiempo t, C_t, y se consideran los bonos tanto de moneda extranjera como de moneda local con sus correspondientes tasas de interés libres de riesgo:

$$\begin{aligned} C_t &= C_0 \exp(\mu t + \sigma W_t), \\ B_t &= \exp(rt), \\ D_t &= \exp(ut), \end{aligned}$$

donde B_t, D_t son los precios de los bonos para la moneda local y extranjera, y r y u sus respectivas tasas de interés libres de riesgo.

En la vida real, la tendencia y la volatilidad de la paridad entre dos divisas no es constante, ya que en un mes del año esta paridad puede haber fluctuado en rangos más amplios y de forma más agresiva que en otro, lo cual es un problema del modelo de Black-Scholes. Considérese al subyacente como el valor de la tasa de cambio en un momento específico del día entre dos divisas cuyo régimen cambiario es el flotante, el cual se denotará como $\{C_t\}$. El momento puede ser el cierre o la apertura, por ejemplo, y la temporalidad puede ser diaria, semanal, mensual, etc. Sean $\{\mu_t\}$ y $\{\sigma_t\}$ los procesos del valor de la tendencia y la volatilidad. Se propone que C_t satisface la ecuación:

$$C_t = C_{t-1} e^{\mu_{t-1} + \sigma_{t-1}\eta_{t-1}}, \quad \eta_t \sim N(0,1). \tag{5.5}$$

El problema de este modelo radica en el desconocimiento de la distribución de los procesos de tendencia $\{\mu_t\}$ y volatilidad $\{\sigma_t\}$. Una solución sería emplear el enfoque bayesiano, proponiendo distribuciones *a priori* para μ_t y σ_t y, posteriormente, obtener una función de distribución *a posteriori* para el proceso C_t denotada como $\mathbb{P}(C_t|\mu_{t-1}, C_{t-1}, C_t)$. El estimador máximo bayesiano de σ_{t-1} correspondería a $\mathbb{E}[\sigma_{t-1}|\mu_{t-1}, C_{t-1}, C_t]$.

Otra opción es deshacerse de la tendencia, por ejemplo, mediante un cambio de medida tal que la tendencia sea cero. Así,

$$C_t = C_{t-1} e^{\sigma_{t-1}\eta_{t-1}}. \tag{5.6}$$

Sin embargo, η_t no necesariamente sigue una distribución $N(0,1)$ debido a que se pueden presentar colas pesadas en la distribución de los rendimientos logarítmicos, por lo que no se puede utilizar el teorema de Girsanov para deshacerse de la tendencia, por lo que se propone que η_t corresponde a alguna otra distribución del error como la t-Student o la distribución del error generalizada (normal generalizada o *power exponential*), entre otras, la cual a su vez puede incluir a la tendencia μ_t como una constante para todo $t \geq 0$.

Desarrollando (5.5):

$$\begin{aligned} r_{t+1} &= \mu_t + \sigma_t \eta_t \\ &= r_t + \mu_t + \sigma_t \eta_t - \mu_{t-1} - \sigma_{t-1}\eta_{t-1} \\ &= r_t + \ldots + r_{t-p+1} + \mu_t + \sigma_t \eta_t - \ldots - \mu_{t-p} - \sigma_{t-p}\eta_{t-p}. \end{aligned}$$

Se puede apreciar una similitud con un proceso ARMA. Tiene sentido suponer que la tendencia no cambia ya que en la vida real, para periodos de tiempo cortos, la tendencia es constante. Generalizando, se obtiene que los rendimientos satisfacen la dinámica:

$$\phi_p(B) r_{t+1} = \theta_q(B)\sigma_t \eta_t \tag{5.7}$$

Así, otra propuesta para el proceso $\{r_t\}$ es que corresponde a un proceso ARMA(p_1, q_1) y $\varepsilon_t = \sigma_t \eta_t$ un GARCH(p_2, q_2):

$$\begin{aligned} \phi_{p_1}(B) r_t &= \theta_{q_1}(B)\varepsilon_t, \\ \varepsilon_t &= \sigma_t \eta_t, \\ \sigma_t^2 &= \omega + \sum_{i=1}^{q_2} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p_2} \beta_i \sigma_{t-i}^2. \end{aligned}$$

El proceso GARCH puede ser reemplazado por algún otro proceso heterocedástico.

Capítulo 6

Predicción de la Volatilidad de la Paridad USD/MXN

En el presente capítulo se analizarán los datos históricos de la tasa de cambio dólar-peso y, posteriormente, se ajustarán diversos modelos empleando tanto el enfoque clásico como el de *machine learning*. Así mismo, se propondrán modelos que utilizarán la información proporcionada por alguno de los modelos ajustados bajo el enfoque clásico como entrada adicional a uno ajustado bajo el enfoque de *machine learning*. Se seleccionará un modelo para uno de los tipos de modelo ajustados para luego ser comparados. A este tipo de modelos que requieren de dos modelos ajustados bajo enfoques estadísticos distintos se les denominará modelos híbridos.

6.1. Planteamiento del Problema

Como el mercado siempre está en movimiento, existe incertidumbre asociada al valor de la tasa de cambio entre dos divisas cuyo valor depende de la oferta y la demanda, ya sea de forma favorable o negativa, y debido a la sensibilidad del mercado, el valor de la tasa de cambio puede sufrir cambios más agresivos en un periodo de tiempo determinado que en otro. Para casos en los que la tasa de cambio tiene o se prevé que tenga mucha volatilidad se recomienda comprar un derivado, por ejemplo, una opción sobre el tipo de cambio. Puede ser un buen momento para obtener cierta ganancia si se realizan buenas estimaciones y se está consciente del riesgo que ello conlleva.

Es de suma importancia considerar el sentimiento que se tiene en los mercados ya que, si se tienen expectativas favorables, habrá menos incertidumbre, por lo que la variable a predecir será más estable.

Si se pudiera predecir el comportamiento de la tasa de cambio entre dos divisas cuyo valor depende del mercado, por ejemplo, la tasa de cambio USD/MXN, se podría evitar tener grandes pérdidas, por lo que tener cierto nivel de confianza de su volatili-

dad es de gran utilidad. Debido a la volatilidad que existe en los mercados no es posible predecir el valor de los indicadores económicos con exactitud. Sin embargo, es posible realizar pronósticos mediante modelos matemáticos, teniendo en cuenta la incertidumbre que existe en el mundo. Debido a que el valor de la tasa de cambio USD/MXN no se puede predecir, los derivados se vuelven una opción atractiva para las situaciones en las que el valor de dicha tasa de cambio está involucrado. Sin embargo, modelos como el de Black-Scholes requieren de la volatilidad del subyacente, en este caso de la tasa de cambio USD/MXN, la cual no es observable. Por ello, debe ser estimada.

Supóngase que se cuenta con valores de la tasa de cambio en tiempos discretos. El valor o momento puede ser cualquiera como el de apertura, cierre, FIX, máximo y mínimo diarios. Se consideró el valor al cierre debido a que permite tener un panorama de las ganancias o pérdidas de un día respecto a otro.

Es importante notar que, en cierto periodo, el valor de la tasa de cambio puede fluctuar de forma más agresiva en un periodo de tiempo, ya sea un mes o un trimestre, que en otro. Se propuso que la tasa de cambio sigue una dinámica estocástica sin tendencia como la expresada en la ecuación (5.6):

$$C_t = C_{t-1} e^{\sigma_{t-1}\eta_{t-1}},$$

donde la distribución de $\{\eta_t\}$ no necesariamente es normal y $\{\sigma_t\}$ es un proceso asociado a la volatilidad de la tasa de cambio USD/MXN al tiempo t. De la ecuación anterior se obtiene el proceso a tiempo discreto de los rendimientos, $\{r_t\}$, al tiempo t:

$$r_t = log\frac{C_t}{C_{t-1}} = \sigma_{t-1}\eta_{t-1}. \tag{6.1}$$

El modelo clásico de Black-Scholes clásico supone que $\sigma_t = \sigma$ es constante. Existen diversas formas de estimar dicha volatilidad. La forma tradicional más simple de estimar la volatilidad es tomar el conjunto total de las observaciones de rendimientos y mediante la siguiente expresión:

$$\hat{\sigma} = \frac{1}{n-1}\sum_{i=1}^{n}(r_i - \bar{r})^2,$$

donde n es el tamaño de la muestra de los rendimientos logarítmicos y $\bar{r} = \frac{1}{n}\sum_{i=1}^{n} r_i$. Esta forma de estimar la volatilidad no es adecuada por lo que se comentó anteriormente (no es constante).

Los modelos heteroscedásticos de series de tiempo son adecuados para $\{r_t\}$, por lo que se consideraron los modelos GARCH y EGARCH bajo el enfoque clásico. A su vez, se considero el enfoque de *machine learning*. Al ser el proceso $\{r_t\}$ una serie de tiempo, se optó por los modelos de redes neuronales recurrentes LSTM y GRU.

Los modelos propuestos bajo ambos enfoques no cuentan con los mismos supuestos, además de que la forma en la que están constituidos es diferente. En el caso de los modelos de series de tiempo anteriormente propuestos, se supone que los residuales siguen cierta distribución, mientras que en los modelos de redes neuronales, no es

necesario suponer que los residuales siguen una distribución en específico. Por otro lado, en el enfoque de *machine learning*, es fundamental contar con una gran cantidad de datos, mientras que en enfoque clásico, si bien ello mejora las estimaciones de los parámetros, no es fundamental.

Por ello, se propuso un modelo que combina ambos enfoques. Al obtener información de un modelo bajo uno de los enfoques y emplearla como entrada adicional para otro modelo bajo otro enfoque, la eficiencia de la tarea **T** podría mejorar. Al modelo conformado por dos modelos cuya estimación de sus respectivos parámetros es bajo enfoques diferentes se les denominó *modelos híbridos*. A los modelos que únicamente constan de uno solo se les denominó *modelos individuales*. Cuando se mencione la eficiencia de un modelo, se referirá a la eficiencia del modelo al generalizar, es decir, al realizar la tarea **T**.

Así, se propusieron los modelos híbridos tales que la información de los modelos de series de tiempo, bajo el enfoque clásico, se utilice como entrada adicional a los modelos de redes neuronales. Dicho de otra forma, los modelos de series de tiempo bajo el enfoque clásico alimentan una red neuronal.

Se propuso un modelo individual para cada uno de los modelos GARCH, EGARCH, LSTM y GRU; y uno híbrido conformado por un modelo de serie temporal y una red neuronal recurrente, para cada tipo de red neuronal, de modo que los parámetros de la serie temporal corresponden a la entrada de una red neuronal, por lo que se compararon 6 modelos en total. Los modelos híbridos propuestos se conformaron por las combinaciones posibles entre los modelos de series de tiempo GARCH y EGARCH y los de redes neuronales LSTM y GRU, así como uno que combina ambos modelos de series temporales con ambos tipos de redes.

El conjunto de datos **E** se conformó por las volatilidades estimadas de un cierto periodo, para cada tiempo t. La tarea **T** corresponde a la predicción de la volatilidad al tiempo $t+1$. Como la volatilidad tiende a estabilizarse, las predicciones que se consideraron fueron a un día.

Para la estimación de las volatilidades, se consideraron ventanas de 5, 10 y 22 días. Se denotará como $\hat{\sigma}_{d,t}$ a la volatilidad estimada considerando una ventana de d días a tiempo t, la cual se calculó como:

$$\hat{\sigma}_{d,t} = \frac{1}{d-1} \sum_{i=0}^{d-1} (r_{t-i} - \bar{r}_{d,t})^2,$$

donde $\bar{r}_{d,t} = \frac{1}{d} \sum_{i=0}^{d-1} r_{t-i}$. Se referirá a la volatilidad estimada con una ventana de d días como volatilidad de d días con el fin de simplificar nombres. Además, se aludirá a la volatilidad estimada mediante la ecuación (6.1) como volatilidad observada (aunque no es observable).

Debido a que, a mayor tamaño de las ventanas, mayor estabilidad de la volatilidad, se consideró la ventana de 5 días como la de referencia, con la cual se ajustaron los parámetros de los modelos. Para considerar el valor estimado de la volatilidad, es más adecuado tomar en cuenta la ventana de 22 días, la cual corresponde a 28 días (aproximadamente un mes). Sin embargo, se ajustaron los modelos considerando la volatilidad

estimada bajo una ventana de 5 días, ya que es la menos estable que la estimada bajo el resto de las ventanas propuestas.

El intervalo de confianza al $100(1-\alpha)\,\%$ para un parámetro θ se construyó como

$$\left(\hat{\mu}_\theta - \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_\theta, \hat{\mu}_\theta + \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_\theta\right), \tag{6.2}$$

donde $\hat{\mu}_\theta$ y $\hat{\sigma}_\theta$ son la media estimada y la desviación estándar estimada de θ y $\Phi^{-1}(1-\alpha/2)$ es el cuantil al $100(1-\alpha/2)\,\%$ de una distribución $N(0,1)$. Se suele considerar que $\Phi^{-1}(1-0.05/2) = 1.96$.

La significancia de los coeficientes de los modelos de series de tiempo fue con base en la creación de intervalos de confianza al $100(1-\alpha)\,\%$, es decir, para el estimador $\hat{\theta}$ del parámetro θ, si el cero está contenido en el intervalo de confianza al $100(1-\alpha)\,\%$ de $\hat{\theta}$ entonces $\hat{\theta}$ es estadísticamente no significativo. El nivel de significancia considerado por defecto fue de $5\,\%$, a menos que se especifique lo opuesto.

De forma aproximada, el intervalo que contiene el $100\beta\,\%$ de los datos de un conjunto X se calculó mediante el intervalo de confianza al $100\beta\,\%$, mientras que, de forma exacta, el intervalo se construyó de la forma $\left(q_{1-\frac{\beta}{2}}, q_{\frac{\beta}{2}}\right)$, donde $q_{1-\frac{\beta}{2}}$ y $q_{\frac{\beta}{2}}$ representan los cuantiles al $100\left(1-\frac{\beta}{2}\right)\,\%$ y $100\frac{\beta}{2}\,\%$, respectivamente. Cuando no se especifique si el $100\beta\,\%$ de los datos se encuentran de forma exacta o aproximada, quedará implícito que es de forma exacta.

Uno de los conceptos que se utilizará es el del *Value at Risk* (*VaR*). El *VaR* de una variable aleatoria X al $\alpha\,\%$ es el cuantil que acumula el $\alpha\,\%$ de la probabilidad de X.

Se utilizó el lenguaje de programación R. Existen diversas librerías para desarrollar y ajustar modelos bajo el enfoque de *machine learning*. Se eligió emplear *Tensorflow*, desarrollada por Google®, y la *API* Keras. Keras es una *API* de alto nivel de redes neuronales que permiten utilizar librerías como *Tensorflow*, *CNTK*, o *Theano*. Estas librerías están desarrolladas en Python. La *API Keras* está desarrollada tanto en Python como en R. Esta *API* facilita el desarrollo y entrenamiento de redes neuronales.

Debido a la cantidad de recursos computacionales y el tiempo requerido en el entrenamiento de las redes neuronales, las librerías de *machine learning* aprovechan los recursos disponibles como el número de núcleos o *GPU* si es que ésta lo permite. Por tanto, se optó por no fijar la semilla en las tareas relacionadas a *machine learning* ya que se aprovecharán todos los núcleos de la *CPU*. En el resto de los procesos ajenos al entrenamiento de los modelos de *machine learning* se fijó la semilla. *Tensorflow* permite guardar los modelos ajustados en un archivo `.h5`, lo cual permite recuperarlos en tiempos posteriores.

Para ajustar los parámetros de los modelos heterocedásticos mediante el enfoque clásico se utilizó la librería `rugarch`.

Debido a la extensión del código empleado para el presente capítulo, se decidió publicarlo, al igual que los modelos entrenados y el entorno de las variables de R, en un repositorio de GitHub®, el cual se encuentra ubicado en

˜https://github.com/Yizuzhub/.[1]

6.2. Análisis Exploratorio

El histórico de las tasas de cambio USD/MXN diarias al cierre fue obtenido de *Investing.com*. El periodo de los datos es del 3 de enero de 2000 al 2 de mayo de 2019, siendo un total de 5091 registros correspondientes a los días hábiles bancarios incluyendo los sábados. La base no contó con datos faltantes. Sin embargo, existieron registros de algunos domingos, los cuales fueron duplicados del último día hábil. Se descartaron aquellos registros cuya diferencia respecto al día anterior fuese cero, dejando un total de 5066 registros (Figura 6.1).

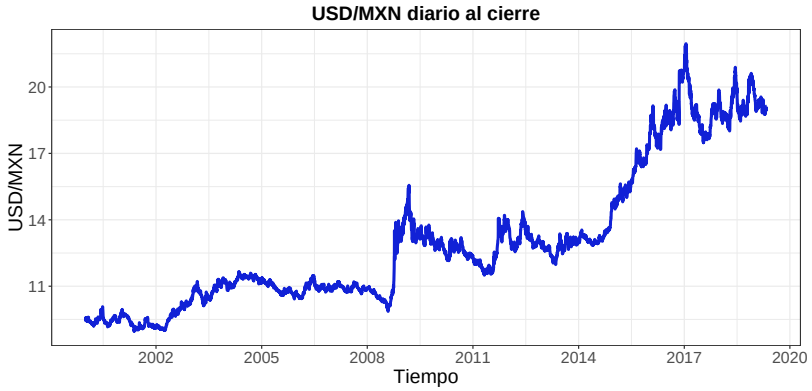


Figura 6.1: Tasa de cambio dólar-peso al cierre. Periodo: 3 de enero de 2000 al 2 de mayo de 2019.

Los rendimientos diarios se calcularon de acuerdo a la ecuación 6.1, es decir, se consideraron los rendimientos logarítmicos (Figura 6.2). Al hablar de los rendimientos, queda implícito que se alude a los rendimientos logarítmicos. El número de rendimientos que conformaron el conjunto de datos fue de 5065.

Se observó la presencia de datos atípicos. El *VaR* de los valores absolutos de los rendimientos al 99.75 % fue de 0.0381. El *VaR* de los valores absolutos de los rendimientos al 97.50 % fue de 0.0172. Se conservaron los rendimientos atípicos porque los valores atípicos son de gran importancia.

En la Tabla 6.1 se puede apreciar que la volatilidad histórica calculada como la desviación estándar de los rendimientos fue de 0.007136. La tendencia de los rendimientos fue, en promedio, del 0.0140 %. El máximo de los valores absolutos de los rendimientos fue de 0.0876. La distribución de los rendimientos fue leptocúrtica y asimétrica positiva. Más del 50 % de los rendimientos fueron negativos.

[1] A la fecha de la realización del presente trabajo, la url del repositorio es `https://github.com/Yizuzhub/USDMXNVolatilityPrediction`.

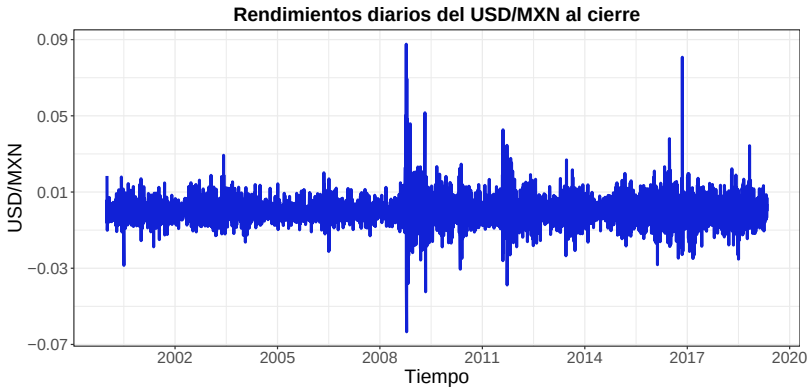


Figura 6.2: Rendimientos logarítmos de la tasa de cambio dólar-peso al cierre. Periodo: 3 de enero de 2000 al 2 de mayo de 2019.

Media	0.000140
Desviación estándar	0.007136
Coeficiente de asimetría	0.992558
Coeficiente de curtosis	17.476850
Mínimo	-0.063353
Máximo	0.087590
Mediana	-9.272997×10^{-5}

Tabla 6.1: Características de los rendimientos.

De la Figura 6.3 se aprecia que los rendimientos no siguen una distribución normal. De la prueba de Anderson-Darling se rechaza la hipótesis de normalidad, por lo que se concluye que los rendimientos no siguen una distribución normal. Se presentan colas pesadas y asimetría en la distribución de los rendimientos.

De forma aproximada, el 95 % de los rendimientos se encontraron en el intervalo $(-0.0127663, 0.01394141)$; el 99 %, en el intervalo $(-0.02157559, 0.02273389)$[2]. En la Tabla 6.2 se muestra que, de forma exacta, el 95 % de los rendimientos se encontraron entre -0.002766 y 0.013941, mientras que el 95 % de los rendimientos absolutos se encontraron entre 0.000163 y 0.017186. El VaR al 95 % de los rendimientos absolutos fue de 0.013418.

De la prueba de Dickey-Fuller aumentada[3] se obtuvo un *p-value* menor que 0.01, por lo que se rechaza la existencia de raíces unitarias, concluyendo que los rendimientos son estacionarios.

[2]Los intervalos de confianza al $100(1-\alpha)\,\%$ se constuyeron como se mencionó al principio del capítulo (expresión (6.2)).

[3]La prueba de Dickey-Fuller aumentada se utiliza para probar que los datos son estacionarios. Para mayor información consulte [33] y [24].

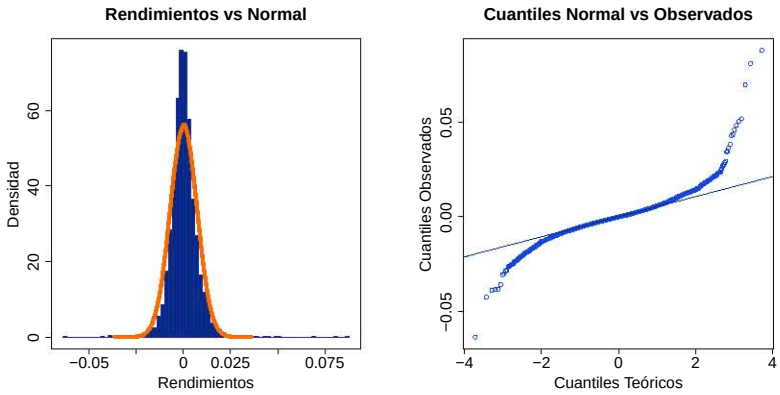


Figura 6.3: A la izquierda: histograma de los rendimientos y curva de la densidad de una distribución normal con media y desviación estándar correspondientes a las de los rendimientos. A la derecha: cuantiles de los rendimientos *vs* cuantiles de una distribución normal con media y desviación estándar correspondientes a las de los rendimientos.

$\alpha\%$	$q_{\alpha\%}(r_t)$	$q_{\alpha\%}(\lvert r_t\rvert)$
2.50	-1.2766305×10^{-2}	1.630007×10^{-4}
5.00	-9.882951×10^{-3}	2.818895×10^{-4}
95.00	1.1317616×10^{-2}	1.341822×10^{-2}
97.50	1.3941412×10^{-2}	1.718566×10^{-2}
99.50	2.2733893×10^{-2}	2.739927×10^{-2}
99.75	3.0991548×10^{-2}	3.813015×10^{-2}

Tabla 6.2: Cuantiles a diversos niveles de los rendimientos y rendimientos absolutos. $q_{\alpha\%}(r_t)$ y $q_{\alpha\%}(|r_t|)$ corresponden a los cuantiles al $\alpha\%$ de los rendimientos y del valor absoluto de los rendimientos, respectivamente.

Los correlogramas del ACF y el PACF (Figura 6.4) indicaron la existencia de cierta dependencia entre los rendimientos. Mediante la prueba de Ljung-Box[4] se rechazó la hipótesis de no correlación entre los rendimientos. Se puede inferir que, posiblemente, los rendimientos correspondan a un ARMA y los residuales a algún un proceso de heterocedasticidad condicional como un GARCH o un EGARCH. Para verificar que los rendimientos no corresponden a un proceso puramente heterocedástico (ya sea GARCH, EGARCH, etc.), los correlogramas del ACF y el PACF del valor absoluto de los rendimientos (Figura 6.5) y los del cuadrado de los rendimientos (Figura 6.6) mostraron que los rendimientos no corresponden a ruido blanco.

Para comprobar la efectividad de las predicciones, las volatilidades fueron estima-

[4]La prueba de Ljung-Box se utiliza para probar la no correlación entre los datos. Para mayor información consulte [13] y [29].

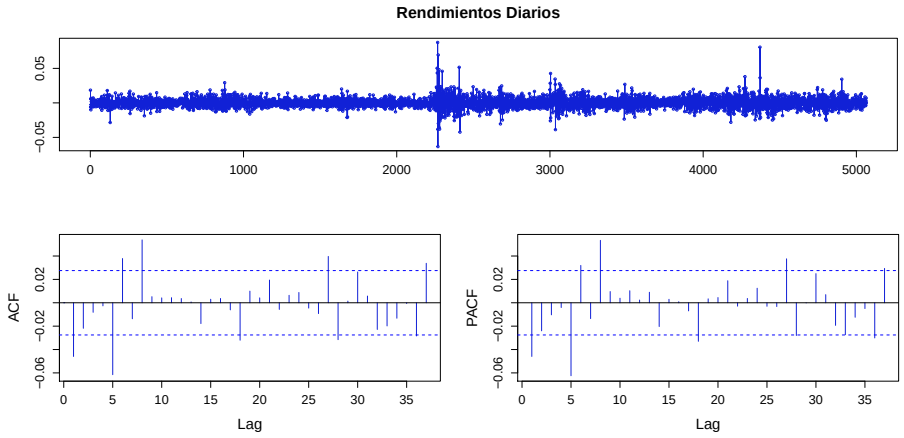


Figura 6.4: Rendimientos a un día de la tasa de cambio dólar-peso al cierre. Periodo: 3 de enero de 2000 al 2 de mayo de 2019. Inferior izquerda: ACF. Inferior derecha: PACF.

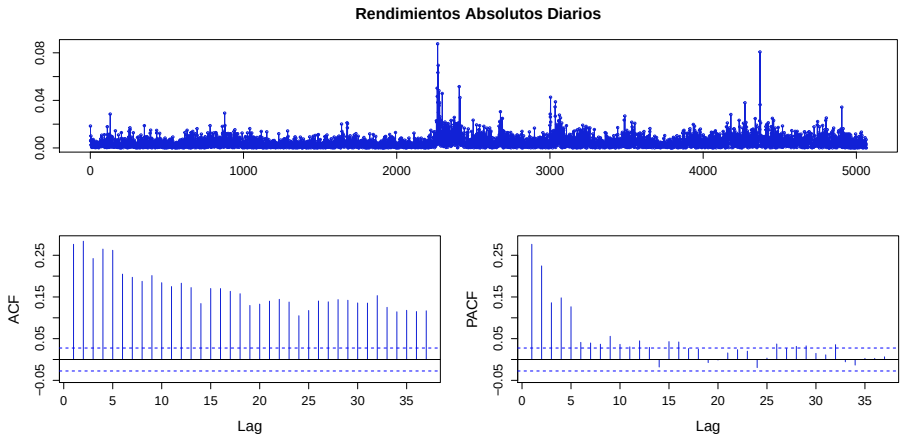


Figura 6.5: Valor absoluto de los rendimientos a un día de la tasa de cambio dólar-peso al cierre. Periodo: 3 de enero de 2000 al 2 de mayo de 2019. Inferior izquerda: ACF. Inferior derecha: PACF.

das considerando ventanas de 5, 10 y 22 días, periodos correspondientes a 1 semana, 2 semanas y un mes, respectivamente. Se compararon las predicciones de cada modelo con dichas ventanas, teniendo así una referencia de qué tan cercanos a la realidad fueron los modelos.

En la Figura 6.7 se puede apreciar que, a medida que aumenta la ventana de días, más estable es la estimación de la volatlidad. Los periodos de alta volatilidad del 2017

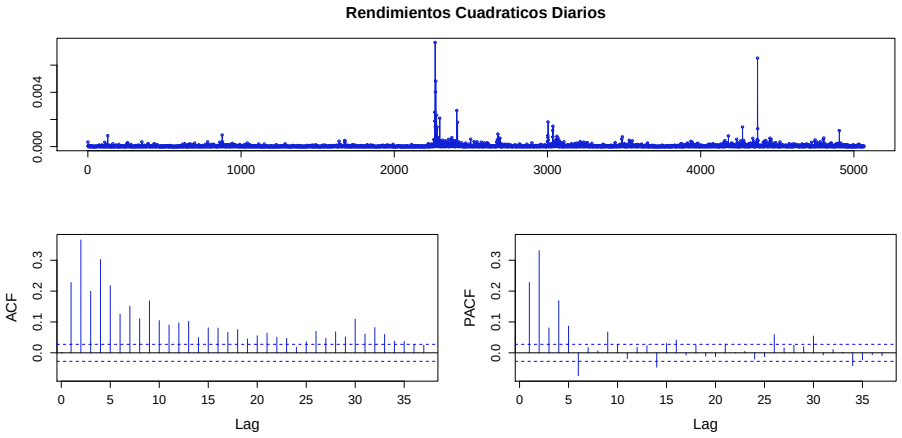


Figura 6.6: Rendimientos al cuadrado a un día de la tasa de cambio dólar-peso al cierre. Periodo: 3 de enero de 2000 al 2 de mayo de 2019. Inferior izquerda: ACF. Inferior derecha: PACF.

corresponden en parte a la toma de posesión de la presidencia de Estados Unidos por parte de Donald Trump, mientras que en el 2018 corresponden a las elecciones presidenciales en México (1 de julio) y fechas cercanas a la cancelación por parte de AMLO del NAIM (29 de octubre), así como otros acontecimientos cercanos a estas fechas. Uno de los periodos de menor volatilidad corresponde a la toma de posesión de la presidencia de la república mexicana por parte de Andrés Manuel López Obrador (12 de diciembre) y diversos eventos que acontecieron en fechas cercanas.

En la Tabla 6.3 se muestran algunas características de la volatilidad estimada bajo ventanas de 5 (Figura 6.8), 10 y 22 días. Las volatilidades siguen una distribución leptocúrtica y asimétrica positiva. Aproximadamente el 95 % de las volatilidades históricas calculadas bajo una ventana de 5 días se encontraron entre 0.001602354 y 0.01465665; el 99 %, entre 0.0009931323 y 0.02774118. Esto implica que 2.5 % de las volatilidades de 5 días se encuentran entre 0.0147 y 0.0656, lo que refleja la presencia de datos atípicos en las volatilidades de 5 días.

En la Tabla 6.4 se concluye que el 95 % de las volatilidades de 5, 10 y 22 días se encontraron entre 0.001602 y 0.014657, 0.002225 y 0.147195, y entre 0.002521 y 0.014725, respectivamente. El VaR al 95 % de las volatilidades de 5, 10 y 22 días fue de 0.012242, 0.011777 y 0.011660, en ese orden.

6.3. Ajuste de los Modelos

Con base en a los resultados obtenidos del análisis exploratorio, se concluyó que el proceso $\{r_t\}$ presenta heteroscedasticidad. Así, para los modelos individuales, bajo el

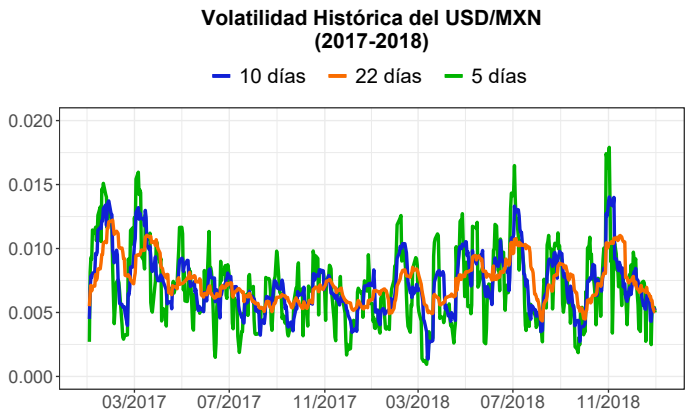


Figura 6.7: Volatilidad histórica de la tasa de cambio dólar-peso al cierre considerando ventanas de tiempo de 5, 10 y 22 días. Periodo: 01 de enero de 2017 al 01 de enero de 2019.

	$\hat{\sigma}_{5,t}$	$\hat{\sigma}_{10,t}$	$\hat{\sigma}_{22,t}$
Media	0.005902	0.006115	0.006266
Desviación estándar	0.004173	0.003789	0.003472
Coeficiente de asimetría	4.260108	4.048218	3.552347
Coeficiente de curtosis	39.29260	32.95068	24.65556
Mínimo	0.0004384	0.001106	0.001827
Máximo	0.0656062	0.048940	0.037812
Mediana	0.0050124	0.005300	0.005498

Tabla 6.3: Características de las volatilidades estimadas con ventanas de 5, 10 y 22 días. $\hat{\sigma}_{5,t}$, $\hat{\sigma}_{10,t}$ y $\hat{\sigma}_{22,t}$ representan las volatilidades estimadas bajo una ventana de 5, 10 y 22 días, respectivamente.

$\alpha\%$	$q_{\alpha\%}(\hat{\sigma}_{5,t})$	$q_{\alpha\%}(\hat{\sigma}_{10,t})$	$q_{\alpha\%}(\hat{\sigma}_{22,t})$
2.50	1.602354×10^{-3}	2.225202×10^{-3}	2.521447×10^{-3}
5.00	1.896656×10^{-3}	2.507223×10^{-3}	2.899396×10^{-3}
95.00	1.224154×10^{-2}	1.177695×10^{-2}	1.166005×10^{-2}
97.50	1.465665×10^{-2}	1.471952×10^{-2}	1.472534×10^{-2}
99.50	2.774118×10^{-2}	2.680349×10^{-2}	2.304035×10^{-2}
99.75	3.568205×10^{-2}	3.173275×10^{-2}	3.377680×10^{-2}

Tabla 6.4: Cuantiles a diversos niveles de las volatilidades estimadas bajo las ventanas de 5, 10 y 22 días. $q_{\alpha\%}(\hat{\sigma}_{d,t})$ es el cuantil al $\alpha\%$ de las volatilidades de d días al tiempo t.

Figura 6.8: Volatilidad histórica de la tasa de cambio dólar-peso al cierre considerando una ventana de tiempo de 5 días. Periodo: 10 de enero de 2000 al 03 de mayo de 2019.

enfoque frecuentista, se elegieron los procesos ARMA-GARCH y ARMA-EGARCH, siendo $\{r_t\}$ un proceso ARMA y los residuales del proceso ARMA corresponden a un proceso heterocedástico, ya sea GARCH o EGARCH, donde el proceso de volatilidad para el proceso heterocedástico es $\{\sigma_{d,t}\}$. Bajo el enfoque de *machine learning*, se eligieron los modelos de redes neuronales LSTM y GRU. Los modelos ajustados bajo el enfoque frecuentista fueron, a su vez, utilizados en los modelos híbridos. Al final, se realizó una comparación entre los modelos seleccionados de cada uno de los tipos.

6.3.1. Partición de la Muestra

Para el entrenamiento de las redes neuronales se eligió el optimizador *Adagrad* por lo que, con el objetivo de tener comodidad al momento de programar, el número de ejemplos de cada una de las particiones del conjunto de datos debe ser múltiplo del tamaño de los minilotes considerados. Se eligió un tamaño de 84 para los minilotes.

Considerar volatilidades de los últimos días para predecir la del día siguiente permite que la red aprenda mejor, pero la cantidad de días debe ser razonable, ya que si el periodo es muy extenso, se podrían tener dificultades al entrenar la red. Por ello, se consideró una dependencia temporal de los últimos 3 días, es decir, *time steps* de 3 días. La elección tanto del tamaño de los minilotes como de los *time steps* es libre.

Como el objetivo es comprobar si los modelos híbridos pueden mejorar la eficiencia al realizar la tarea **T** de los modelos ajustados bajo un único enfoque, es conveniente contar con una cantidad de datos que pueda ser suficiente para que el ajuste de las redes neuronales permita ajustar los parámetros de los modelos de redes neuronales de tal

forma que su eficiencia sea aceptable, pero no tan grande ya que una de las cuestiones que se desea comprobar es si los modelos híbridos pueden mejorar las predicciones aún cuando los datos sean escasos. Debido a que a partir del 2008 la volatilidad fue más inestable (Figura 6.8), la partición de la muestra se conformó de la siguiente manera: 40 % para el conjunto de entrenamiento, 30 % para el conjunto de validación y el 30 % restante para el conjunto de prueba.

Para el ajuste de los modelos de series de tiempo se consideró como conjunto de entrenamiento el mismo de la red neuronal más los primeros rendimientos con los que se estimó la primera volatilidad histórica para cada una de las ventanas. Así, los periodos que conformaron cada una de las particiones del conjunto de datos se muestran en la Tabla 6.5. Para la ventana de 5 días, el conjunto de entrenamiento estuvo integrado por 2016 ejemplos; y los conjuntos de validación y de prueba, por 1512 cada uno.

Ventana (días)	Conjunto	Periodo		
5	Entrenamiento	$2000-01-03$	al	$2007-10-29$
	Validación	$2007-10-30$	al	$2013-08-20$
	Prueba	$2013-08-21$	al	$2019-04-10$
10	Entrenamiento	$2000-01-03$	al	$2007-11-05$
	Validación	$2007-11-06$	al	$2013-08-27$
	Prueba	$2013-08-28$	al	$2019-04-16$
22	Entrenamiento	$2000-01-03$	al	$2007-11-21$
	Validación	$2007-11-22$	al	$2013-09-12$
	Prueba	$2013-09-13$	al	$2019-04-30$

Tabla 6.5: Particiones del conjunto de datos. El formato de las fechas es aaaa-mm-dd.

6.3.2. Ajuste de Modelos Bajo un Enfoque Clásico

Como primer acercamiento, se ajustó un modelo ARMA con el fin de verificar si los rendimientos podrían tratarse de un proceso ARMA y los respectivos residuales corresponder a un proceso heterocedástico. Posteriormente, se ajustaron los modelos ARMA heterocedásticos donde los órdenes del proceso ARMA se obtuvieron a partir del primer acercamiento.

Ajustes Preliminares

De forma inicial, se ajustó un modelo ARMA$(2,3)$ para los rendimientos, obtenido como sugerencia mediante la función `auto.arima` del paquete `forecast`, ya que ésta sugiere un modelo ARIMA con base en los menores puntajes del *AIC*, *BIC* o *DIC*. En el ajuste del modelo se utilizó el conjunto de datos en su totalidad. El modelo ajustado se muestra en la Tabla 6.6. Los parámetros, con excepción de la tendencia, fueron significativos. La tendencia en principio debería ser cero ya que, de lo contrario,

implicaría una especie de arbitraje. Sin embargo, para poder analizar mejor los datos se optó por no eliminarla. En los modelos ajustados de la presente sección se empleó el conjunto total de datos, ya que se buscan modelos de series de tiempo que describan las volatilidades estimadas, en particular, las volatilidades estimadas de 5 días.

Coeficiente	ar 1	ar 2	ma 1	ma 2	ma 3	*Drift*
Valor	-1.4818	-0.6312	1.4403	0.5389	-0.0753	0.0001
Desv. Estándar	0.0998	0.0935	0.1000	0.0962	0.0149	0.0001

Tabla 6.6: Modelo obtenido mediante la función `auto.arima`.

Los correlogramas del ACF y el PACF sugieren que los residuales del modelo ARMA$(2,3)$ se tratan de ruido blanco (Figura 6.9). Sin embargo, los correlogramas del ACF y el PACF del valor absoluto y de los residuales cuadráticos (Figura 6.9) dan indicios de que los residuales no corresponden a ruido blanco.

Mediante la prueba de Ljung-Box, se concluye que no existen pruebas suficientes para rechazar la hipótesis nula de independencia entre los residuales, de modo que los residuales no están correlacionados.

Con la prueba de Anderson-Darling para normalidad se rechaza la hipótesis nula de normalidad, por lo que los residuales no siguen una distribución normal.

Mediante la prueba ARCH de Engle se rechaza la hipótesis nula de que los residuales cuadráticos no son autorregresivos, por lo que se confirma la existencia de efectos ARCH.

Con base en los resultados obtenidos, se decidió asumir que la tasa de cambio sigue una dinámica ARMA y los residuales corresponden a algún proceso heterocedástico y sin considerar una tendencia en los rendimientos. Se propone utilizar la distribución del error generalizada sesgada (SGED)[5] debido a que se puede notar claramente la presencia de asimetría y de colas pesadas.

Debido a que la tendencia no era significativa, se ajustaron modelos cuyos coeficientes fueran significativos, obteniendo los modelos AR(1), MA(1) y ARMA$(1,1)$, cuyos coeficientes se muestran en la Tabla 6.7.

Para cada uno de los modelos ajustados AR(1), MA(1) y ARMA$(1,1)$, mediante la prueba de Ljung-Box y con un *lag* máximo de 4 se concluye que no existen pruebas suficientes para rechazar la hipótesis nula de independencia entre los residuales,

[5] Sea X una variable aleatoria que sigue una distribución *SGED*. La función de densidad de X está dada por:

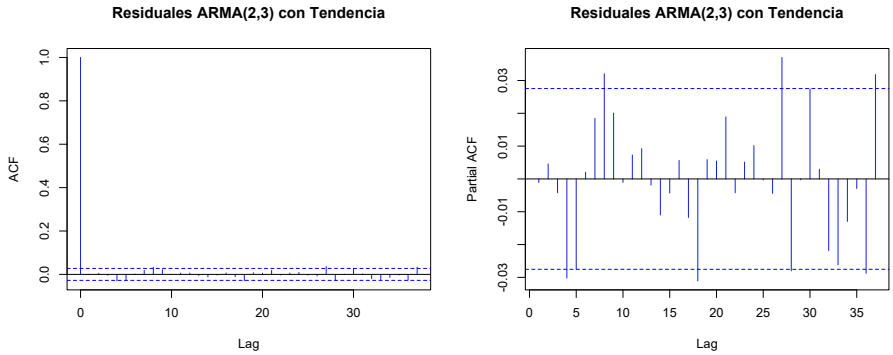
$$f(x) = \frac{k^{1-\frac{1}{k}}}{2\phi}\Gamma\left(\frac{1}{k}\right)^{-1} exp\left\{-\frac{|u|^k}{(1+sign(u)\lambda)^k\phi^k k}\right\},$$

donde $u = x - m$, m es la moda de X, ϕ es un parámetro de escala, λ es un parámetro de localización, k es un parámetro de kurtosis, *sign* es la función signo que toma el valor de -1 si $u < 0$ y 1 si $u > 0$, y $\Gamma(a)$ es la función gamma:

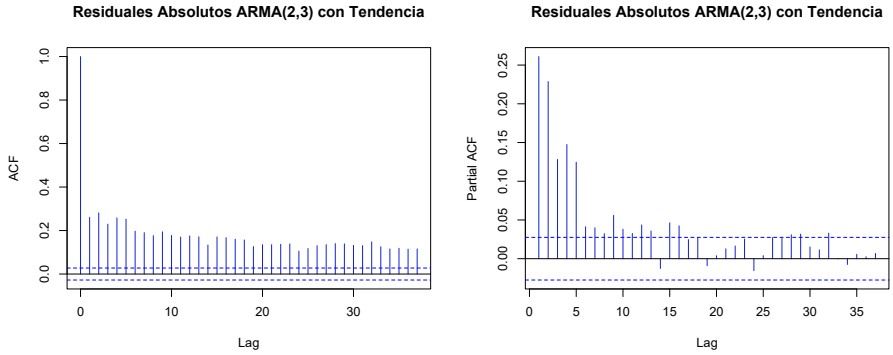
$$\Gamma(a) = \int_0^\infty t^{a-1}e^{-t}dt.$$

Para mayor información consulte [32].

a) Residuales del proceso ARMA(2,3) con tendencia.



a) Residuales absolutos del proceso ARMA(2,3) con tendencia.



b) Residuales cuadráticos del proceso ARMA(2,3) con tendencia.

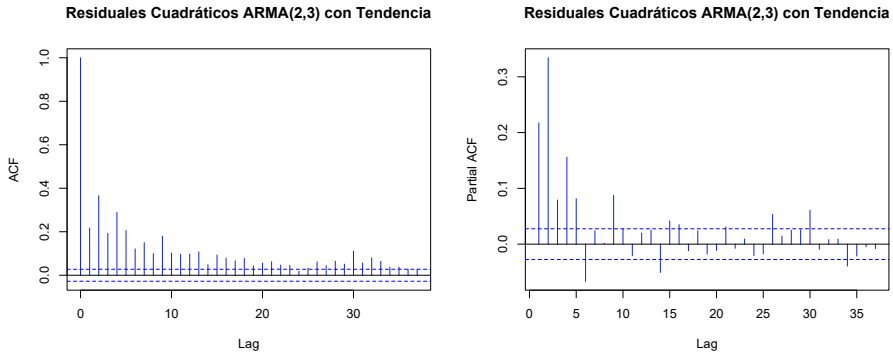


Figura 6.9: Residuales absolutos y cuadráticos de los rendimientos de la paridad USD/MXN bajo el modelo ARMA$(2,3)$ propuesto.

	$AR(1)$	$MA(1)$	$ARMA(1,1)$	
Coeficiente	ar	ma	ar	ma
Valor	-0.0456	-0.0478	0.4698	-0.5182
Desv. Estándar	0.0140	0.0144	0.1815	0.1770

Tabla 6.7: Modelos iniciales.

de modo que los residuales no están correlacionados; para un *lag* superior a 4, se rechaza la no correlación entre residuales. Por ende, se concluye que los residuales están correlacionados.

Estos modelos fueron ajustados bajo el supuesto de que la varianza de los residuales es constante pero, como se vió en la sección 6.2 y en los ajustes preliminares, se presentaron efectos ARCH en los rendimientos, por lo que se prosiguió a ajustar los modelos ARMA para los rendimientos y los modelos heterocedásticos para los residuales.

Ajuste de los Modelos Heterocedásticos

Con base en los resultados obtenidos en los ajustes preliminares, el orden máximo considerado para el proceso ARMA, tanto para la parte autorregresiva como la de promedio móvil, fue 1 y para el proceso heterocedástico fue 2, además de las combinaciones posibles entre un $ARMA(2,3)$ y un proceso heterocedástico de hasta orden 2. Los procesos heterocedásticos considerados fueron GARCH y EGARCH. A partir de esta sección, se respetó la partición de la muestra como en la Tabla 6.5.

La tendencia no se agregó en el proceso ARMA porque eso implicaría la existencia de arbitraje, obligando al mercado a disminuir esa tendencia a cero. En el caso del proceso GARCH, no se quitó la tendencia para asegurar que la volatilidad no se anule. Esta tendencia se puede interpretar como el nivel hacia el cual tiende la volatilidad al estabilizarse.

Se propusieron modelos candidatos entre el total de modelos ajustados. Los modelos candidatos fueron seleccionados por ser los que se consideraron más eficientes de los modelos ajustados. Para la selección de modelos candidatos, bajo el enfoque clásico, se tomaron en consideración varios criterios: aquellos que tuvieran los menores criterios de información AIC, BIC, y los errores de ajuste con las métricas MAE, RMSE y MAPE, teniendo mayor preferencia el BIC sobre el AIC ya que el primero penaliza de acuerdo al número de parámetros estimados y al tamaño de muestra (Tabla 6.8).

Los errores porcentuales de predicción se midieron con la métrica MAPE por lo que un MAPE de 0.10 indicaría un error porcentual relativo del $10\,\%$. También se tomó en consideración, pero no necesariamente, aquellos cuyos coeficientes fuesen estadísticamente significativos al nivel de confianza del $95\,\%$.

Del total de modelos ajustados, se seleccionaron 3 modelos ARMA-GARCH y ARMA-EGARCH. Los coeficientes de los modelos ARMA-EGARCH de orden $(0,1)-$

$(1,1)$ y $(1,0)-(1,1)$ fueron significativos.

Modelo	Orden	AIC	BIC	MAE	MAPE (%)
ARMA-GARCH	(0,1)-(1,1)	$-7.91448^{\star}$	$-7.89782^{\star}$	0.00132403	42.3004
	(1,0)-(1,1)	-7.91429	-7.89764	0.00132286	42.2620
	(1,1)-(1,1)	-7.91362	-7.89419	$0.00131751^{\star}$	$42.1964^{\star}$
ARMA-EGARCH	(0,1)-(1,1)	-7.91997	$-7.90054^{\star}$	0.0013232	41.5454
	(1,0)-(1,1)	-7.91991	-7.90048	$0.0013219^{\star}$	$41.5030^{\star}$
	(0,1)-(1,2)	$-7.92531^{\star}$	-7.90033	0.0013587	42.0787

Tabla 6.8: Modelos de series de tiempo candidatos. Errores de predicción calculados con el conjunto de entrenamiento y la volatilidad de 5 días. Los puntajes con el símbolo * fueron los mejores de acuerdo a cada tipo de proceso heterocedástico.

Modelo	Orden	MAE	RMSE	MAPE (%)
ARMA-GARCH	(0,1)-(1,1)	0.00345497	0.00617685	$46.9491^{\star}$
	(1,0)-(1,1)	$0.00345427^{\star}$	$0.00617584^{\star}$	46.9591
	(1,1)-(1,1)	0.00345823	0.00617880	47.0651
ARMA-EGARCH	(0,1)-(1,1)	$0.00353454^{\star}$	$0.00628460^{\star}$	45.9182
	(1,0)-(1,1)	0.00353491	0.00628492	45.9242
	(0,1)-(1,2)	0.00353547	0.00629916	$45.1791^{\star}$

Tabla 6.9: Modelos de series de tiempo candidatos. Errores de predicción calculados con el conjunto de validación y la volatilidad de 5 días. Los puntajes con el símbolo * fueron los mejores de acuerdo a cada tipo de proceso heterocedástico.

Antes de continuar con el análisis, se harán unas observaciones:

Observación 1 *Sean X,Y conjuntos. Sean $\hat{f}_1$ y $\hat{f}_2$ dos modelos ajustados (1 y 2, respectivamente) que aproximan una función $f : X \to Y$. Sean $\hat{y}_i^{(1)}$ y $\hat{y}_i^{(2)}$ las variables predichas $\hat{f}_1$ y $\hat{f}_2$. Considérese una muestra de tamaño m del conjunto X. Supóngase que $MAPE_1 < MAPE_2$, donde $MAPE_i$ corresponde al MAPE del modelo i. Entonces,*

$$\frac{1}{m}\sum_{i=1}^{m}\left|1-\frac{\hat{y}_i^{(1)}}{y_i}\right| < \frac{1}{m}\sum_{i=1}^{m}\left|1-\frac{\hat{y}_i^{(2)}}{y_i}\right|.$$

Caso 1: $y_i \geq \hat{y}_i^{(j)} \geq 0, y_i > 0, j = 1,2.$

$$\begin{aligned}\sum_{i=1}^{m}\frac{\hat{y}_i^{(1)}}{y_i} &> \sum_{i=1}^{m}\frac{\hat{y}_i^{(2)}}{y_i}\\ \sum_{i=1}^{m}\frac{\hat{y}_i^{(1)}-\hat{y}_i^{(2)}}{y_i} &> 0,\end{aligned}$$

lo cual se asegura si $\hat{y}_i^{(1)} \geq \hat{y}_i^{(2)}$.

Caso 2: $0 < y_i < \hat{y}_i^{(j)}, j = 1,2.$

De forma análoga al caso 1 se concluye que $\hat{y}_i^{(1)} < \hat{y}_i^{(2)}$.

De los casos anteriores se concluye que:

Si $y_i \geq \hat{y}_i^{(j)} > 0$*, entonces* $\hat{y}_i^{(1)} > \hat{y}_i^{(2)}$.

Si $0 < y_i < \hat{y}_i^{(j)}$*, entonces* $\hat{y}_i^{(1)} < \hat{y}_i^{(2)}$.

De la observación 1 se puede apreciar que, dados dos modelos ajustados $\hat{f}_1$ y $\hat{f}_2$, si $MAPE_1 < MAPE_2$ y además $MAE_1 = MAE_2$, para los dos casos expuestos en la observación, entonces en momentos de alta volatilidad $\hat{f}_2$ subestimas más la volatilidad que $\hat{f}_1$ o, en periodos de baja volatilidad, $\hat{f}_2$ sobreestima más la volatilidad que $\hat{f}_1$. La condición $MAE_1 = MAE_2$ se agregó ya que ello permite conocer de forma adicional si los dos modelos ajustados realizan las predicciones de forma aparentemente igual.

Así, de acuerdo a la Tabla 6.9, los modelos MA-EGARCH de órdenes (1)-(1,1) y (1)-(1,2) tienen un MAE muy similar, ya que el MAE del modelo MA(1)-EGARCH(1,2) es mayor al del MA(1)-EGARCH(1,1) por menos del 0.0263 %. Así, por la observación 1 se puede deducir que, en promedio, el modelo MA(1)-EGARCH(1,1) subestima más la volatilidad en periodos de alta incertidumbre que el modelo MA(1)-EGARCH(1,2) o la sobreestima más en tiempos de baja incertidumbre que el modelo MA(1)-EGARCH(1,2).

La elección de un modelo para cada tipo de modelo heterocedástico se realizó con base en los resultados obtenidos y los errores de predicción utilizando el conjunto de validación (Tabla 6.9). Los modelos GARCH y EGARCH seleccionados fueron:

⋆ Proceso $\mathrm{MA}(1)-\mathrm{GARCH}(1,1)$:

$$\begin{aligned} y_t &= \varepsilon_t - 0.0884\varepsilon_{t-1} \\ \sigma_t^2 &= 6.4195\text{x}10^{-7} + 0.0602\varepsilon_{t-1}^2 + 0.9121\sigma_{t-1}^2 \end{aligned}$$

con parámetros de asimetría *skew*= 1.1221 y de forma *shape*= 1.4037 para la distribución del error.

⋆ Proceso $\mathrm{MA}(1)-\mathrm{EGARCH}(1,1)$:

$$\begin{aligned} y_t &= \varepsilon_t - 0.0785\varepsilon_{t-1} \\ \log\sigma_t^2 &= -0.3482 + 0.0535 g_1(\eta_{t-1}) + 0.9677\log\sigma_{t-1}^2 \\ g(\eta_{t-1}) &= \eta_{t-1} + 0.1179(|\eta_{t-1}| - \mathbb{E}[|\eta_{t-1}|]) \end{aligned}$$

con parámetros de asimetría *skew*= 1.1157 y de forma *shape*= 1.4108 para la distribución del error.

En la Tabla 6.10 se muestra que los coeficientes de cada modelo son significativos con excepción de dos del modelo MA-GARCH. El nivel de confianza considerado es del 0.05.

Modelo	Coeficiente	Estimado	Desv. estándar	*p-value*
MA(1)− GARCH(1,1)	ma1	-8.839112×10^{-2}	2.265748×10^{-2}	0.0001
	ω	6.419495×10^{-7}	6.211317×10^{-7}	0.3014
	α_1	6.024317×10^{-2}	1.326640×10^{-3}	0.0000
	β_1	9.120909×10^{-1}	1.584011×10^{-2}	0.0000
	sesgo	1.122077×10^{0}	2.829553×10^{-2}	0.0000
	forma	1.403666×10^{0}	4.790416×10^{-2}	0.0000
MA(1)− EGARCH(1,1)	ma1	-7.8508×10^{-2}	2.053966×10^{-2}	0.0001
	ω	-3.48239×10^{-1}	7.526730×10^{-3}	0.0000
	α_1	5.3490×10^{-2}	1.356224×10^{-2}	0.0001
	β_1	9.67710×10^{-1}	7.447588×10^{-4}	0.0000
	γ_1	1.17918×10^{-1}	1.224662×10^{-2}	0.0000
	sesgo	1.115688×10^{0}	2.954330×10^{-2}	0.0000
	forma	1.410827×10^{0}	6.154464×10^{-2}	0.0000

Tabla 6.10: Coeficientes de los modelos seleccionados, MA(1)−GARCH(1,1) y MA(1)−EGARCH(1,1).

Para verificar su eficiencia al momento de predecir la volatilidad estimada, se realizaron predicciones con el conjunto de prueba. Los errores de predicción con el conjunto de prueba se muestran en la Tabla 6.11.

En el caso de la volatilidad de 5 días, el MAE del MA(1)-GARCH(1,1) fue menor al del modelo MA(1)-EGARCH(1,1) pero el MAPE del MA(1)-GARCH(1,1) fue mayor, lo cual sugiere que el MA(1)-GARCH(1,1) fue menos eficiente que el MA-EGARCH para pronosticar bajas volatilidades, o fue más eficiente para pronosticar altas volatilidades.

El modelo MA(1)-GARCH(1,1) fue más eficiente que el MA-EGARCH bajo todas las ventanas de tiempo y las métricas MAE, RMSE y MAPE, con excepción del MAPE de la ventana de 5 días, en donde ocurrió lo contrario.

Modelo	Ventana (días)	MAE	RMSE	MAPE (%)
MA(1)− GARCH(1,1)	5	0.00272721*	0.00413057*	43.8616
	10	0.00250435*	0.00374312*	33.2177*
	22	0.00243653*	0.00345121*	30.6031*
MA(1)− EGARCH(1,1)	5	0.00282370	0.00424953	43.3410*
	10	0.00264514	0.00389107	34.1908
	22	0.00263187	0.00362985	32.8126

Tabla 6.11: Comparación de los errores de predicción entre los distintos modelos de series de tiempo heterocedásticos con el conjunto de prueba mediante diversas métricas y bajo las diferentes ventanas de tiempo.

Uno de los inconvenientes de calcular los errores de predicción mostrados en la

Tabla 6.11 es que los errores de predicción se acumulan, además de que la volatilidad tiende a ser estable. Con la finalidad de comparar la eficiencia entre los distintos modelos de series de tiempo y los de *machine learning* se realizaron predicciones a un día. Con predicción a un día se refiere a que las volatilidades se predicen, dada la información hasta t, a un día después, es decir, $\hat{\sigma}_{t+1}$ se predice con la información hasta tiempo t. Las volatilidades pronosticadas $\hat{\sigma}_{t+1}$ no forman parte de la σ-álgebra $\mathscr{F}_{t+1}$. En otras palabras, las predicciones se realizan con los datos observados. Los errores de predicción a un día con el conjunto de prueba se encuentran en la Tabla 6.12.

En la Figura 6.10 se muestran las predicciones con el conjunto de prueba mediante ambos modelos seleccionados. Bajo la ventana de 5 días, el modelo MA(1)-EGARCH(1,1) predijo mejor las bajas volatilidades, mientras que en el resto de los casos, el modelo MA(1)-GARCH(1,1) fue más eficiente. Por otro lado, bajo la ventana de 22 días, el modelo MA(1)-GARCH(1,1) fue más eficiente que el MA(1)-EGARCH(1,1), además de que éste último sobreestima más las altas volatilidades que el modelo MA(1)-GARCH(1,1).

Las predicciones a un día mejoraron para ambos modelos con excepción del MAPE de la ventana de 5 días. Además, los errores de predicción bajo las métricas MAE, RMSE y MAPE del MA(1)-EGARCH(1,1) fueron menores a los del MA(1)-GARCH(1,1), lo cual podría deberse a las asimetrías existentes en el modelo EGARCH, ya que el mercado no se mueve con la misma intensidad y magnitud ante momentos de relativa calma que en momentos donde la incertidmubre es muy alta o muy baja. Además, se puede deducir que, a corto plazo, el modelo MA(1)-EGARCH(1,1) es más eficiente que el modelo MA(1)-GARCH(1,1) (Tabla 6.12), mientras que a largo plazo, sucede lo contrario, es decir, el modelo MA(1)-GARCH(1,1) es más eficiente (Tabla 6.11).

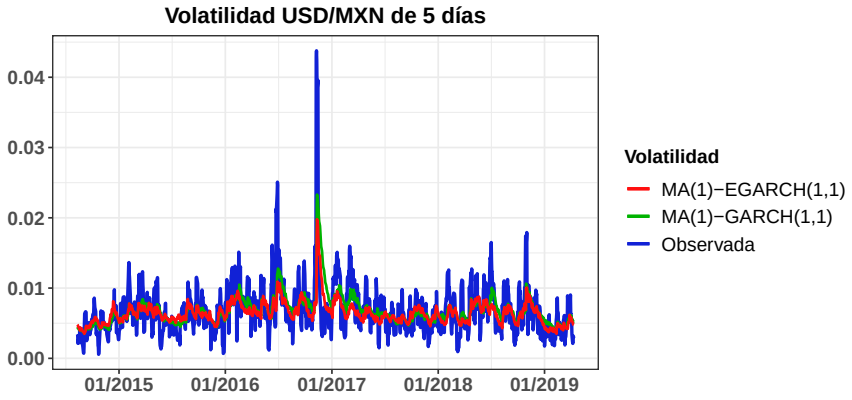
Modelo	Ventana (días)	MAE	RMSE	MAPE (%)
MA(1)− GARCH(1,1)	5	0.00192988*	0.00280228*	38.4362
	10	0.00122053*	0.00180848*	18.6487*
	22	0.00082251*	0.00123695*	11.0704*
MA(1)− EGARCH(1,1)	5	0.00193410	0.00290142	37.0404*
	10	0.00134949	0.00210754	19.4122
	22	0.00115935	0.00181964	14.8799

Tabla 6.12: Comparación de los errores de predicción a un día de los distintos modelos de series de tiempo heterocedásticos con el conjunto de prueba mediante diversas métricas y bajo las diferentes ventanas de tiempo.

6.3.3. Modelos de Redes Neuronales

Para el entrenamiento de las redes se empleó una tasa de aprendizaje de 0.01. La función de costos empleada, es decir, la medida **P** que se utilizó para saber si la red está aprendiendo o no, fue el MSE; el algoritmo de aprendizaje, *Adagrad* con $\varepsilon = 0$, 200 épocas y *Early Stopping* de 10 épocas.

a) Volatilidad de 5 días.



b) Volatilidad de 22 días.

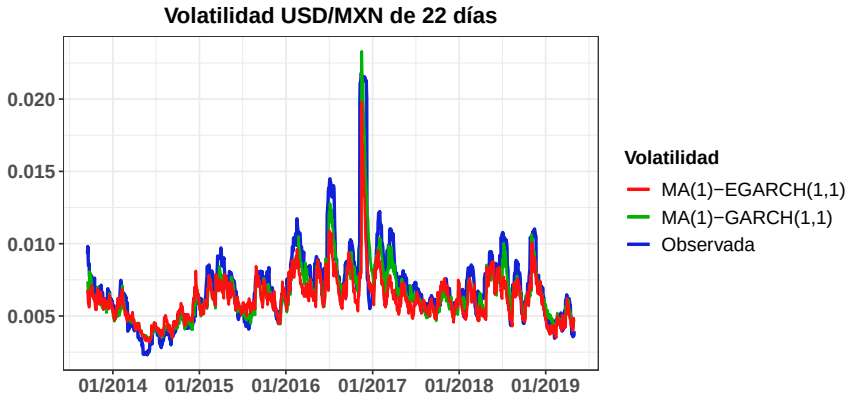


Figura 6.10: Predicción de la volatilidad de 5 (a) y 22 (b) días mediante los modelos MA(1)-GARCH(1,1) y MA(1)-EGARCH(1,1) con el conjunto de prueba.

Debido a que se requirió utilizar información de varios días para predecir un valor, se puede emplear una arquitectura muchos a uno. Otra alternativa es emplear la arquitectura uno a uno, ingresando las volatilidades de los días inmediatos anteriores de acuerdo al número de *time steps* seleccionado, en este caso, 3, a la red como entrada única, por lo que cada *input* es un vector de dimensión 3.

La arquitectura elegida consistió, tanto para la red LSTM como la GRU, en seis unidades en la capa oculta, siendo 2 unidades por cada *time step*. La capa de salida consistió en una neurona cuya función de activación fue la función identidad $f(x) = x$, ya que se requiere que la salida de la red sea un número, el cual corresponde a la volatilidad predicha del tiempo $t+1$. Con la finalidad de evitar *overfitting*, se utilizó un *dropout* de 0.3.

Para definir el conjunto de datos $\mathbf{E}$, se crearon subconjuntos de k valores correspondientes a los h tiempos previos (*time steps*) para cada tiempo. Para el tiempo t, los elementos del conjunto X (*inputs*) estuvieron conformados por los vectores de la forma $(\hat{\sigma}_{5,t-h+1}, \hat{\sigma}_{5,t-h+2}, ..., \hat{\sigma}_{5,t})$, mientras que los elementos del conjunto Y (*outputs*) correspondieron a $\hat{\sigma}_{5,t+1}$. En este caso, $h = 3$.

Los tiempos de entrenamiento pueden servir para tener una idea del tiempo que llevaría entrenar cada uno de los modelos bajo las mismas condiciones si los tiempos de entrenamiento son extensos, por ejemplo, 1 hora, mas no un indicador absoluto para realizar comparaciones acerca del modelo que sea más rápido de entrenar. Por esta razón, no se tomaron en cuenta. En su lugar, se debe considerar el número de parámetros a estimar, la rapidez de convergencia y la complejidad del modelo, ya que los tiempos de entrenamiento no siempre son los mismos debido a que depende de varios factores relacionados con el uso de la *CPU*, como la ejecución de procesos en primer y segundo plano.

Una de las recomendaciones es que los datos se encuentren reescalados entre -1 y 1. Las volatilidades estimadas son positivas e inferiores a 0.07. Sin embargo, es fácil notar que el entrenamiento de las redes podría mejorar si los datos se encuentran entre -1 y 1 (las redes están conformadas por las funciones sigmoide y tanh). Por ende, se decidió escalar los datos para cada uno de los modelos. Los errores de predicción considerarán los datos reescalados. Las volatilidades estimadas se multiplicaron por $\left\lfloor \text{máx}\{\hat{\sigma}_{5,t}^{-1}\} \right\rfloor = 15$.

En la Tabla 6.13 se muestra una comparación entre los modelos LSTM y GRU. Los errores de predicción, tanto con el conjunto de entrenamiento como con el conjunto de validación, bajo las métricas MAE, RMSE y MAPE, fueron inferiores en la red GRU a los de la red LSTM.

Los errores de predicción con el conjunto de validación fueron superiores a los errores de predicción con el conjunto de entrenamiento, con un incremento del 52 % y del 37 % en el RMSE de las redes LSTM y GRU, respectivamente, lo cual se debió en parte a que la red fue entrenada con el 40 % de los datos, mientras que el conjunto de validación correspondió al 30 % de los datos.

Los errores de predicción de la red GRU tanto con el conjunto de entrenamiento

	LSTM	GRU
No. Parámetros	511	385
Entrenamiento		
MAE	0.00100433	0.00096102*
RMSE	0.00138220	0.00133549*
MAPE	0.28520156	0.27227582*
Validación		
MAE	0.00138393	0.00113435*
RMSE	0.00210093	0.00182446*
MAPE	0.19810129	0.11773579*

Tabla 6.13: Comparación de los modelos ajustados de redes neuronales recurrentes LSTM y GRU, y sus respectivos errores de predicción con los conjuntos de entrenamiento y de validación mediante diversas métricas.

Ventana (días)	Modelo	MAE	RMSE	MAPE
5	LSTM	0.00123212	0.00174199*	0.18914312
	GRU	0.00117131*	0.00176236	0.18472534*
10	LSTM	0.00107248	0.00142675	0.14137734
	GRU	0.00092980*	0.00134893*	0.12158502*
22	LSTM	0.00106698	0.00134379	0.13634354
	GRU	0.00087706*	0.00119204*	0.10927173*

Tabla 6.14: Comparación de los errores de predicción de los modelos de redes neuronales recurrentes propuestos con el conjunto de prueba mediante diversas métricas.

como con el conjunto de validación fueron inferiores a los de la red LSTM, lo cual en parte puede deberse a que el tamaño de muestra para el conjunto de entranmiento puede no ser suficiente para aprovechar la mayor complejidad de la red LSTM sobre la red GRU, tanto por el diseño como por el número de parámetros a ajustar (una red LSTM permite tener un mejor control sobre los datos que una red GRU, mas no por eso el rendimiento será siempre superior). La cantidad de parámetros a ajustar de la red GRU fue inferior a la de la red LSTM (126 parámetros menos), lo cual a su vez afecta el proceso de entrenamiento de ambas las redes.

Debido a que el objetivo principal del presente trabajo no es ajustar un modelo para predecir la volatilidad sino comprobar si los modelos híbridos pueden mejorar la eficiencia de los modelos que los conforman, cuyos parámetros fueron ajustados bajo enfoques estadísticos distintos, no se optimizó el proceso de aprendizaje.

En la Tabla 6.14 se pueden apreciar los errores de predicción con el conjunto de prueba. Para las ventanas de 10 y 22 días, la red GRU tuvo menores errores de predicción que la red LSTM bajo las métricas MAE, RMSE y MAPE. Para la ventana de 5 días, la red GRU fue más eficiente que la red LSTM bajo las métricas MAE y MAPE, aunque fue menos eficiente según la métrica RMSE. En la Figura 6.11 se muestran

las predicciones realizadas con el conjunto de prueba mediante los modelos LSTM y GRU, respectivamente. De forma gráfica se puede apreciar que la red GRU es menos eficiente, en promedio, para momentos de donde la volatilidad es alta o es baja.

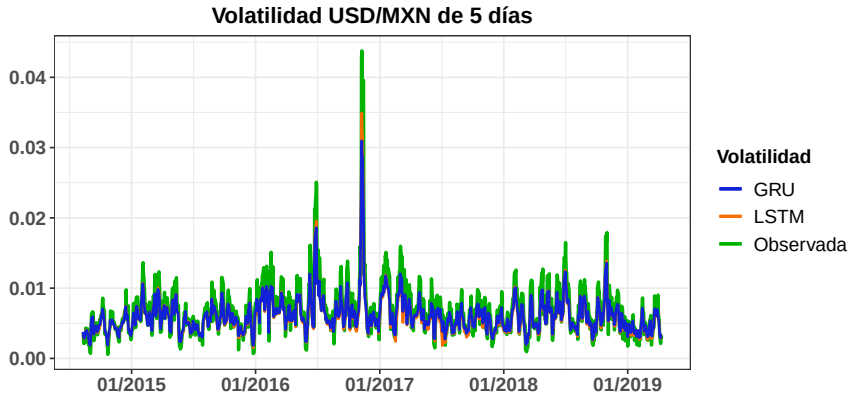


Figura 6.11: Predicciones realizadas mediante las redes LSTM y GRU con el conjunto de prueba.

6.3.4. Modelos Híbridos

Si bien, los modelos de redes neuronales ajustados bajo el enfoque de *machine learning* fueron más eficientes que los de series de tiempo heterocedásticos MA-GARCH y MA-EGARCH ajustados bajo el enfoque clásico, al aumentar la cantidad de datos a predecir que, en este caso corresponde al transcurso del tiempo, los errores de predicción no son los deseados. Al aumentar el tamaño del conjunto de datos, el ajuste de los parámetros debería mejorar ya que se tiene mayor información.

Cuando la cantidad de datos es escasa, los modelos ajustados podrían no ser capaces de generalizar de la forma que se desea. Al ajustar modelos bajo los enfoques clásico y de *machine learning*, se hacen supuestos que no necesariamente serán los mismos como se puede notar en los ajustes realizados anteriormente. Por ello, se propone aprovechar la información que porporcione un modelo ajustado bajo un enfoque y agregarlo como entrada a otro modelo ajustado bajo un enfoque de *machine learning*. Se eligió utilizar, como entrada de una red neuronal, modelos de series de tiempo ajustados bajo el enfoque clásico, además de la volatilidad estimada. A este tipo de modelos se les denominará modelos híbridos.

Debido a que cada uno de los modelos GARCH y EGARCH puede aportar información diferente, se considerarán ambos en lugar de seleccionar uno de ellos.

La nomenclatura de los modelos híbridos será de la forma *TS-RNN*, donde *TS* representa un modelo de series de tiempo (queda implícito que los modelos de series de

tiempo están compuestos por un modelo ARMA y uno heterocedástico condicional) y *RNN* uno de los modelos de red recurrente, ya sea LSTM o GRU.

Sean $x_{gr,t}$, $x_{er,t}$ y $x_{ger,t}$ vectores de entradas al tiempo t de los modelo híbridos GARCH-RNN, EGARCH-RNN y GARCH-EGARCH-RNN, respectivamente. En el modelo GARCH-EGARCH-RNN, se obtiene información de los modelos GARCH y EGARCH para alimentar de forma adicional a una red neuronal recurrente, en este caso, las redes LSTM y GRU. Los vectores de entrada de las redes neuronales se conformaron de la forma siguiente:

$$\begin{aligned} x_{gr,t} &= \kappa_1 * (x_{g,t},\ \hat{\sigma}_t) \\ x_{er,t} &= \kappa_2 * (x_{e,t},\ \hat{\sigma}_t) \\ x_{ger,t} &= \kappa_3 * (x_{g,t},\ x_{e,t},\ \hat{\sigma}_t) \end{aligned}$$

donde

$x_{g,t} = \kappa_g \delta_t \hat{\sigma}_{g,t}$,

$x_{e,t} = \kappa_e \xi_t \delta_t \hat{\sigma}_{e,t}$,

$\hat{\sigma}_t$, $\hat{\sigma}_{g,t}$ y $\hat{\sigma}_{e,t}$ representan las volatilidades estimadas con la ventana de referencia y mediante los modelos GARCH y EGARCH, respectivamente,

$\kappa_g = \left\lfloor \text{máx}\{|\delta_t^{-1}|\} \right\rfloor$,

$\kappa_e = \left\lfloor \text{máx}\{|(|\gamma| + |\theta|)\delta_t|^{-1}\} \right\rfloor$,

$\delta_t = y_t - y_{t-1}$, $\{y_t\}$ rendimientos observados del USD/MXN,

$\kappa_1 = \left\lfloor \text{máx}\{\hat{\sigma}_{g,t}^{-1}\} \right\rfloor$,

$\kappa_2 = \left\lfloor \text{máx}\{\hat{\sigma}_{e,t}^{-1}\} \right\rfloor$,

$\kappa_3 = \text{mín}\{\kappa, \kappa_1, \kappa_2\}$,

$\kappa = \left\lfloor \text{máx}\{\hat{\sigma}_t^{-1}\} \right\rfloor$,

$\xi_t = (\theta - \gamma)\mathbb{I}_{\delta_{1,t}<0} + (\theta + \gamma)\mathbb{I}_{\delta_{1,t}>=0}$, γ, θ corresponden a los coeficientes respectivos del modelo MA(1)-EGARCH$(1,1)$.

La información proporcionada por los modelos GARCH y EGARCH corresponde a $x_{g,t}$ y $x_{e,t}$, respectivamente.

Se consideró la diferencia entre los rendimientos al tiempo t y $t-1$ ajustados porque ello permite tener una idea sobre qué tanto cambiaron los rendimientos del tiempo $t-1$ al tiempo t y en qué sentido (positivo o negativo) y, al multiplicarla por la volatilidad estimada por el modelo heterocedástico, se le da un nivel de relevancia a la estimación realizada por el modelo heterocedástico respecto al tiempo t.

La finalidad de los escalares κ_i es mantener los datos dentro de un rango similar al de las volatilidades estimadas.

Se puede observar que la información proporcionada por el modelo EGARCH permite aprovechar los cambios asimétricos dependiendo de si el rendimiento al tiempo t respecto al día anterior fue positivo o negativo.

Para ajustar los modelos híbridos, se emplearon los modelos ajustados bajo el enfoque clásico, mientras que los de redes neuronales correspondientes a los modelos híbridos se entrenarán bajo los mismos parámetros de entrenamiento de los modelos individuales de redes neuronales, es decir, bajo la misma arquitectura, número de épocas, métodos de optimización de aprendizaje (*dropout*), tasa de aprendizaje, etc.

En la Tabla 6.15 se muestra la matriz de correlación entre las variables de entrada de los modelos híbridos GARCH-EGARCH-RNN. Debido a que las entradas de los modelos híbridos *TS-RNN* son a su vez entradas de los modelos GARCH-EGARCH-RNN, la matriz de correlación de las variables de entrada de los modelos *TS-RNN* se puede obtener a partir de la matriz mostrada en la Tabla 6.15.

	$\{x_{g,t}\}$	$\{x_{e,t}\}$	$\{\hat{\sigma}_{5,t}\}$
$\{x_{g,t}\}$	1.0000	−0.1806	0.6129
$\{x_{e,t}\}$	−0.1806	1.0000	0.0229
$\{\hat{\sigma}_{5,t}\}$	0.6129	0.0229	1.0000

Tabla 6.15: Matriz de correlación entre las variables de entrada de los modelos híbridos GARCH-EGARCH-RNN. Los procesos $\{x_{g,t}\}$ y $\{x_{e,t}\}$ corresponden a la información proporcionada por los modelos GARCH y EGARCH, respectivamente.

Se presentó una baja correlación entre las variables de entrada correspondientes a la información proporcionada por el modelo GARCH y por el EGARCH, la cual fue del $-18.06\,\%$. La correlación entre la información proporcionada por el modelo GARCH y la volatilidad estimada fue del $61.29\,\%$, lo cual indica que ambas variables comparten más de la mitad de la información. Por otra parte, la correlación entre la volatilidad estimada y la información obtenida a partir del modelo EGARCH fue del $2.29\,\%$, por lo que ambas variables están escasamente correlacionadas. Debido a que conocer qué tan correlacionadas están las variables es más relevante para la tarea **T** que si lo están, no se realizó alguna prueba de hipótesis para probar la no correlación.

Los errores de predicción con los conjuntos de entrenamiento y de validación empleando cada una de las redes (LSTM y GRU) se muestran en las Tablas 6.16 y 6.17, respectivamente.

Referente a los modelos integrados por una red LSTM, el modelo que obtuvo los menores errores de predicción bajo las métricas MAE y RMSE con el conjunto de entrenamiento fue el GARCH-EGARCH-LSTM, mientras que bajo la métrica MAPE fue el modelo EGARCH-LSTM. Con el conjunto de validación, el modelo GARCH-LSTM obtuvo los menores puntajes bajo las métricas MAE, RMSE y MAPE.

Para el caso de los híbridos compuestos por la red GRU, el modelo GARCH-GRU obtuvo los menores errores de predicción bajo las métricas MAE y RMSE con el conjunto de entrenamiento, mientras que la red EGARCH-GRU obtuvo el menor MAPE. Con el conjunto de validación, el modelo GARCH-GRU obtuvo los menores puntajes

bajo las métricas MAE, RMSE y MAPE.

	GARCH-LSTM	EGARCH-LSTM	GARCH-EGARCH-LSTM
No. Parámetros	535	535	559
Entrenamiento			
MAE	0.00098265*	0.00101452	0.00099032
RMSE	0.00134905*	0.00139056	0.00135183
MAPE	0.28137042*	0.28929473	0.28272251
Validación			
MAE	0.00116101*	0.00131933	0.00131109
RMSE	0.00178295*	0.00208559	0.00203163
MAPE	0.17886567*	0.18832722	0.19852505

Tabla 6.16: Comparación entre los modelos de híbridos propuestos conformados por la red LSTM y sus respectivos errores de predicción con los conjuntos de entrenamiento y de validación mediante diversas métricas.

	GARCH-GRU	EGARCH-GRU	GARCH-EGARCH-GRU
No. Parámetros	403	403	421
Entrenamiento			
MAE	0.00093050*	0.00096173	0.00098108
RMSE	0.00129388*	0.00132962	0.00135332
MAPE	0.26481139*	0.27366709	0.27996571
Validación			
MAE	0.00122539	0.00131879	0.00122085*
RMSE	0.00202128*	0.00226779	0.00204777
MAPE	0.18189514	0.18763855	0.17968622*

Tabla 6.17: Comparación entre los modelos de híbridos propuestos conformados por la red GRU y sus respectivos errores de predicción con los conjuntos de entrenamiento y de validación mediante diversas métricas.

Con relación a los modelos híbridos conformados por una red LSTM, en la Figura 6.12 se aprecia que, con el conjunto de validación, el modelo EGARCH-LSTM fue el más eficiente al predecir bajas volatilidades, mientras que el GARCH-EGARCH-LSTM lo fue para periodos de alta volatilidad. En periodos donde la volatilidad se encuentra en niveles relativamente normales, el modelo GARCH-LSTM fue más eficiente. El desempeño del modelo GARCH-EGARCH-LSTM fue un equilibrio entre la eficiencia de los otros dos modelos.

Sobre los modelos híbridos conformados por una red GRU, en la Figura 6.13 se aprecia que, con el conjunto de validación, el modelo GARCH-EGARCH-GRU fue el que modela mejor, en promedio, la volatilidad. Para los periodos de baja volatilidad, el más eficiente fue el modelo EGARCH-GRU. Por otra parte, la eficiencia del modelo

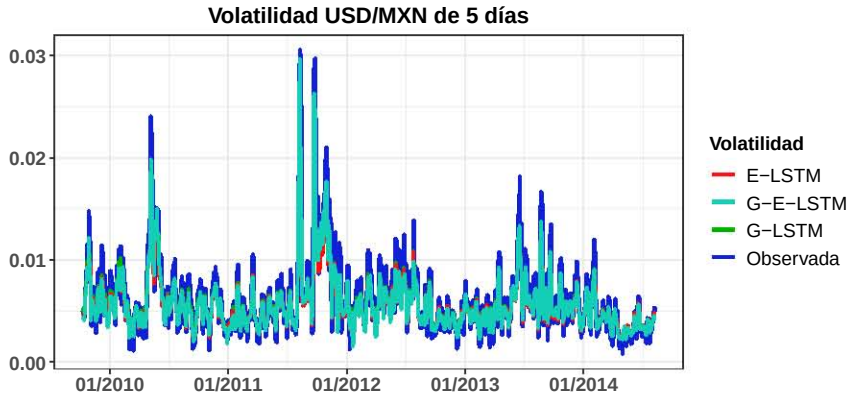


Figura 6.12: Comparación entre las predicciones de los modelos híbridos de la red LSTM propuestos con el conjunto de validación.

GARCH-GRU fue un intermedio de la eficiencia de los otros dos modelos, además de que fue el más eficiente para periodos de alta volatilidad.

Los errores de predicción con el conjunto de prueba se muestran en la Tabla 6.18. El modelo GARCH-LSTM obtuvo el mejor desempeño de los modelos híbridos conformados por una red LSTM bajo todas las métricas y ventanas de tiempo analizadas. En el caso de los modelos híbridos conformados por una red GRU, el más eficiente fue el modelo GARCH-EGARCH-GRU bajo todas las métricas y ventanas de tiempo analizadas.

El modelo GARCH-EGARCH-GRU tuvo menores errores de predicción que el GARCH-LSTM bajo todas las ventanas de tiempo. Además, se puede apreciar que, al aumentar la ventana de tiempo para estimar la volatilidad, el desempeño del modelo GARCH-EGARCH-GRU mejora de forma más notoria que el del modelo GARCH-LSTM.

Respecto a los modelos compuestos por la red LSTM, en la Figura 6.14 se aprecia que, con el conjunto de prueba, se obtuvieron resultados similares a los obtenidos con el conjunto de validación respecto a cuál modelo fue el más eficiente para predecir la volatilidad, es decir, el modelo GARCH-LSTM fue el que modeló mejor la volatilidad en periodos donde ésta se encuentra en niveles relativamente normales, mientras que para los periodos de baja volatilidad, el más eficiente fue el modelo EGARCH-LSTM. Por otra parte, el modelo GARCH-EGARCH-LSTM es un intermedio el cual, a su vez, fue más eficiente para periodos de alta volatilidad.

Respecto a los modelos híbridos conformados por una red GRU, en la Figura 6.15 se aprecia que, con el conjunto de prueba, se obtuvieron resultados similares a los obtenidos con el conjunto de validación respecto a cuál modelo fue el más eficiente para predecir la volatilidad, es decir, el modelo GARCH-GRU fue el que modeló mejor las

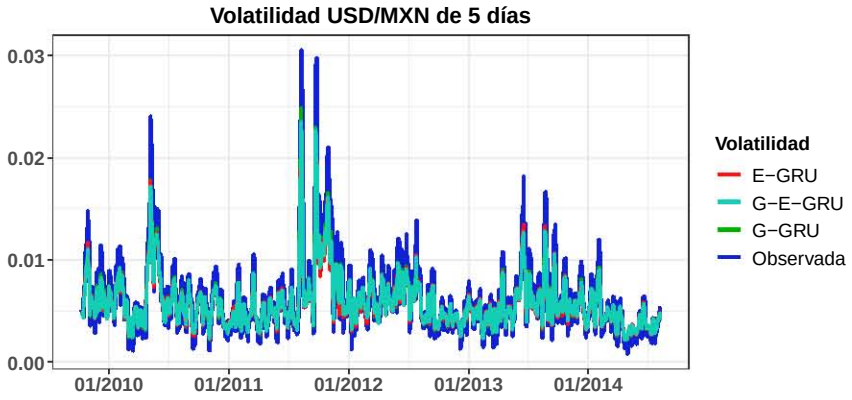


Figura 6.13: Comparación entre las predicciones de los modelos híbridos de la red GRU propuestos con el conjunto de validación.

altas volatilidades, mientras que para los periodos de baja volatilidad, el más eficiente fue el modelo EGARCH-GRU. Por otra parte, el modelo GARCH-EGARCH-GRU fue un intermedio el cual, en promedio, fue el más eficiente al predecir la volatilidad.

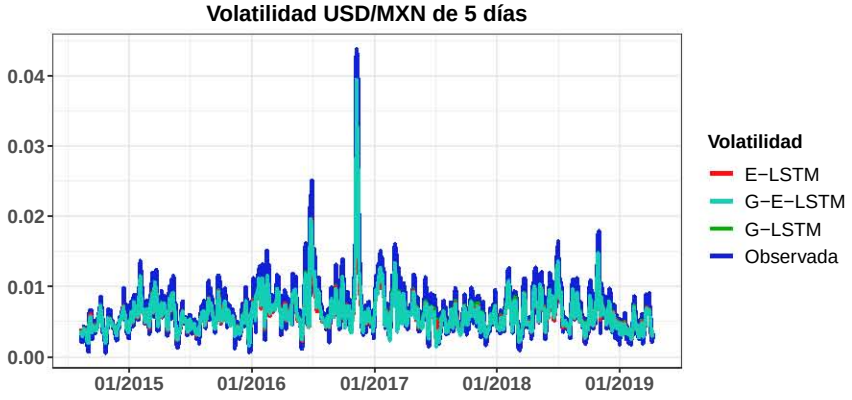


Figura 6.14: Comparación entre las predicciones con el conjunto de prueba de los modelos híbridos conformados por una red LSTM ajustados.

Ventana (días)	Modelo	MAE	RMSE	MAPE
5	G-LSTM	0.00110159⋆	0.00158615⋆	0.17884573⋆
	E-LSTM	0.00117365	0.00168212	0.18379739
	G-E-LSTM	0.00115254	0.00164210	0.18571388
	G-GRU	0.00113511	0.00170096	0.18534835
	E-GRU	0.00116033	0.00176323	0.18686244
	G-E-GRU	0.00110034⋆	0.00163630⋆	0.18096477⋆
10	G-LSTM	0.00088551⋆	0.00122397⋆	0.11840780⋆
	E-LSTM	0.00098983	0.00135169	0.13034863
	G-E-LSTM	0.00095929	0.00131065	0.12946612
	G-GRU	0.00085694	0.00124529	0.11475372
	E-GRU	0.00089879	0.00132493	0.11863983
	G-E-GRU	0.00083862⋆	0.00122111⋆	0.11259320⋆
22	G-LSTM	0.00085586⋆	0.00113225⋆	0.10851621⋆
	E-LSTM	0.00097137	0.00125554	0.12325415
	G-E-LSTM	0.00094556	0.00123912	0.12199673
	G-GRU	0.00078783	0.00109191	0.09888778
	E-GRU	0.00083642	0.00115826	0.10450653
	G-E-GRU	0.00076677⋆	0.00106457⋆	0.09599342⋆

Tabla 6.18: Comparación entre los errores de predicción de los modelos híbridos propuestos con el conjunto de prueba mediante diversas métricas. En la nomenclatura de los modelos, "G-" indica que el modelo está conformado por un modelo MA-GARCH, mientras que "E-" indica que el modelo está conformado por un modelo MA-EGARCH.

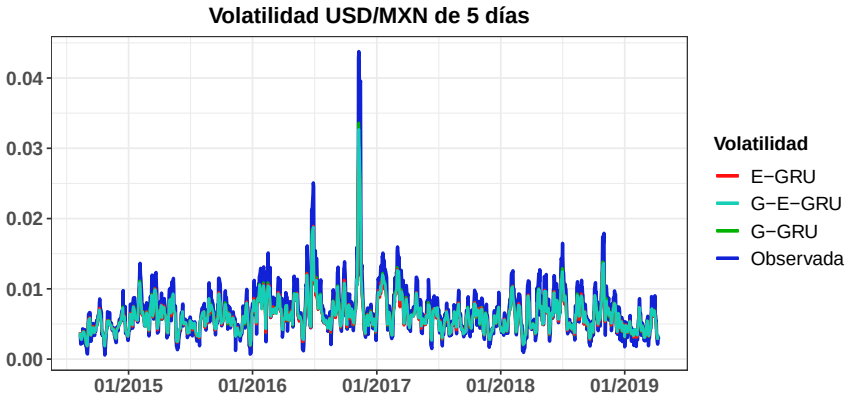


Figura 6.15: Comparación entre las predicciones con el conjunto de prueba de los modelos híbridos conformados por una red GRU ajustados.

6.3.5. Validación en Secuencia

Debido a la aleatoriedad involucrada tanto en la asignación de los parámetros iniciales de los modelos de redes neuronales como en el entrenamiento de la misma (se utilizaron minilotes en lugar del conjunto total de datos de entrenamiento) los parámetros ajustados de las redes neuronales no son replicables, por lo que se realizará un proceso de validación en secuencia.

Por un lado, la validación en secuencia permitirá estimar de mejor forma el riesgo asociado al modelo ajustado $\hat{f}$. Sin embargo, debido a la no replicabilidad de los modelos ajustados, se tomaron cinco escenarios posibles para cada uno de los casos. De esta forma, para cada escenario, se toma el promedio del valor de la función de costos obtenida en cado de los ajustes realizados en la validación en secuencia. Se eligieron los tamaños del conjunto de entrenamiento de $0.1, 0.2, 0.3$ y 0.4 respecto al conjunto de datos y 5 escenarios, resultando un total de 160 modelos a ajustar (8 tipos de modelos, 4 tamaños para el conjunto de entrenamiento y 5 escenarios).

La asignación de los tamaños para los conjuntos de validación y de prueba fue dinámica, estando ambos conformados por la misma proporción respecto del total de conjunto de datos, para cada una de las proporciones del conjunto de entrenamiento respecto al conjunto de datos, por ejemplo, para 0.2, los conjuntos de validación y de prueba se conformaron, cada uno, por el $40\,\%$ de los datos. Si bien una opción es reservar un subconjunto de tamaño fijo para el conjunto de prueba, se consideró más eficiente la asignación dinámica de los tamaños de los conjuntos de validación y de prueba porque, a menor cantidad de datos, se esperaría que la capacidad de generalización de los modelos disminuya. Con el objetivo de minimizar estos efectos y debido a la estocasticidad de la volatilidad estimada se realizó este tipo de asignación dinámica. Los resultados, por subconjunto del conjunto de datos, se muestran en la Tabla 6.19.

Respecto a los modelos conformados por una red LSTM, el modelo GARCH-EGARCH-LSTM fue el más eficiente bajo la métrica MAPE. Bajo la métrica MAE, la red GARCH-LSTM fue la más eficiente según el riesgo estimado empleando el conjunto de prueba, mientras que, bajo los conjuntos de entrenamiento y de validación, el más eficiente fue el GARCH-EGARCH-LSTM.

De los modelos que involucran una red GRU, el más eficiente fue el modelo GARCH-EGARCH-GRU con excepción del riesgo estimado empleando el conjunto de entrenamiento, en el cual el más eficiente fue el modelo GARCH bajo las métricas MAE y MAPE.

Considerando los modelos propuestos tanto individuales de redes neuronales como híbridos, el modelo GARCH-EGARCH-GRU fue el más eficiente. Por otro lado, los modelos EGARCH-RNN mostraron, en algunos casos, un peor desempeño que la respectiva red neuronal (modelo individual de red neuronal) considerando todas las ventanas de tiempo y las tres métricas. De aquí se puede concluir que agregar información no necesariamente permitirá que, al ajustar un modelo, mejore la eficiencia del modelo al realizar una tarea específica.

Conjunto	Modelo	MAE	RMSE	MAPE
Entrenamiento	LSTM	0.001141658	0.001541104	0.3154625
	G-LSTM	0.001097996	0.001479185*	0.3062223
	E-LSTM	0.001121536	0.001516692	0.3096171
	G-E-LSTM	0.001097903*	0.001481170	0.3044605*
	GRU	0.001061631	0.001451078	0.2941724
	G-GRU	0.001029733*	0.001405180	0.2862084*
	E-GRU	0.001071919	0.001457666	0.2994306
	G-E-GRU	0.001030305	0.001403957*	0.2875209
Validación	LSTM	0.001301489	0.002050408	0.2418799
	G-LSTM	0.001197624	0.001823556	0.2341170
	E-LSTM	0.001278936	0.002019684	0.2363595
	G-E-LSTM	0.001195335*	0.001788601*	0.2322128*
	GRU	0.001207093	0.002128992	0.2169989
	G-GRU	0.001129067	0.001835566	0.2159457
	E-GRU	0.001209842	0.002116977	0.2212154
	G-E-GRU	0.001110439*	0.001821866*	0.2131208*
Prueba	LSTM	0.001324580	0.001875609	0.2055447
	G-LSTM	0.001239302*	0.001732033*	0.1991014
	E-LSTM	0.001292701	0.001844759	0.1998437
	G-E-LSTM	0.001240734	0.001740998	0.1969695*
	GRU	0.001195419	0.001793496	0.1903550
	G-GRU	0.001170343	0.001723186	0.1909078
	E-GRU	0.001187755	0.001780560	0.1909019
	G-E-GRU	0.001146756*	0.001704371*	0.1889916*

Tabla 6.19: Riesgo estimado de cada una de los modelos híbridos propuestos para cada uno de los subconjuntos de datos (entrenamiento, validación y prueba) ($\hat{R}(\hat{f})$) bajo la ventana de 5 días, diversas funciones de costos y mediante validación en secuencia. En la nomenclatura de los modelos, "G-" indica que el modelo está conformado por un modelo MA-GARCH, mientras que "E-" indica que el modelo está conformado por un modelo MA-EGARCH.

6.4. Resultados

Los modelos matemáticos no suelen ajustarse a los datos reales, por lo que se deben realizar ciertas pruebas para verificar que se cumplen los supuestos de los modelos propuestos. Bajo el enfoque clásico, si los modelos requieren ajustarse constantemente, supóngase de forma diaria, se vuelve una tarea poco eficiente ya que se debe verificar, cada vez que se ajustan los parámetros del modelo, que se cumplen los supuestos del modelo, además de que no necesariamente se cumplirán todos los supuestos en todo al contar con mayor información, es decir, al aumentar el tamaño del conjunto de datos, por lo que podría ser necesario proponer algún otro tipo de modelo.

Una alternativa es el enfoque de *machine learning*, ya que los modelos requieren de menos supuestos que en los modelos ajustados bajo los enfoques frecuentista o bayesiano. Basta con un poco de programación para ajustar un modelo las veces que sean necesarias y que a su vez podría ser más eficiente, aunque computacionalmente podría ser costoso y los resultados podrían no ser favorables, es decir, la capacidad de generalización de los modelos ajustados mediante el enfoque de *machine learning* podrían ser menos eficientes al realizar una tarea predefinida **T** que los ajustados mediante el enfoque clásico.

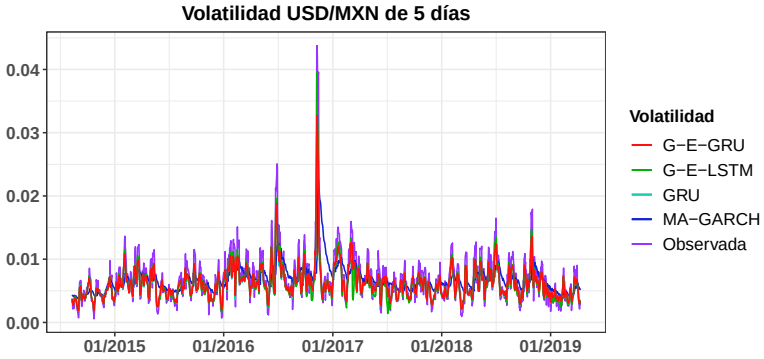
Ambos ofrecen ventajas y desventajas, por lo que ajustar modelos considerando ambos enfoques y compararlos sería una buena idea. Sin embargo, contar con un modelo conformado por modelos ajustados bajo diferentes enfoques estadísticos podría mejorar los resultados obtenidos por los modelos de forma individual.

Con base en los resultados obtenidos en la sección 6.3, el modelo MA(1)-GARCH(1,1) fue más eficiente al predecir la volatilidad al tiempo $t+1$, es decir, a un día, que el MA(1)-EGARCH(1,1) ya que tuvo, en promedio, menores errores de predicción a un día. En cuanto a predicciones a largo plazo (a tiempo s dada la información hasta tiempo t, $s > t$), el modelo MA(1)-GARCH(1,1) también fue más eficiente. Sin embargo, en ambos modelos, los errores relativos de predicción (MAPE) con respecto a la volatilidad de 5 días fueron ¡mayores al 37 %! Considerando la volatilidad de los últimos 22 días la cual es más estable, los errores relativos de predicción d elos modelos MA(1)-GARCH(1,1) y MA(1)-EGARCH(1,1) fueron del 11.07 % % y 14.88 %, respectivamente.

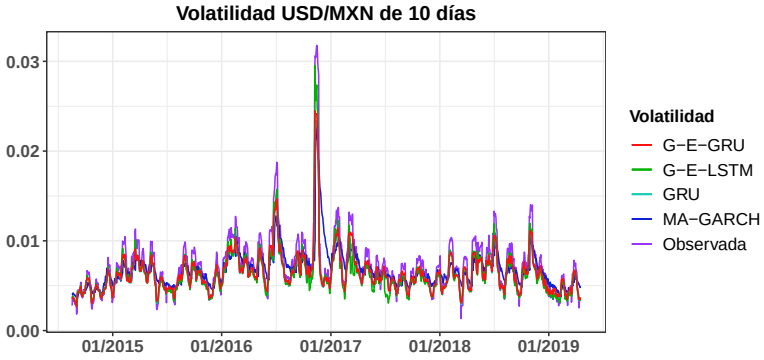
Las redes neuronales recurrentes, por su parte, mostraron un mejor desempeño. A pesar de no ser profundas, puesto que ambas estuvieron constituidas de seis celdas, ya sea de LSTM o GRU según corresponda, en una capa oculta, y no se tomó mucha importancia a la optimización del entrenamiento de las redes neuronales, los errores de predicción fueron inferiores. De haber sido entrenadas las redes de mejor forma, las diferencias en la eficiencia de los modelos podrían ser más difíciles de detectar.

Los errores de predicción con el conjunto de prueba de las redes neuronales fueron inferiores a los de los modelos MA(1)-GARCH(1,1) y MA(1)-EGARCH(1,1) respecto a las métricas MAE, RMSE y MAPE para las ventanas de 5 y 10 días; para la ventana de 22 días, el error de las redes fue inferior bajo las métricas RMSE y MAPE, mientras que bajo la métrica MAE, el más eficiente fue el MA(1)-GARCH(1,1), seguido por la red

a) Volatilidad de 5 días.



b) Volatilidad de 10 días.



c) Volatilidad de 22 días.

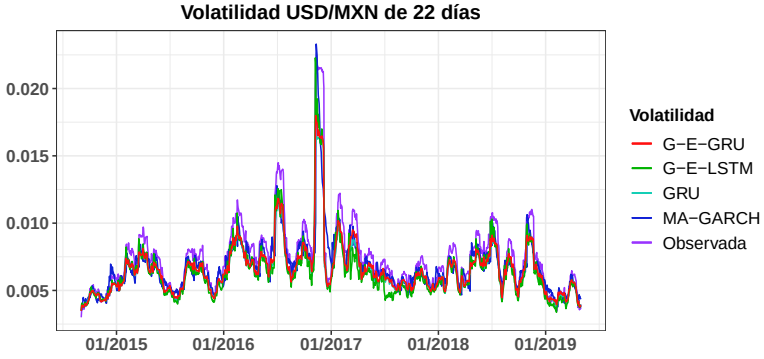


Figura 6.16: Predicción de la volatilidad de 5 (a), 10 (b), y 22 días (c) mediante los modelos MA-EGARCH EGARCH-LSTM y GARCH-EGARCH-GRU con el conjunto de prueba.

GRU, la red LSTM y el MA(1)-EGARCH(1,1). Para momentos de alta incertidumbre, el modelo elegido fue la red LSTM, mientras que para periodos de baja incertidumbre se eligió la red GRU. De forma general, se escogió a la red GRU como el mejor modelo de los ajustados tanto bajo el enfoque clásico como del enfoque de *machine learning* (modelos individuales).

Para comparar los modelos híbridos con los modelos individuales de redes neuronales se tomaron en consideración los resultados obtenidos en la validación en secuencia.

De los modelos conformados por una red LSTM, los más eficientes fueron el GARCH-LSTM y GARCH-EGARCH-LSTM, de los cuales se seleccionó el GARCH-EGARCH-LSTM ya que el RMSE bajo el conjunto de entrenamiento fue superior lo cual en parte se debió a que el GARCH-LSTM está conformado por una menor cantidad de parámetros a ajustar que el GARCH-EGARCH-LSTM. Esto, a su vez, indica que, al mejorar el ajuste de ambos modelos, GARCH-LSTM y GARCH-EGARCH-LSTM, de tal forma que se obtenga un RMSE muy similar entre ambos modelos, el GARCH-EGARCH-LSTM podría ser más eficiente, y cabe la posibilidad de que su eficiencia supere a la del GARCH-LSTM. Sin embargo, se puede optar por el GARCH-LSTM ya que éste tiene menor cantidad de parámetros a ajustar.

Respecto a los modelos conformados por una red GRU, el modelo GARCH-EGARCH-GRU fue el más eficiente. El modelo GARCH-GRU fue más eficiente que el modelo GRU bajo las ventanas con los conjuntos de entrenamiento y de validación según las métricas MAE y RMSE, mientras que con el conjunto de prueba ocurrió lo opuesto.

Los modelos híbridos GARCH-EGARCH-RNN fueron más eficientes que los modelos LSTM y GRU bajo los tres conjuntos (entrenamiento, validación y prueba) y las métricas MAE, RMSE y MAPE.

Considerando el total de modelos ajustados, el modelo más eficiente fue el GARCH-EGARCH-GRU.

En la Figura 6.16 se muestran las predicciones de los modelos seleccionados con el conjunto de prueba con las ventanas de 5, 10 y 22 días, respectivamente, de la cual se puede deducir que, al aumentar la ventana de tiempo para estimar la volatilidad, incrementa la eficiencia de los modelos MA-GARCH y MA-EGARCH.

De igual forma, para la ventana de 5 días, el modelo GARCH-EGARCH-LSTM mostró ser el más adecuado para momentos de alta volatilidad, mientras que para niveles normales y bajos de la volatilidad, el modelo GARCH-EGARCH-GRU fue el más eficiente. La red LSTM fue la menos eficiente de los modelos conformados por alguna red neuronal.

Para las ventanas de 10 y 22 días, la red GARCH-EGARCH-LSTM fue la más eficiente para momentos de baja volatilidad y de relativa calma, mientras que para momentos de alta incertidumbre, la red GARCH-EGARCH-LSTM y el modelo MA-GARCH resultaron ser los más eficientes bajo las ventanas de 10 y 22 días, respectivamente.

Así, para periodos donde la volatilidad se encuentra en niveles normales, el modelo

GARCH-EGARCH-LSTM es el más eficiente. En cambio, en periodos ya sea de baja o alta volatilidad, la red GARCH-EGARCH-GRU tiene un mejor desempeño.

Conclusiones

Debido a que los modelos raramente representan la realidad, es de gran ayuda considerar los posibles estados de la naturaleza del problema, lo que conlleva a emplear más de un modelo para elegir aquél que sea más eficiente al realizar la tarea deseada.

Cuando los modelos usualmente propuestos de acuerdo a la tarea **T** y a los datos[6] no se ajustan de forma adecuada a los datos, se deben ampliar los horizontes. Las redes neuronales permiten modelar el comportamiento de los datos (aproximar una infinidad de funciones) bajo pocos supuestos pero pagando con recursos computacionales.

Contar con más de dos modelos y elegir el que mejor desempeño tenga es una opción recomendable porque no existe un modelo que sea el mejor para resolver todos los problemas.

Cuando los datos con los que se cuenta son insuficientes, o se busca ampliar el conjunto de datos, existen varias opciones entre las que se encuentra la simulación de datos.

Al combinar al menos dos modelos diferentes entre sí, cada uno con sus ventajas y desventajas, el modelo resultante podría ser más eficiente al realizar una tarea específica **T**. Ya sea que alguno de los modelos permitar transformar los datos de forma que el ajuste de los parámetros de otro modelo sea más sencillo, o para obtener mayor información con los datos con los que se cuenta. El modelo híbrido resultante aumenta de complejidad y no por ello ser más eficiente que los modelos individuales que lo conforman. Sin embargo, si se desea mejorar la eficiencia de los modelos aún contando con escasos datos, es buena idea ajustar modelos híbridos.

Como en todos los modelos, a mayor información relevante, el ajuste puede mejorar. Si se agrega información irrelevante a un modelo, la eficiencia del modelo ajustado podría no ser la deseada y empeorar en comparación de si no se agregase dicha información. Al ingresar información preprocesada de los modelos de series de tiempo bajo el enfoque frecuentista a los modelos de redes neuronales cabe la posibilidad de que la red pueda realizar mejores predicciones. Si bien, los modelos individuales de redes recurrentes fueron más eficientes que los de series de tiempo, los modelos híbridos lo fueron aún más.

[6] Al hablar de modelos usuales se alude a los modelos que comúnmente se proponen para problemas de cierta índole. Por ejemplo, en las series de tiempo, los modelos homocedásticos usuales son los autorregresivos - medias móviles o ARMA, mientras que los modelos heterocedásticos usuales son los GARCH.

Con relación a los modelos ajustados, cuando existe un ambiente donde la incertidumbre se considera alta, el modelo GARCH-EGARCH-LSTM sería el más conveniente de emplear, mientras que cuando la volatilidad se encuentra en niveles relativamente normales o bajos, el modelo más eficiente resultó ser el GARCH-EGARCH-GRU. Es importante destacar que las redes neuronales no fueron entrenadas para cada una de las ventajas de tiempo, ni se puso énfasis en la optimización del proceso de aprendizaje, por lo que su eficiencia puede mejorar aún más.

Si bien el ajuste de los modelos híbridos propuestos son computacionalmente costosos, se puede optar por considerarlos o no. Además, el diseño de las redes puede hacerse más complejo, así como optimizar los algoritmos de aprendizaje, lo cual podría mejorar la eficiencia de los modelos ajustados mediante un enfoque de *machine learning*.

Los modelos híbridos GARCH-EGARCH-GRU fueron más eficientes al predecir la volatilidad estimada de la paridad dólar-peso mediante ventanas de 5, 10 y 22 días que los modelos MA-GARCH, MA-EGARCH y GRU (por separado). Lo mismo ocurrió con los modelos GARCH-EGARCH-LSTM. Los modelos EGARCH-LSTM y EGARCH-GRU fue menos eficientes que las redes LSTM y GRU, respectivamente, en algunos casos (algunas ventanas de tiempo y algunas métricas) lo cual es un ejemplo de que, al agregar información, no necesariamente el modelo ajustado tendrá mayor capacidad de generalización.

Cuando se requiere reajustar los parámetros del modelo con frecuencia, por ejemplo cada día, el modelo híbrido parece ser el menos factible de entrenar a comparación de los modelos individuales. En el modelo ajustado GARCH-EGARCH-LSTM se requeriría ajustar los modelos MA-GARCH y MA-EGARCH, y verificar supuestos de dichos modelos. Esto no es necesario, ya que la red neuronal es el modelo que devuelve el valor de la predicción, por lo que bastaría con ajustar los modelos MA-GARCH y MA-EGARCH cada cierto tiempo, ya sea cada mes o cada trimestre, mientras que la red LSTM reajustarse únicamente si se considera pertinente.

Los modelos híbridos fueron más recomendables pero es importante destacar que los costos computacionales de su ajuste son mucho mayores a los del ajuste de cada uno de los modelos que los conforman. Si la mejora en la eficiente (reducción de los errores de generalización) son convincentes a criterio propio y el costo computacional es aceptable, entonces el ajuste de modelos híbridos es una buena idea, de lo contrario, los modelos de redes neuronales pueden ser suficientes.

Al mejorar la eficiencia de los modelos empleados para predecir la volatilidad, se pueden tomar mejores decisiones ya que, si se predice que la volatilidad aumentará en los próximos días, podría ser un momento adecuado para comprar opciones sobre tipo de cambio o, considerando diversas variables macroeconómicas y el sentimiento del mercado, optar por adquirir cierta cantidad de alguna de las divisas, ya sea dólares o pesos. A su vez, el proceso de valuación de derivados sobre tipo de cambio podría mejorar en el sentido de que, al estimar mejor la volatilidad, el cálculo del precio del derivado será más acertado lo cual, a su vez, ayudará a construir y administrar de forma más eficiente el portafolio para quien vende la opción (En el caso de una opción, gene-

ralmente recibe una cantidad de dinero. El vendedor de la opción tiene que constuir un portafolio para poder hacer frente a la posibilidad de que el cliente le ejerza la opción).

Emplear más de un modelo en la toma de decisiones proporcionará mejores resultados que utilizar uno solo, pero antes de declarar un modelo como el más eficiente de todos se debe tener un sustento. En el caso de las redes neuronales, se recomienda emplear algún tipo de validación y optimizar el proceso de aprendizaje.

Bibliografía

[1] BANCO DE MÉXICO. Tipo de cambio. http://educa.banxico.org.mx/banco_mexico_banca_central/sist-finc-tipo-cambio.html.

[2] BAO, W., YUE, J., AND RAO, Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLOS ONE 12*, 7 (07 2017).

[3] BAXTER, M., AND RENNIE, A. *Financial Calculus: An Introduction to Derivative Pricing*. Cambridge University Press, 1996.

[4] BELKIN, M., HSU, D., MA, S., AND MANDAL, S. Reconciling modern machine learning and the bias-variance trade-off. *arXiv e-prints* (Dec 2018), arXiv:1812.11118.

[5] BOLLERSLEV, T. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 31*, 3 (1986).

[6] CHO, K., VAN MERRIENBOER, B., GULCEHRE, C., BAHDANAU, D., BOUGARES, F., SCHWENK, H., AND BENGIO, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv e-prints* (Jun 2014), arXiv:1406.1078.

[7] COHEN, S. N., AND ELLIOTT, R. J. *Stochastic Calculus and Applications*. Springer, New York, NY, 2015.

[8] DU, K. L., AND SWAMY, M. N. S. *Neural networks in a softcomputing framework*. Springer, London, 2006.

[9] DU, K.-L., AND SWAMY, M. N. S. *Neural networks and statistical learning.* Springer, Londres, 2014.

[10] DUCHI, J., HAZAN, E., AND SINGER, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research 12*, 7 (2011), 2121 – 2159.

[11] ENGLE, R. F. Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica 50*, 4 (1982), 987–1007.

[12] FINANCIERA NACIONAL DE DESARROLLO AGROPECUARIO, RURAL, FORESTAL Y PESQUERO. *Circular 3/2012 del Banco de México*. México, 2016.

[13] FRANCQ, C., AND ZAKOIAN, J. *GARCH Models: Structure, Statistical Inference and Financial Applications*. Wiley, 2011.

[14] GAL, Y., AND GHAHRAMANI, Z. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. *arXiv e-prints* (Dec 2015), arXiv:1512.05287.

[15] HASTIE, T., TIBSHIRANI, R., AND FRIEDMAN, J. H. *The elements of statistical learning: data mining, inference, and prediction.* Springer series in statistics. Springer, New York, 2009.

[16] HOCHREITER, S., AND SCHMIDHUBER, J. Long short-term memory. *Neural computation 9* (12 1997), 1735–80.

[17] HSUEH, B. Y., LI, W., AND WU, I.-C. Stochastic Gradient Descent with Hyperbolic-Tangent Decay on Classification. *arXiv e-prints* (Jun 2018), arXiv:1806.01593.

[18] IOFFE, S., AND SZEGEDY, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv e-prints* (Feb 2015), arXiv:1502.03167.

[19] KINGMA, D. P., AND BA, J. Adam: A method for stochastic optimization. *arXiv e-prints* (Dec. 2014), arXiv:1412.6980.

[20] KNOX, S. W. *Machine learning: a concise introduction*. Wiley series in probability and statistics. Wiley, 2018.

[21] KWOK, Y.-K. *Mathematical models of financial derivatives*. Springer finance. Springer, Berlin, 2008.

[22] MITCHELL, T. M. *Machine Learning*, 1 ed. McGraw-Hill, New York, USA, 1997.

[23] MONTAVON, G., ORR, G., AND MÜLLER, K.-R. *Neural networks: tricks of the trade.* Lecture notes in computer science: 7700. Springer, 2012.

[24] PAPARODITIS, E., AND POLITIS, D. N. The asymptotic size and power of the augmented Dickey–Fuller test for a unit root. *Econometric Reviews 37*, 9 (2018), 955–973.

[25] PASCANU, R., MIKOLOV, T., AND BENGIO, Y. On the difficulty of training Recurrent Neural Networks. *arXiv e-prints* (Nov 2012), arXiv:1211.5063.

[26] REDDI, S. J., KALE, S., AND KUMAR, S. On the convergence of adam and beyond. *CoRR abs/1904.09237* (2018).

[27] ROBERT, C. P. *The bayesian choice: from a decision-theoretic foundations to computational implementation.* Springer texts in statistics. Springer, 2007.

[28] SEIFFERTT, J., AND WUNSCH, D. C. Backpropagation and ordered derivatives in the time scales calculus. *IEEE Transactions on Neural Networks 21*, 8 (Aug 2010).

[29] SERRA, R., AND RODRÍGUEZ, A. C. The Ljung-Box test as a performance indicator for vircs. In *International Symposium on Electromagnetic Compatibility - EMC EUROPE* (Sep. 2012), pp. 1–6.

[30] SILVA, I. N. D., HERNANE SPATTI, D., ANDRADE FLAUZINO, R., LIBONI, L. H. B., AND DOS REIS ALVES, S. F. *Artificial neural networks: a practical course*. Springer, Cham, 2017.

[31] SKANSI, S. *Introduction to deep learning: from logical calculus to artificial intelligence.* Undergraduate topics in computer science. Springer, 2018.

[32] THEODOSSIOU, P. Skewed generalized error distribution of financial assets and option pricing. *Multinational Finance Journal 19*, 4 (2015), 223 – 266.

[33] WEI, W. W. S. Time series analysis: Univariate and multivariate methods. *Journal of the American Statistical Association 86*, 413 (1991), 245.