Proyección del comportamiento de enlaces en redes inalámbricas LLN mediante series temporales no estacionarias aplicando algoritmos de aprendizaje automático

Juan David Mantilla López

Trabajo de Grado para optar el Título de Ingeniero de Sistemas

Director

Pedro Javier Trujillo Tarazona

Magíster en Informática

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingeniería de Sistemas e Informática

Bucaramanga

2021

Dedicatoria

Quiero dedicar este proyecto de grado a toda mi familia, en especial a mi madre Junia y a mi padre Jairo, quienes siempre confiaron y apoyaron incondicionalmente mis estudios, además de ser ellos el pilar fundamental para sostenerme ante adversidades y ser el motor de motivación para darles este título y muchos otros éxitos en mi vida profesional.

A mis hermanas Ana y Paola, que siempre creyeron en mí y en mis capacidades para superarme, que con su ejemplo y experiencia me proyecto hoy en día en ser un mejor ciudadano, hermano y amigo.

A mis verdaderos amigos, que fueron ellos los que me acompañaron en este camino llenándome de conocimiento, debates, risas y situaciones.

Agradecimientos

Agradezco a la Universidad Industrial de Santander por darme la oportunidad de formarme en tan prestigiosa institución, gracias a ella mejoré en gran manera como persona y profesional.

También agradezco a la Escuela de Ingeniería de Sistemas que, con su gran talento humano, me ayudó a entender el mundo de la computación y la informática, desde sus fundamentos teóricos hasta sus aplicaciones que nos benefician hoy día.

A mi director Pedro Javier Trujillo quien, gracias a su dirección, me facilitó el entendimiento de los conceptos que implican un proyecto investigativo.

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Lista de apéndices

Los apéndices deben ser consultados en la base de datos de la Biblioteca.

Glosario

Bagging: Metodología que emplea varios algoritmos de aprendizaje automático para solucionar problemas de clasificación o regresión.

Crawdad: Portal de alojamiento de conjuntos de datos enfocados en redes de comunicaciones inalámbricas.

Dickey-Fuller*:* Prueba de hipótesis que determina la estacionalidad de una serie temporal a través del cálculo de raíces unitarias.

Dropout: Funcionalidad que reduce el sobreajuste en redes neuronales artificiales.

GPU: Hardware especializado para el procesamiento gráfico

IEEE: Instituto de Ingeniería Eléctrica y Electrónica que crea y especifica estándares usados alrededor del mundo.

Kernel: Componente en SVR que permite mapear los datos a un espacio de mayor dimensión.

TPU: Hardware especializado para el procesamiento tensorial en problemas de aprendizaje automático

Walk Forward Validation: Metodología que genera predicciones a través de la ampliación del conjunto de entrenamiento de forma iterativa.

Wireless Sensor Network: Red interconectada de nodos inalámbricos computacionales que pueden ejecutar tareas conjuntamente o de forma individual.

Lista de acrónimos

ARIMA: Autoregressive Integrated Moving Average

GPU: Graphics Proccessing Unit

IEEE: Institute of Electrical and Electronics Engineers

LQI: Link Quality Indicator

LSTM: Long Short-Term-Memory

MAE: Mean Absolute Error

MAPE: Mean Absolute Percentage Error

MSE: Mean Squared Error

RBF: Radial Basis Function

RF: Random Forest

RSSI: Received Strength Signal Indicator

SVR: Support Vector Regressor

TPU: Tensor Processing Unit

WFV: Walk Forward Validation

WLLN: Wireless Low power and Lossy Network

Resumen

Título: Proyección del comportamiento de enlaces en redes inalámbricas LLN mediante series temporales no estacionarias aplicando algoritmos de aprendizaje automático[*]

Autor: Juan David Mantilla López[**]

Palabras Clave: IEEE 802.15.4, series de tiempo no estacionaria, LSTM, Random Forest, Support Vector Regressor, ARIMA, predicciones series de tiempo.

Descripción: Las series temporales son idóneas para representar, a través del tiempo, la calidad de los enlaces en redes IEEE 802.15.4. Por ende, es considerable proyectar o predecir las métricas de la calidad de enlace como RSSI y LQI para optimizar el rendimiento en este tipo de redes. Este proyecto se sustenta en el uso de algoritmos de aprendizaje automático para predecir o proyectar métricas de la calidad de enlaces en series temporales univariantes no estacionarias. A lo largo de este documento se introducen conceptos teóricos-matemáticos de las series temporales, los algoritmos aplicados (RF, SVR, LSTM y ARIMA), pretratamiento y tratamiento de los datos, experimentación, resultados y evaluación de los modelos. Se concluye que, para la media de los 104 experimentos ejecutados, la metodología 'Walk Forward Validation' es de beneficio para el modelo ARIMA, ya que permite una buena precisión en las observaciones, pero un alto tiempo de ejecución, superando en este factor a los demás modelos. Por su parte, RF se asemeja notoriamente a LSTM en términos de error, presentando RF un tiempo de ejecución menor. Experimentalmente SVR obtuvo mejora notoria al aplicar promedios móviles a las series temporales, demostrando una disminución del error porcentual de más del 6%.

[*] Trabajo de Grado

[**] Facultad de ingenierías fisicomecánicas. Escuela de ingeniería de sistemas e informáticas. Director: Pedro Javier Trujillo Tarazona. Mg. En Informática

Abstract

Title: Behavior estimation of the links in LLN wireless networks through non-stationary series applying machine learning algorithms [*]

Author: Juan David Mantilla López[**]

Key Words: IEEE 802.15.4, non-stationary time series, LSTM, Random Forest, Support Vector Machine, ARIMA, time series forecasting.

Description: Time series are well suited to represent, over time, the quality of links in IEEE 802.15.4 networks. Therefore, it is considerable to project or predict link quality metrics such as RSSI and LQI to optimize the performance in this type of networks. This project is based on the use of machine learning algorithms to predict or project link quality metrics in univariate non-stationary univariate time series. Throughout this paper, theoretical-mathematical concepts of time series, applied algorithms (RF, SVR, LSTM and ARIMA), data pretreatment and treatment, experimentation, results and model evaluation are introduced. It is concluded that, for the average of the 104 experiments executed, the 'Walk Forward Validation' methodology is beneficial for the ARIMA model, since it allows a good precision in the observations, but a high execution time, surpassing the other models in this factor. On the other hand, RF is notoriously like LSTM in terms of error, with RF having a shorter execution time. Experimentally, SVR obtained a notorious improvement when applying moving averages to the time series, showing a decrease in the percentage error of more than 6%.

[*] Degree Work

[**] Mechanical-Physical Engineering Faculty. Systems and Computer Engineering School. Director: Pedro Javier Trujillo Tarazona. Mg. in Computer Science.

Introducción

Los dispositivos diseñados bajo el estándar IEEE 802.15.4 son enfocados hacia redes inalámbricas de baja potencia y con pérdidas *WLLN (Wireless Low power and Lossy Network),* caracterizados por un consumo mínimo de energía (alimentado por baterías o pilas) y cómputo reducido. Uno de los obstáculos recurrentes, es la retransmisión de tramas y creación de caminos, esto es provocado por una baja calidad de los enlaces presentes en los nodos de la red. Por esta razón, es importante comprender y predecir el comportamiento de la calidad de los enlaces en forma de series temporales. Métricas como *RSSI (Received Strength Signal Indicator)* y *LQI (Link Quality Indicator)* son relevantes para definir este aspecto. Para predecir dichas métricas, se implementaron algoritmos como *Random Forest, Support Vector Regressor, ARIMA* y *LSTM,* en donde se evaluó el desempeño de la predicciones generadas por medio de métricas de error y tiempos de ejecución.

En la sección de series temporales se introducen los conceptos teóricos de estacionalidad, tendencia, media móvil y componente aleatorio. En las secciones de los modelos, se especifica, teórica y matemáticamente su funcionamiento, características y los recursos para la implementación tecnológica. En la sección métricas, se describen las ecuaciones de error para problemas de regresión o predicción. En el apartado de metodologías se resalta el origen, pretratamiento y tratamiento de los datos, la topología de red, el proceso de automatización para generar las series temporales univariantes, la metodología general del proyecto y las metodologías individuales de los modelos. También se detallan las tecnologías usadas, como el lenguaje de programación, entornos de ejecución, librerías o bibliotecas. Por último, en la sección de resultados se resumen las métricas y tiempos de ejecución para los 104 experimentos ejecutados.

1 Objetivos

1.1 Objetivo General

Evaluación de la predicción del comportamiento de los enlaces en redes LLN tipo IEEE 802.15.4 mediante series temporales no estacionarias y aplicando algoritmos de aprendizaje automático.

1.2 Objetivos Específicos

Evaluar un algoritmo de árboles regresores *(RF, Random Forest)* para la proyección del comportamiento de los enlaces en redes IEEE 802.15.4.

Evaluar un algoritmo de vector de soporte regresivo (*SVR, Support Vector Regressor*) para la proyección del comportamiento de los enlaces en redes IEEE 802.15.4.

Evaluar un algoritmo de redes neuronales LSTM (*Long Short Term-Memory*) para la proyección del comportamiento de los enlaces en redes IEEE 802.15.4.

2 Cuerpo del Trabajo

2.1 Marco Referencial

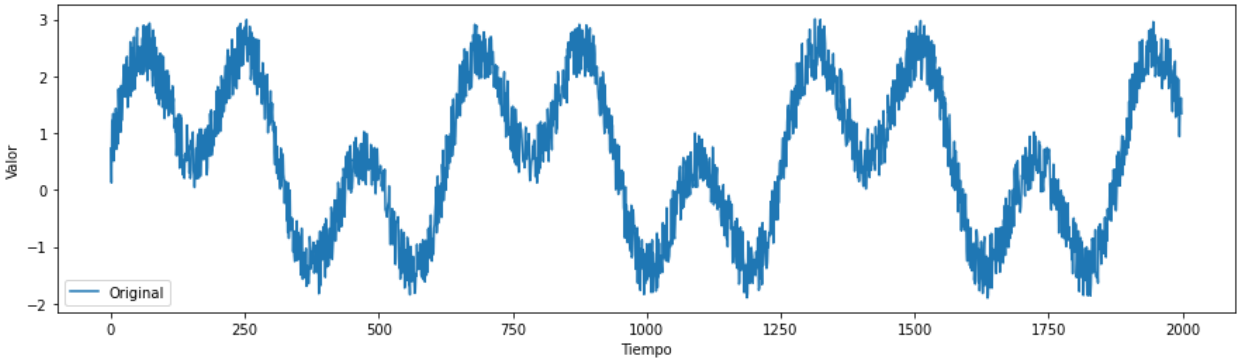
En ese apartado se definirán los conceptos teóricos y matemáticos de las series temporales univariantes, la tendencia y la autocorrelación, los modelos de RF, SVR, ARIMA y LSTM. Igualmente, se muestran los conceptos de las métricas de error tenidos en cuenta para la evaluación cuantitativa de los modelos.

2.1.1 Series temporales

Las series temporales son, en esencia, un conjunto de observaciones de una variable dependiente con respecto al tiempo y representan las variaciones o comportamiento de un evento natural o artificial. Estas observaciones tienen la particularidad que están definidas en intervalos de tiempo y en orden cronológico. Los componentes de las series temporales corresponden a la estacionalidad, la tendencia, el componente aleatorio y la autocorrelación (Rey Graña & Ramil Diaz, 2011)

2.1.1.1 Estacionalidad. Una serie temporal es estacional si las observaciones presentes en ella tienen un carácter repetitivo y predecible a lo largo del tiempo (Mills, 2019). Note que en la figura 1 se ilustra una serie temporal estacionaria.

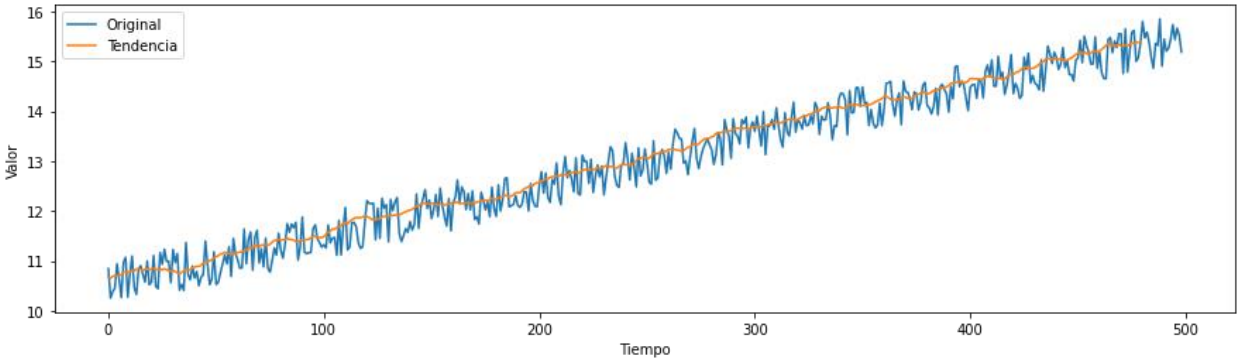
Figura 1. *Serie temporal estacionaria.*



Nota: Serie temporal generada con propósitos ilustrativos.

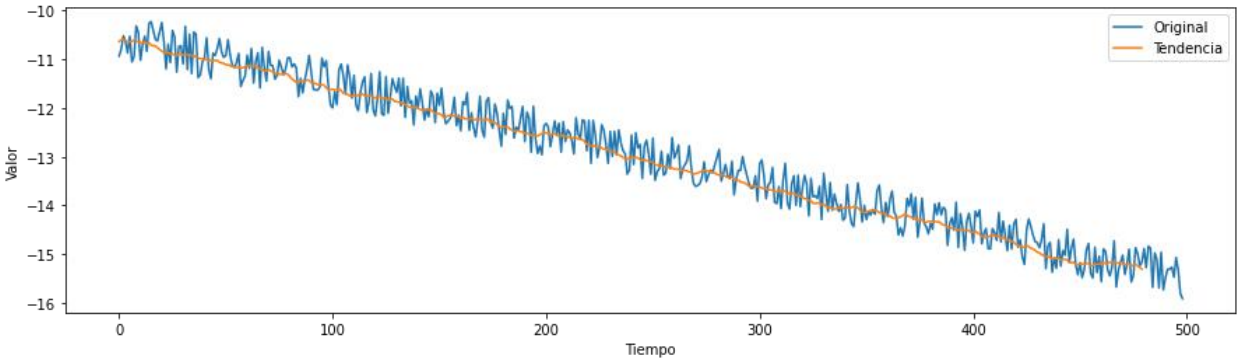
2.1.1.2 Tendencia. Se dice que existe tendencia en una serie temporal cuando el promedio de sus observaciones varía en el tiempo y no es constante a largo plazo, así pues, la tendencia puede ser positiva o negativa, según el comportamiento creciente o decreciente de las observaciones de la serie temporal. En la figura 2 y 3 reflejan este comportamiento.

Figura 2. *Tendencia positiva de serie temporal.*



Nota: Serie temporal con tendencia positiva (en naranja) generada con propósitos ilustrativos.

Figura 3. *Tendencia negativa de serie temporal.*



Nota: Serie temporal con tendencia negativa (en naranja) generada con propósitos ilustrativos.

2.1.1.3 Tendencia determinista. Indica que la tendencia de la serie temporal no es incierta y de fácil suposición en su comportamiento futuro. Generalmente se representa en forma de línea recta

$$T_t = a + bt$$

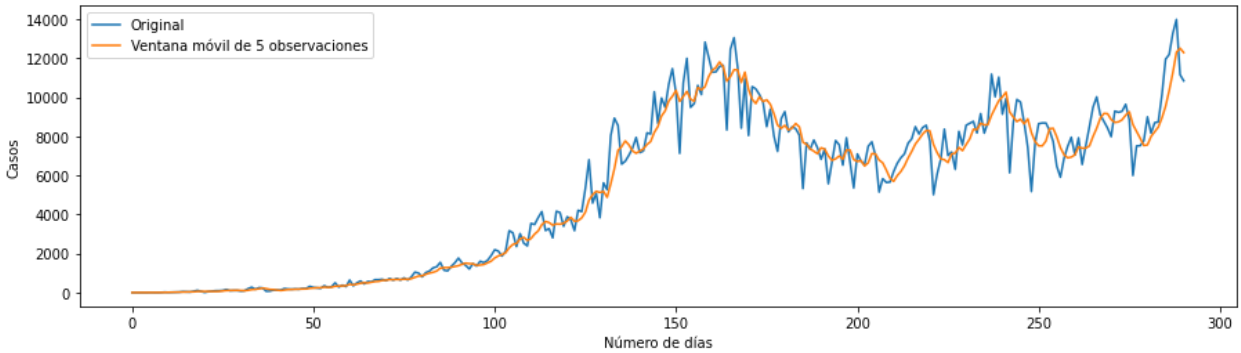
Donde a y b son constantes que se obtienen por medio del método de mínimos cuadrados

2.1.1.4 Tendencia evolutiva. Se considera que la tendencia cambia suavemente a lo largo de la serie temporal. Las medias móviles permiten crear series de tiempo de medias móviles a partir de un subconjunto de observaciones de la serie temporal original. A este subconjunto se le conoce como ventana móvil, e indica la cantidad de observaciones históricas que se consideran para formar un valor medio de dichas observaciones.

$$M_t = \frac{1}{n}\sum_{i=1}^{n} x_{t-i}$$

Donde M_t es la observación generada por el promedio de las n observaciones anteriores al tiempo t. n es el tamaño de la ventana de media móvil y las observaciones x_t son un subconjunto de la serie temporal S. Como se muestra en la *figura 4*, la tendencia evolutiva puede ser positiva, negativa o sin tendencia según la ventana de promedios móviles.

Figura 4. *Tendencia evolutiva*



Nota: Tendencia evolutiva (en naranja). Esta figura fue generada a partir de la cantidad de casos confirmados diarios por COVID-19 en Colombia hasta el mes de diciembre de 2020. Fuente de datos: www.datosabiertos.esri.co

2.1.1.5 Componente aleatorio. Componente que precisa la variabilidad que existe entre una observación u otras, cambios ocurridos por variables o sucesos no conocidos. El modelo de la serie temporal aditiva S_t , se define como la suma de sus componentes.

$$S_t = E_t + T_t + I_t$$

Donde E_t indica la estacionalidad, T_t la tendencia e I_t es el componente aleatorio.

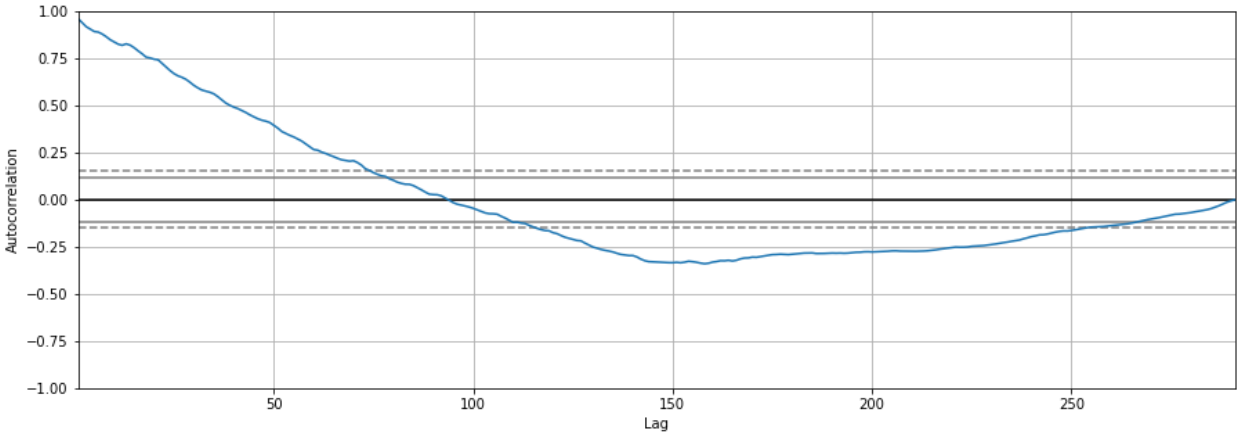
2.1.1.6 Clasificación. Las series temporales pueden clasificarse en dos tipos:

- **Estacionarias:** Mencionado anteriormente en la sección Estacionalidad, estas series temporales son fácilmente predecibles ya que sus observaciones en el tiempo tienen un carácter repetitivo, además la media y la varianza son constantes en el tiempo. Estas series no presentan tendencia positiva o negativa. (Seymour et al., 1997)
- **No estacionarias:** Como su nombre lo indica, son totalmente lo contrario a aquellas series estacionarias, esto implica que las observaciones a lo largo del tiempo no

tienen algún carácter repetitivo o predecible. Sus medias y varianzas no son constantes en el tiempo (Seymour et al., 1997).

2.1.1.7 Autocorrelación. La autocorrelación es la correlación de la misma serie temporal con ella misma desplazada cierta cantidad de observaciones, lo cual permite reconocer patrones o periodicidad implícita de la misma serie temporal a través de observaciones pasadas. (Mills,

Figura 5. *Autocorrelación de serie temporal para múltiples retrasos de tiempo*



Nota. Observe la correlación entre observaciones en la serie temporal y la disminución de esta a mayor cantidad de retrasos temporales. Esta ejemplificación fue creada a partir de los datos de la serie temporal de la cantidad de casos confirmados de COVID-19 en Colombia hasta el mes de diciembre de 2020. Fuente de datos: www.datosabiertos.esri.co

2.1.2 Modelo de bosques aleatorios (Random Forest, RF)

El modelo de RF es un algoritmo *bagging* (potenciado) que consiste en la generación de múltiples árboles de decisión basado en reglas y en el cual se promedia las proyecciones resultantes de los árboles generados. Este algoritmo permite realizar “regresiones” o “proyecciones” con respecto a un conjunto de datos de entrenamiento seleccionados aleatoriamente.

2.1.2.1 Definición. Se considera como bosque aleatorio como aquella compilación de clasificadores con organización de árboles de decisión de la forma $\{h(x, \Theta\_k), k = 1, \dots\}$, donde Θ_k son vectores aleatorios independientes idénticamente distribuidos y cada árbol emite un voto unitario para la clase más popular en el vector de entrada x (Segal, 2004).

2.1.2.2 Convergencia de RF. De un conjunto de clasificadores $h_1(x), h_2(x), \dots, h_k(x)$ y con un conjunto de entrenamiento con la distribución del vector aleatorio Y, X se define como la función margen como:

$$mg(X,Y) = av_k I(h_k(X) = Y) - \max_{j \neq Y} av_k I(h_k(X) = j)$$

Donde $I(.)$ es la función indicadora. El margen mide la medida en que el promedio de votos X, Y para la clase derecha supera el promedio de votos para cualquier otra clase.

$$PE^* = P_{X,Y}(mg(X,Y) < 0)$$

Donde los subíndices X, Y indican que la probabilidad está sobre el espacio X, Y. En un bosque aleatorio $h_k(X, \Theta_k) = h(X, \Theta_k)$ para un gran número de árboles se cumple la ley de los grandes números como: A medida que el número de árboles aumenta, seguramente todas las secuencias $\Theta_1, \dots, PE^*$ converge a (Breiman, 1999):

$$P_{X,Y}(P_\Theta(h(X,\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(X,\Theta) = j) < 0)$$

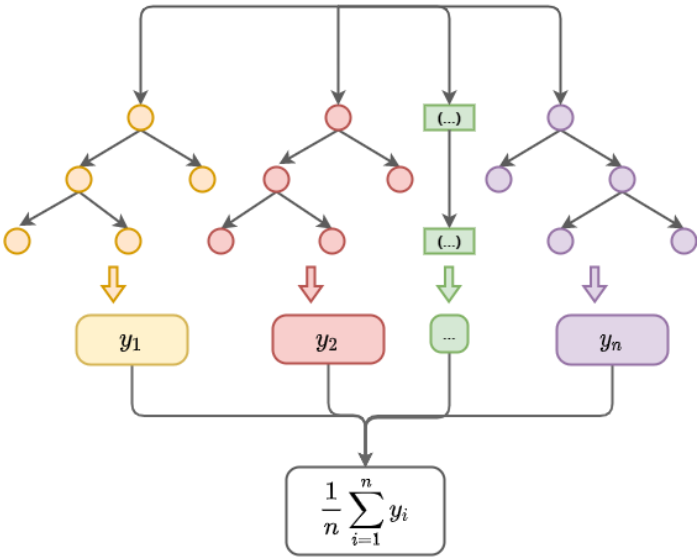
2.1.2.3 RF para la regresión. Son conformados por árboles que crecen dependiendo de un vector aleatorio Θ de tal manera que el predictor de $h(x, \Theta)$ toma valores y las salidas son numéricas, donde se admite que el conjunto de entrenamiento es independiente de la distribución del vector aleatorio Y, X. El error cuadrático medio general para cualquier predictor numérico $h(x)$ es de la forma (Breiman, 1999):

$$E_{X,Y}\big(Y - h(X)\big)^2$$

Se hace un único valor promedio predictor de los k árboles $\{h(x, \Theta_k)\}$ predictores formados. Ahora se define el error cuadrático general como:

$$PE^*(tree) = E_\Theta E_{X,Y}\big(Y - h(X, \Theta)\big)^2$$

Figura 6. *Diagrama Random Forest Regressor*



Nota. Este diagrama es basado en el diagramado propuesto por Mishra.A, Ramteke. S, Sen.P, et al. (2018) [Figura] Recuperado de https://www.springer.com/

RF comúnmente se ilustran como en la *figura 6*, donde se generan múltiples regresiones por parte de árboles de decisión y posteriormente se realiza una media de los resultados para generar una única regresión.

2.1.2.4 Algoritmo implementado. La puesta en marcha del algoritmo de RF está inspirada en la solución propuesta por (Tilgner, 2019), en donde explica detalladamente el proceso de la transformación de los datos, la generación del modelo y las predicciones hechas en el lenguaje de programación R. Cabe mencionar, que para la implementación en este proyecto se agregaron procesos y rutinas lógicas para otorgarle flexibilidad a la ejecución de los modelos, al igual que existen notables diferencias con respecto a la generación de los conjuntos de datos, metodología en fase de entrenamiento y prueba y transformación en el dominio de las observaciones predichas. Es importante resaltar que se definió el modelo con los siguientes hiperparámetros: 250 estimadores, criterio de regresión de MSE y profundidad máxima de los árboles de 150.

2.1.3 Vectores de soporte regresivo (Support Vector Regressor, SVR)

Las máquinas de soporte vectorial (*Support Vector Machine, SVM*) y los vectores de soporte regresivo (*Support Vector Regressor, SVR*) son modelos ampliamente usados en problemas de clasificación y regresión, caracterizados por su versatilidad y adaptación a múltiples espacios dimensionales.

2.1.3.1 Definición. Los SVR tienen la capacidad de realizar regresiones sobre un conjunto de datos de prueba de la forma $\{(x_1, z_1), \dots, (x_l, z_l)\}$, donde $x_i \in R^n$ es un vector característica y $z_i \epsilon R^1$ es la etiqueta de salida. Bajo los parámetros $C > 0$ y $\varepsilon > 0$, el modelo de un vector de soporte regresivo es (Chang & Lin, 2011) :

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i + C\sum_{i=1}^{l}\xi_i^*$$

El primer término de la ecuación anterior indica el problema de maximización al que se somete el vector o hiperplano w para obtener el máximo margen que abarque la mayoría de los

datos. Por otro lado, los términos restantes se someten a un problema de minimización. Allí la constante C mantiene un umbral de equilibrio que describe la tolerancia a desviaciones mayores a los márgenes de soporte, así como ξ y ξ^* que indican los errores de los datos que están por fuera de los márgenes de soporte

$$w^T \phi(x_i) + b - z_i \leq \epsilon + \xi_i$$

$$z_i - w^T \phi(x_i) - b \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0, i = 1, \dots, l$$

El problema doble es:

$$min_{\alpha,\alpha^*} \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \varepsilon e^T(\alpha + \alpha^*) - y^T(\alpha - \alpha^*)$$

Sujeto a:

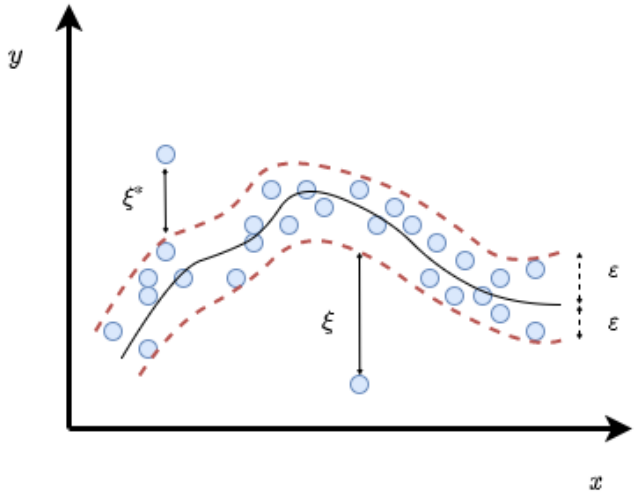
$$e^T(\alpha - \alpha^*) = 0$$

$$0 \leq \alpha_i, \alpha^* \leq C, 1 = 1, \dots, n$$

Donde e es un vector de unos, Q es una matriz semi-definida positiva de n por n. $Q_{ij} \equiv K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ es un kernel. La regresión creada es de la forma (Chang & Lin, 2011):

$$\sum_{i=1}^{l} (\alpha_i - \alpha_i^*) K(x_i, x) + b$$

Figura 7. *Vector de soporte regresivo*



Nota. Figura basada a partir del diagrama creado por Awa. M y Khanna. R (1999). Efficient Learning Machines Theories, concepts, and applications for engineers and system designers [Figura]. Recuperado de https://www.researchgate.net/

2.1.3.2 Algoritmo implementado. Al igual que el modelo RF, el modelo de SVR también está inspirado en la solución propuesta por (Tilgner, 2019), sin embargo, también presenta algunas modificaciones y adaptaciones, por ejemplo, el cambio del dominio de los datos, el retorno de las predicciones y la especificación de los hiperparámetros como lo son: el *kernel* de tipo RBF y *gamma* de tipo escala.

2.1.4 Modelo autorregresivo integrado de medias móviles (ARIMA).

El modelo ARIMA es un modelo estadístico para la predicción de series temporales, de uso frecuente por su versatilidad y adaptabilidad, ya que se basa en componentes autorregresivos y de medias móviles para generar una predicción.

2.1.4.1 Definición. Una predicción x_t es formada por medio de observaciones históricas $\{x_{t-1}, x_{t-2}, \dots, x_{t-p}\}$, donde p es el indicativo de la cantidad de pasos históricos tenidos en cuenta. El operador autorregresivo de orden p, comúnmente llamado como $AR(p)$ tiene la forma (Zhang, 2020) :

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t$$

Donde x_t es estacionario, w_t y $\phi_1, \phi_2, \dots, \phi_p$ son constantes diferentes de cero. La ecuación anterior representa una observación x_t de la serie temporal univariante a través de las observaciones pasadas junto con las constantes $\phi_1, \phi_2, \dots, \phi_p$ y w_t. Cabe resaltar que si la media de x_t no es cero se reemplaza x_t por $x_t - \mu$, quedando de la siguiente forma:

$$x_t - \mu = \phi_1(x_{t-1} - \mu) + \phi_2(x_{t-2} - \mu) + \cdots + \phi_p(x_{t-p} - \mu) + w_t$$

Reduciendo como:

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t$$

$$\alpha = \mu(1 - \phi_1 - \cdots - \phi_p)$$

En adición, el modelo ARIMA contiene el modelo de promedios móviles de orden q, simplificado como $MA(q)$.

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

Donde w_t y $\theta_1, \theta_2, \dots, \theta_q$ son parámetros diferentes de cero. Este modelo es análogo al modelo autorregresivo. A partir de las anteriores ecuaciones se construye el modelo autorregresivo de medias móviles (*ARMA*) de la siguiente manera:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

Para $ARIMA(p, d, q)$, los parámetros p, d y q son valores positivos mayores a cero; donde p representa el orden autorregresivo, es decir, la cantidad de observaciones históricas que son

consideradas para hacer una predicción. d es el orden integrado que involucra la cantidad de veces de diferencia que se aplica a la serie temporal hasta que esta sea estacionaria. El parámetro q indica el factor de media móvil aplicado al modelo ARIMA.

2.1.4.2 Algoritmo implementado. En esta ocasión la metodología en la creación del modelo ARIMA discrepa de los modelos antes vistos. Para (Brownlee, 2020) la metodología para hacer predicciones usando el modelo ARIMA corresponde a Walk Forward Validation (WFV), en donde el modelo se entrena con el conjunto de entrenamiento y, agrega por cada iteración, la observación del conjunto de prueba al conjunto de entrenamiento después de hacer una predicción. En consecuencia, este modelo se entrena de forma recurrente hasta finalizar con el conjunto de prueba. En la sección de anexos se presentan los resultados para ARIMA con y sin esta metodología.

2.1.5 Memoria de largo y corto plazo (LSTM).

Las redes neuronales recurrentes suponen un avance significativo en los algoritmos de aprendizaje automático dado que éstas incluyen funcionalidades robustas para generar una salida dependiendo de características o patrones implícitos en los datos.

2.1.5.1 Definición. La variante de las redes recurrentes, conocidas como *Long Short Term-Memory (LSTM),* opera con celdas de memoria y compuertas para la adición o sustracción de elementos dependiendo de las activaciones en las compuertas de memoria. Dentro de las celdas de memoria se agrega una compuerta multiplicativa de entrada, esto para proteger el estado de memoria almacenado en la celda j de perturbaciones o entradas irrelevantes. A su vez también se agrega una compuerta multiplicativa de salida, esto para proteger las otras unidades de perturbaciones o irrelevancias en el estado de memoria en la celda j (Hochreiter & Schmidhuber, 1997).

$$y^{in_j}(t) = f_{in_j}\left(net_{in_j}(t)\right)$$

$$y^{out_j}(t) = f_{out_j}\left(net_{out_j}(t)\right)$$

La entrada $y^{in_j}(t)$ se transforma por la unidad multiplicativa en la celda j por $f_{in_j}\left(net_{in_j}(t)\right)$ y la salida $y^{out_j}(t)$ por $f_{out_j}\left(net_{out_j}(t)\right)$. Donde:

$$net_{in_j}(t) = \sum_u w_{in_j u} y^u(t-1)$$

$$net_{out_j}(t) = \sum_u w_{out_j u} y^u(t-1)$$

Se generaliza como:

$$net_{c_j}(t) = \sum_u w_{c_j u} y^u(t-1)$$

La metodología basada en “cintas transportadoras” y las compuertas multiplicativas de entrada y salida permiten que a lo largo de más de una celda de memoria se añada o se remueva información según su relevancia, por ello se considera a este tipo de redes artificiales con carácter

de memoria de corto y largo plazo puesto que tienen la capacidad de “recordar” los patrones o características de un conjunto de datos ingresados secuencialmente.

$$y^{c_j}(t) = y^{out_j}(t)h(s_{c_j}, (t))$$

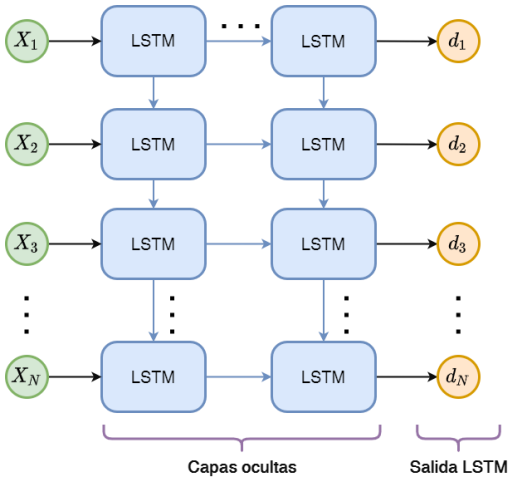
El estado interno de la celda j:

$$s_{c_j}(0) = 0$$

$$s_{c_j}(t) = s_{c_j}(t-1) + y^{in_j}(t)g\left(net_{c_j}(t)\right);\ \ t > 0$$

Como se observa, las celdas *LSTM* tiene la capacidad para agregar o remover información que considere pertinente o irrelevante según las compuertas de activación. De igual manera, las celdas de memoria reciben entradas procesadas por parte de celdas posteriores (note la *figura 8)*, refinando aún más la salida de los datos.

Figura 8. *Arquitectura de red recurrente LSTM*



Nota. Figura basada en la imagen presentada por Hu. J, Zhang.Y et al. (2020). Time Series Prediction Method Based on Variant LSTM Recurrent Neural Network [Figura]. Recuperado de https://springer.com/

2.1.5.2 Algoritmo implementado. Para el modelo LSTM se toma como referencia la implementación propuesta por (Brownlee, 2018), gracias a que detalla la metodología para realizar proyecciones en series temporales univariantes en este tipo de redes. Al igual que los anteriores algoritmos, se tomó de referencia ciertas funcionalidades indispensables como lo es la definición y estructura del conjunto de datos, creación del modelo LSTM a través de capas, cambio de dominio de datos, funciones de activación y declaración de capas ocultas densas. Igualmente, las celdas LSTM poseen bidireccionalidad, eso indica que el flujo de información se presenta en ambas direcciones. Se asignó una capa de 64 celdas de memoria bidireccionales LSTM ya que, durante la fase de experimentación, no se evidenciaba mejora en las métricas de error al aumentar este número. Igualmente sucede con el número de épocas, en donde se declara en 64 épocas. En el apartado de los hiperparámetros se asigna un *dropout* de 0.4 a la primera capa LSTM, además de usar *MSE* como función de pérdida y el optimizador de *Adam*.

2.1.6 Métricas de error

Los algoritmos deben ser evaluados según su capacidad de predicción de observaciones de una serie temporal, por esto se acude a métricas de error que permitan diferenciar cuantitativamente el rendimiento de las predicciones. En la literatura se encuentran varias métricas las cuales representan, en diferentes cuantías, el error en las regresiones.

- **Error cuadrático medio (MSE)**

Este error mide el promedio de los errores al cuadrado de un conjunto de observaciones predichas $\hat{y}$ y observaciones reales y (Pedregosa et al., 2011).

$$MSE(y, \hat{y}) = \frac{1}{n_{ejemplos}} \sum_{i=1}^{n_{ejemplos}} (y_i - \hat{y}_i)^2$$

- **Error absoluto medio (MAE)**

Mide el error absoluto medio que existe entre las observaciones reales y y las observaciones predichas $\hat{y}$ *(Pedregosa et al., 2011).*

$$MAE(y, \hat{y}) = \frac{1}{n_{ejemplos}} \sum_{i=1}^{n_{ejemplos}} |y_i - \widehat{y_i}|$$

- **Error de porcentaje absoluto medio (MAPE)**

Mide en forma porcentual el error absoluto de y y de $\hat{y}$, en donde un error cercano a 1.0 se considera un modelo pobre en predicciones (Cao et al., 2015; Pedregosa et al., 2011)

$$MAPE\ (y, \hat{y}) = \frac{\sum_{i=1}^{n\_ejemplos} \frac{|y - \hat{y}|}{|\widehat{y}|}}{n_{ejemplos}}$$

- **Error absoluto mediano (MedianAE)**

A diferencia del MAE, el error absoluto medio se basa tomando las medianas de todas las diferencias absolutas entre y y $\hat{y}$ (Pedregosa et al., 2011).

$$MedAE(y, \hat{y}) = mediana(|y_1 - \widehat{y_1}|, \dots, \left|y_{n_{ejemplos}} - \hat{y}_{n_{ejemplos}}\right|)$$

- **Error máximo (MaxError)**

Genera el mayor error presente en el conjunto de predicciones $\hat{y}$ con respecto a y. Un error máximo cercano a 0.0 implica que el modelo predictor se ajusta muy bien a los datos de prueba (Pedregosa et al., 2011).

$$MaxError(y, \hat{y}) = \max\ (|y_{n_{ejemplos}} - \hat{y}_{n_{ejemplos}}|)$$

- **R cuadrado (r^2)**

Mide el coeficiente de determinación de una regresión. Un coeficiente de determinación cercano a 1.0 implica una aproximación acertada de las observaciones predichas a las observaciones reales (Pedregosa et al., 2011).

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n\_ejemplos}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n\_ejemplos}(y_i - \bar{y}_i)^2}$$

Donde $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ y $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \epsilon_i^2$

- **Indicador de fuerza de la señal recibida (RSSI)**

El RSSI indica la medición de la potencia en la banda de frecuencias de interés a un receptor. Si existe una transmisión activa en el tiempo de muestreo, entonces el RSSI mide la potencia de la señal recibida. En caso opuesto, mide la potencia de la interferencia más el ruido de fondo. La lectura de este indicador puede ser obtenida directamente desde el chip de radio en un dispositivo receptor. Pero esta lectura contiene la potencia de la señal como el ruido. Este indicador no tiene unidades de medida, sin embargo, puede ser traducida a decibelios (*dB*) (Yuan et al., 2017).

- **Indicador de la calidad del enlace (LQI)**

Se define como la intensidad y/o calidad de un paquete recibido y a este paquete se le asigna un número entero entre 0 y 255. Los paquetes que tengan valores de asignación altos detallan una mejor comunicación entre nodos (Yuan et al., 2017).

2.2 Metodología

Es esta sección se especifica la metodología en el pretratamiento y tratamiento de los datos, la metodología general del proyecto, las metodologías de los modelos y los resultados con sus respectivas métricas de error y tiempos de ejecución.

Durante la realización de este proyecto investigativo se marcaron pautas para mejorar el flujo de aprendizaje, implementación y experimentación con los modelos propuestos. En adición, se realizaron procesos de automatización en la carga, lectura, pretratamiento y ejecución de los algoritmos, esto con el fin de optimizar recursos computacionales, tiempo y trabajo humano. Cabe mencionar que en dichos procesos se realizó la documentación correspondiente, con el objeto de sustentar metodológicamente el trabajo investigativo.

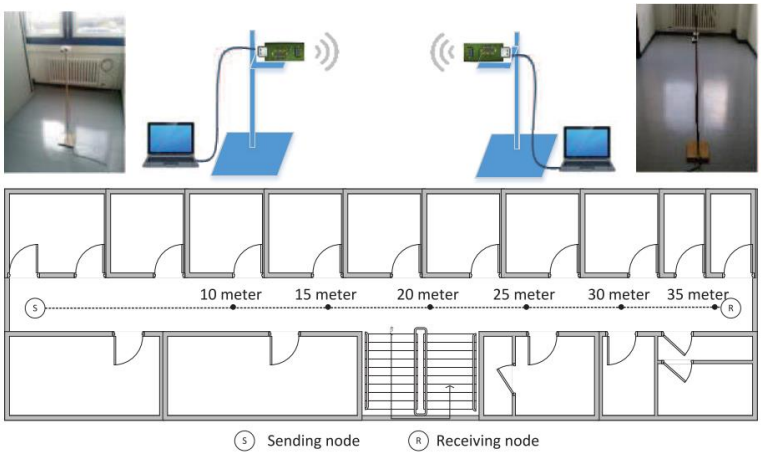
2.2.1 Conjuntos de datos

En este aspecto se efectuó la búsqueda de datos aptos en plataformas o repositorios de uso público o con acceso autorizado, esta búsqueda requirió de filtros concretos para segmentar aquellos datos que contenían las métricas de la calidad de enlace idóneas. Durante esta etapa se adquirieron los ficheros de datos desde la plataforma *Crawdad* otorgados por (Fu & Zhang, 2015) del conjunto de trazas *packet-metadata,* nombrados de la siguiente manera: *distance_10m, distance_15m, distance_20m, distance_25m, distance_30m* y *distance_35m,* en donde por cada fichero había doce archivos en formato *txt* nombrados *raw_data_run1.txt* hasta *raw_data_run12.txt.*

2.2.1.1 Estructura. Cada archivo presenta 8064 registros, cada uno de estos registros contienen información de la entrega para 300 paquetes transmitidos con la misma configuración de parámetros. Estos parámetros están ordenados de la siguiente forma: Experiment_number, Run_Number, Period, Packet_number, Packet_length, Queue_size, Max_tries, Retry_delay, TxP_level, Distance, Packet_sequence_number_00#, Buffer_overflow, Actual_queue_size, ACK, Actual_try_number, RSSI, Noise_floor, LQI y Arrival_time. En este orden y en columnas se encuentran los datos de los 300 paquetes transmitidos, sólo se usaron aquellos parámetros que medían la calidad del enlace, en este contexto, LQI y RSSI.

2.2.1.2 Topología de red. (Fu et al., 2015) detallan en su escrito la disposición de dos nodos *TelosBk*, equipados con *TI CC2420* (chip simple que opera bajo el estándar IEEE 802.15.4 a una frecuencia de 2.4 GHz) en un pasillo de 2 metros de ancho por 40 metros de largo. Los autores mantuvieron la línea de visión entre los dos nodos, permitiendo que personas circularan por el pasillo sin afectar los experimentos. Realizaron los experimentos a diferentes distancias entre los nodos, como se dispone en la *figura 9*.

Figura 9. *Topología de red*

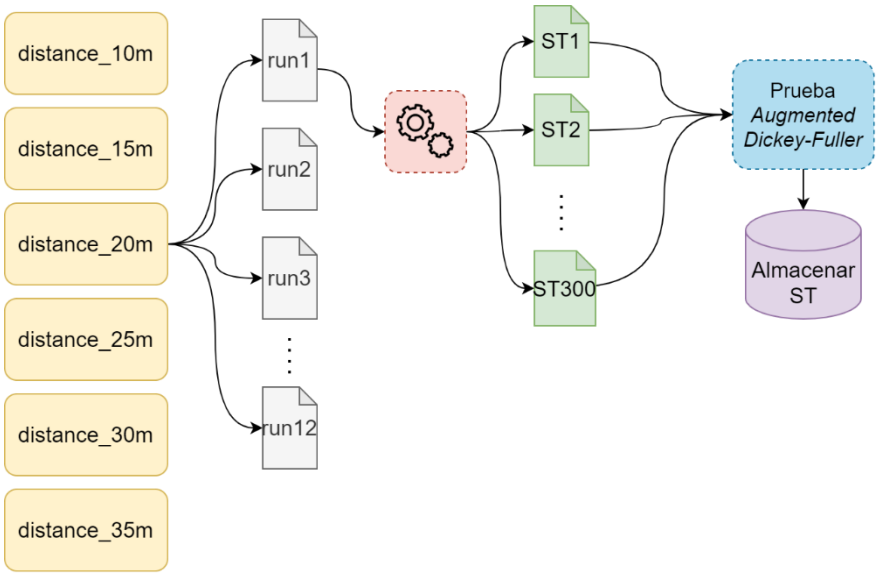


Nota. Imagen otorgada por Fu. S, Zhang.Y et al. Experimental Study for Multi-layer Parameter Configuration of WSN Links [Figura]. Recuperado de www.researchgate.net.

2.2.1.3 Pretratamiento. Los datos se sometieron a un proceso de pretratamiento que consiste en reemplazar los registros en ‘cero’ por ‘menos uno’ para la métrica RSSI y por ‘uno’ para LQI, con la finalidad de no provocar indeterminaciones en el cálculo de MAPE.

2.2.1.4 Obtención de series temporales. Por cada archivo de texto se ejecutó una rutina en *Python* en donde se automatizó el proceso de extracción de los 300 registros de los metadatos de los paquetes transmitidos, y se transformaron en series temporales univariantes. Posteriormente, se efectuó la prueba *Augmented Dickey-Fuller* (Seabold & Perktold, 2010) para determinar si cada serie de tiempo analizada cumplía con la no estacionalidad. Aquellas series temporales que lograban superar la prueba fueron almacenadas en formato *csv*, mientras las que no, eran descartadas automáticamente.

Figura 10. *Diagrama de automatización y extracción de series temporales*



Nota: El ícono de engranaje representa la rutina creada en Python para automatizar el proceso de extracción y transformación de las series temporales.

Una vez ejecutado el proceso de automatización, la distribución de las series temporales resultó como se muestra en las tablas 1 y 2. Cada archivo guardado tiene el nombre como *raw_data_runX__Y.csv*, donde '*X'* representa el número del archivo original y '*Y'* el número de la columna donde se extrajeron y se transformaron a serie temporal univariante.

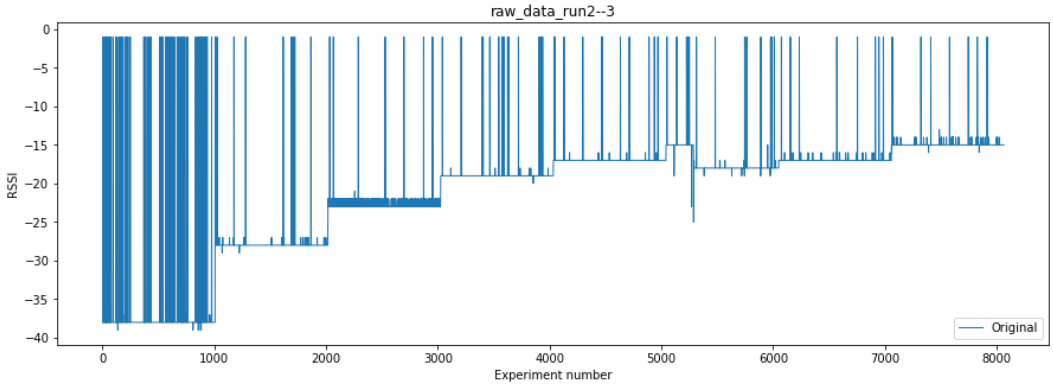
Tabla 1. *Número de series temporales resultantes para LQI.*

Nombre fichero	Número de series temporales
Distance_10m	1
Distance_15m	1
Distance_20m	11
Distance_25m	0
Distance_30m	0
Distance_35m	0

Tabla 2. *Número de series temporales resultantes para RSSI*

Nombre fichero	Número de series temporales
Distance_10m	2
Distance_15m	1
Distance_20m	10
Distance_25m	0
Distance_30m	0
Distance_35m	0

Figura 11. *Serie de tiempo filtrada*



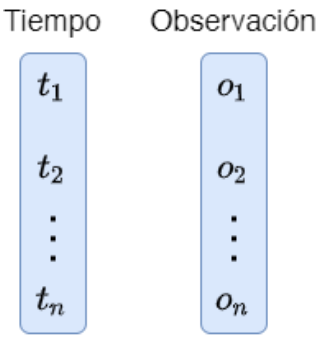
Nota: El eje horizontal 'Experiment Number' hace referencia al registro secuencial de los paquetes transmitidos. Este ejemplo de serie de tiempo corresponde a la obtenida del fichero distance_20m para le métrica RSSI

2.2.2 Tratamiento de los datos

Como se mencionó en la sección de series temporales, éstas son de gran interés por su capacidad de representar eventos repetitivos o definir tendencias en observaciones futuras, que pueden ser proyectadas a través de los algoritmos de aprendizaje automático ya vistos. Estos algoritmos deben alimentarse con conjuntos de datos con estructuras aptas, en consecuencia, se deben aplicar ciertas transformaciones a los datos antes de proceder con su ejecución.

2.2.2.1 Problema supervisado. Una serie temporal univariante es tiene la forma como se ilustra en la *figura 12.*

Figura 12. *Serie temporal univariante*



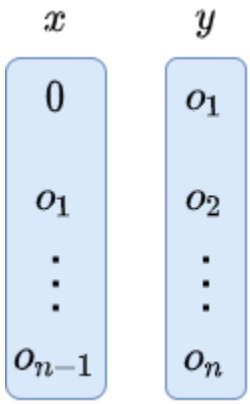
Así pues, las series temporales tienen una relación biyectiva entre la variable independiente temporal t y las observaciones medidas o. Además, se dice que es univariante porque sólo se representa la serie temporal con una sola característica de medición.

2.2.2.2 Un retraso de tiempo. Para esta forma de datos es necesario proceder con una transformación para obtener un problema supervisado, es decir, para un registro etiqueta y sea éste representado a través de la variable o variables característica $x_1, x_2, \dots, x_n$. En estos términos se construye un conjunto de datos donde las características x sean las observaciones y en un retraso de tiempo.

$$x_t = y_{t-1}$$

El conjunto de datos para un problema supervisado con un retraso temporal se ilustra en la *figura 13.*

Figura 13. *Problema supervisado usando un retraso de tiempo*

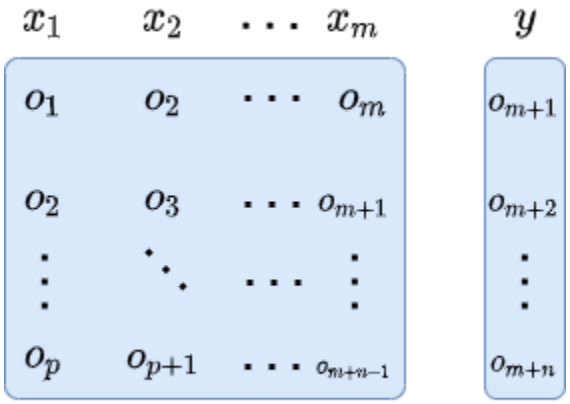


Nota: El vector característica x *es definido por la observación de* y *en un tiempo t-1*

Esta transformación en la estructura de los datos permite que una observación descrita en el tiempo t pueda ser definida a través de sus observaciones pasadas. Es así como se introduce el concepto de múltiples retrasos de tiempo.

2.2.2.3 Múltiples retrasos de tiempo. Es posible representar un registro y con más de un retraso temporal. Esta transformación es ejecutada para incluir más información temporal, en lugar de usar una sola observación, se usan m observaciones retrasadas en el tiempo para describir la variable etiqueta y en un instante de tiempo t.

Figura 14. *Problema supervisado usando múltiples retrasos de tiempo*



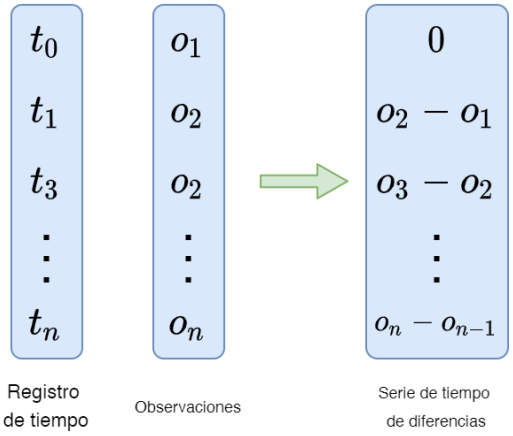
Nota: Observe que las observaciones y *son descritas por m observaciones anteriores en el tiempo.*

Estas transformaciones solamente son aplicables para las estructuras de datos de entrada de RF, SVR y LSTM (para una sola proyección).

2.2.2.4 Serie de tiempo de diferencias. Las series temporales no estacionarias suponen un problema a la hora de intentar crear proyecciones con los algoritmos de aprendizaje de máquina, puesto que las observaciones en el tiempo no poseen un carácter repetitivo o predecible. Como detallan (Hofmann & Dinges, 2006), se puede remover la tendencia de una serie temporal univariante al aplicar un filtro lineal a la misma. Este filtro considera que la tendencia polinomial que posea la serie temporal se reduce al aplicar la diferencia en un mismo instante de tiempo t de una observación y_t y y_{t-1}.

$$y_t = y_t - y_{t-1}$$

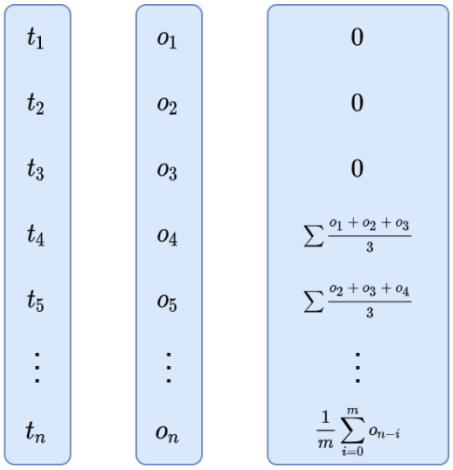
Figura 15. *Serie de tiempo de diferencias*



Cuando se aplica este filtro y, la posterior generación de predicciones por parte de los modelos, se debe ejecutar el proceso inverso, esto con el fin de retornar al dominio original en donde se encontraban las observaciones originales.

2.2.2.5 Serie de tiempo de promedios móviles. La serie de tiempo de promedios móviles consiste en realizar promedios en las observaciones con respecto a una ventana móvil que recorre la totalidad de la serie temporal. La ventana móvil es un actor estabilizador de la serie temporal logrando suavizar la media y la varianza a lo largo de la serie.

Figura 16. *Serie de tiempo de promedios móviles*



Nota. Para este ejemplo se aplica una ventana de 3 observaciones para la serie de tiempo de promedios móviles

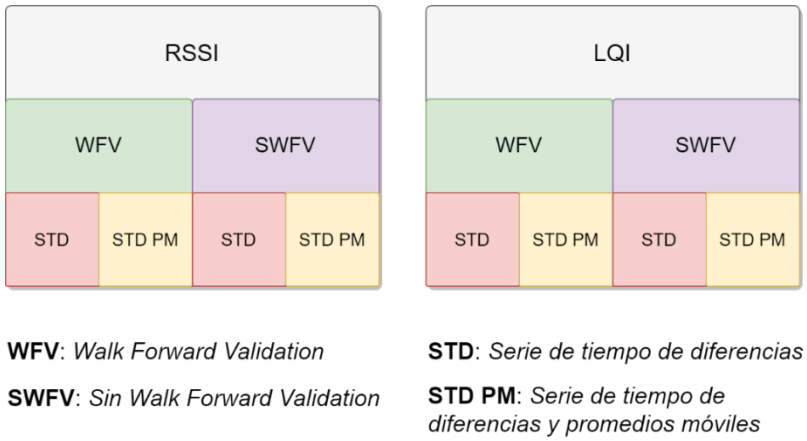
2.2.3 Experimentación

Luego de obtener las series temporales que cumplieron con la prueba de hipótesis, se recurre ahora a la experimentación con los datos. Para ambas métricas de calidad de enlace, RSSI y LQI, se dispusieron las celdas de ejecución para cada uno de los ficheros filtrados, *distance_10m, distance_15m* y *distance_20m.* La manera en cómo se dispusieron los experimentos se lista a continuación.

- **Series de tiempo de diferencias y promedios móviles con *WFV***
- **Series de tiempo de diferencias sin *WFV***
- **Series de tiempo de diferencias sin *WFV***
- **Series de tiempo de diferencias y promedios móviles sin *WFV***

Existen trece archivos de series temporales por cada métrica de calidad de enlace, como lo disponen las *tablas 1* y *2*, por este motivo por cada métrica existen 54 experimentos según la disposición de la lista anterior. Esto implica un total de 104 experimentos para ambas métricas RSSI y LQI. Observe la estructura de la metodología en la *figura 17.*

Figura 17. *Metodología de los experimentos*



Nota: Metodología de experimentos para métricas RSSI y LQI.

En la implementación se definieron funciones que automatizaban el proceso de creación, preparación y ejecución de componentes para la puesta en marcha de los modelos. En la figura 18 se observa la automatización en la ejecución para cinco, diez y veinte pasos temporales, además del llamado de la función *'ejecutarModelos',* la cual prepara los datos y ejecuta los modelos retornando los errores y tiempos de ejecución.

Figura 18. *Función automatizar*

```
def automatizar(dir, metrica, D, PM):
  l = [5, 10, 20]

  path = dir
  indicador = metrica
  numNeuronas = 64
  epocas = 64
  porcentajePrueba = 0.4

  for i in l:


    RF, SVR, ARIMA, LSTM, timeRF, timeSVR, timeARIMA, timeLSTM = ejecutarModelos(D, PM, path, indicador, numNeuronas, epocas, porcentajePrueba, i)
    print ("*********************************************************************************")

    print(RF )
    print(SVR )
    print(ARIMA )
    print(LSTM )

    print ("---------------------------------------------------------------------------------")

    print(timeRF)
    print(timeSVR)
    print(timeARIMA)
    print(timeLSTM)

    print ("*********************************************************************************")
```

2.2.3.1 Experimentos para RSSI. Se dio comienzo con los datos de RSSI, en donde se prepararon todos los componentes para ejecutar los modelos. Esto quiere decir que se ejecutaron las celdas según la lista de disposición de experimentos antes mencionada. En la implementación hecha en el cuaderno de *Jupyter* y *Google Colaboratory*, se ejecutan de forma secuencial los modelos según las métricas especificadas. La función '*automatizar'* mostrada en la *figura 19*, especifica la recepción de los datos a través de la dirección del archivo, la métrica de calidad de enlace que corresponde a los datos de ese archivo, si los datos se deben tratar como serie de tiempo de diferencias y, por último, si se deben tratar los datos como serie de tiempo de promedios móviles. Además, se puede apreciar que la ejecución de esta celda conduce a la evaluación de la serie temporal con la prueba *Augmented Dickey-Fuller*, mostrando el valor *p* asociado a esta prueba. En adición, se muestran los errores de los modelos para los pasos temporales considerados en la función '*automatizar'.* Posteriormente la función retornará la prueba de autocorrelación, la visualización de las predicciones y la correlación de las predicciones versus los valores reales. Note las *figuras 20* y *21*.

Figura 19. *Retorno de resultados de modelos RSSI*

```
automatizar("/content/drive/MyDrive/PROYECTO -- EXPERIMENTOS/Script limpiar datasets/Dataset limpios RSSI/10m/raw_data_run3--1.csv",
            "RSSI", True, False)
```

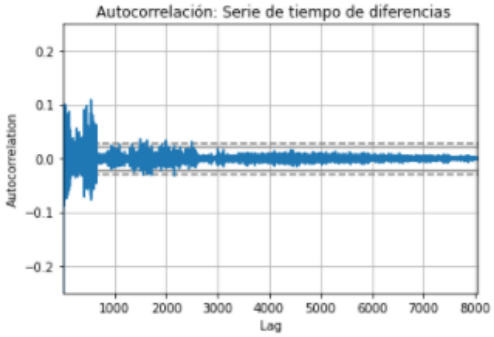
```
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d array
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d
  y = column_or_1d(y, warn=True)
*********************************************************************
{'Errores modelo RF : 5': (0.979, 0.64, 19.091, 0.411, 13.364, 0.679)}
{'Errores modelo SVR : 5': (0.607, 0.604, 18.869, 0.393, 3.97, 0.801)}
{'Errores modelo ARIMA : 5': (0.382, 0.444, 3.781, 0.354, 3.0, 0.875)}
{'Errores modelo LSTM : 5': (0.237, 0.428, 18.147, 0.321, 1.419, 0.922)}
---------------------------------------------------------------------------
{'Tiempo modelo RF : 5': 1.6655559539794922}
{'Tiempo modelo SVR :  5': 0.20197129249572754}
{'Tiempo modelo ARIMA : 5': 4055.962601661682}
{'Tiempo modelo LSTM : 5': 684.292962551117}
*********************************************************************
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_matplotlib/misc.py:411: UserWarning: Requested projection is different from c
  ax = plt.gca(xlim=(1, n), ylim=(-1.0, 1.0))
----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d array
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d
  y = column_or_1d(y, warn=True)
*********************************************************************
{'Errores modelo RF : 10': (2.421, 0.749, 19.938, 0.424, 20.209, 0.207)}
{'Errores modelo SVR : 10': (0.626, 0.595, 18.603, 0.417, 3.943, 0.795)}
{}
{'Errores modelo LSTM : 10': (1.018, 0.966, 18.929, 0.845, 2.304, 0.667)}
---------------------------------------------------------------------------
{'Tiempo modelo RF : 10': 3.559488534927368}
{'Tiempo modelo SVR :  10': 0.24404239654541016}
{}
{'Tiempo modelo LSTM : 10': 1059.7684342861176}
*********************************************************************
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
```

Nota: Observe el retorno de los errores para cada uno de los modelos que genera la implementación.

Figura 20. *Retorno de prueba de autocorrelación. RSSI*

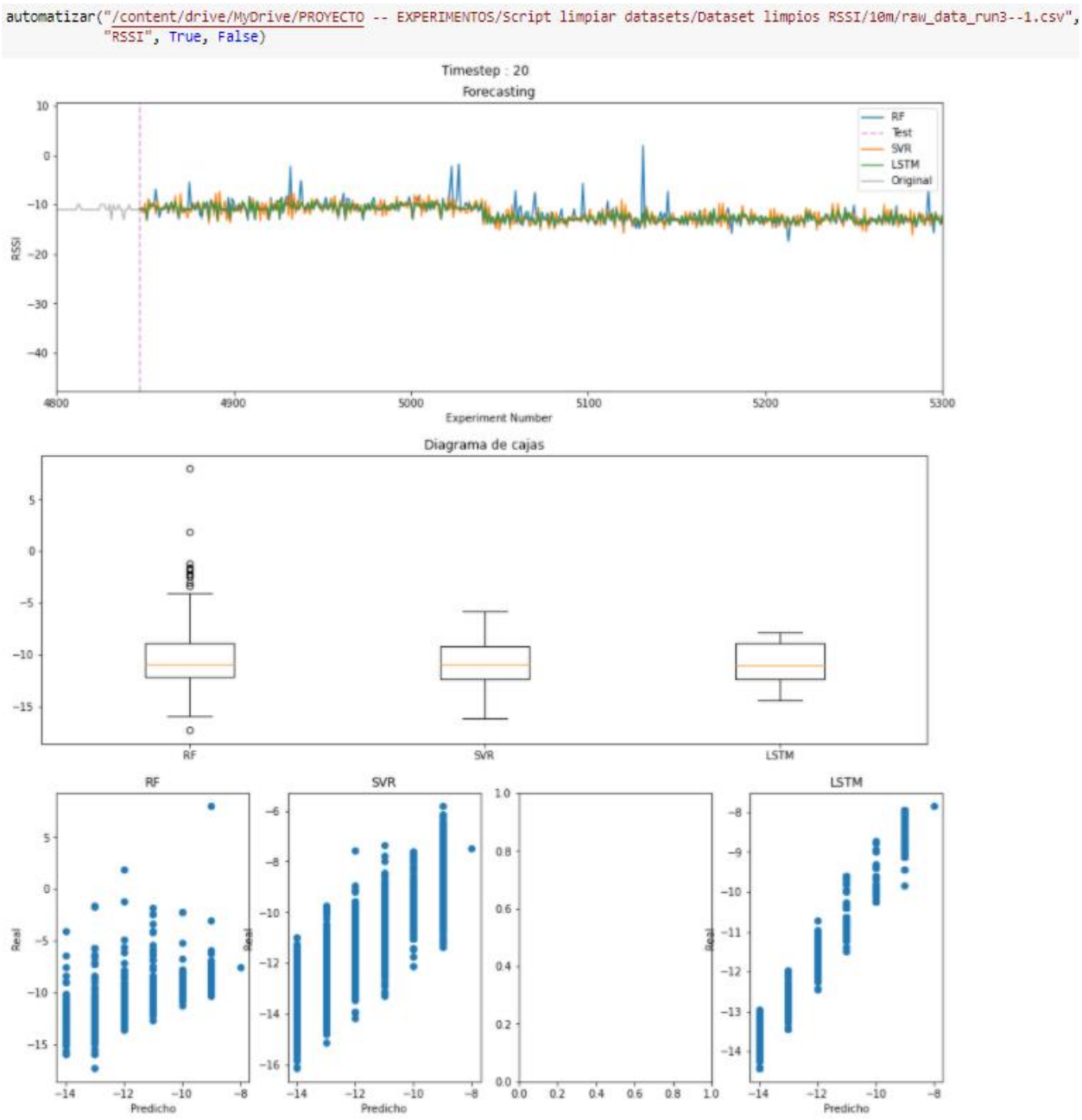
```
automatizar("/content/drive/MyDrive/PROYECTO -- EXPERIMENTOS/Script limpiar datasets/Dataset limpios RSSI/10m/raw_data_run3--1.csv",
            "RSSI", True, False)
```

```
----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d array
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d
  y = column_or_1d(y, warn=True)
**************************************************************************
{'Errores modelo RF : 20': (1.365, 0.65, 19.201, 0.425, 16.982, 0.553)}
{'Errores modelo SVR : 20': (0.832, 0.699, 19.056, 0.53, 4.434, 0.728)}
{}
{'Errores modelo LSTM : 20': (0.134, 0.266, 18.016, 0.148, 1.405, 0.956)}
--------------------------------------------------------------------------
{'Tiempo modelo RF : 20': 7.346376657485962}
{'Tiempo modelo SVR :  20': 0.31047844886779785}
{}
{'Tiempo modelo LSTM : 20': 1826.157473564148}
**************************************************************************
```



Nota: Observe la prueba de autocorrelación cuando la serie temporal es analizada por primera vez y luego cuando se aplica serie de tiempo de diferencias.

Figura 21. *Retorno de visualización, diagrama de cajas y correlación de las predicciones.*



Nota: Las visualización de las predicciones se realizan sobre la sección de prueba de la serie temporal. El eje vertical del diagrama de cajas corresponde a la medida de valores en RSSI.

2.2.3.2 Experimentos para LQI. Después de haber realizado los experimentos para RSSI, se inicia el proceso para LQI. Se aplica la misma disposición de los experimentos que se mencionaron anteriormente y se ejecuta el mismo proceso de extracción de resultados en archivos pertinentes.

Figura 22. *Retorno de resultados de modelos LQI.*

```
automatizar("/content/drive/MyDrive/PROYECTO -- EXPERIMENTOS/Script limpiar datasets/Datset Limpios LQI/10m/raw_data_run3--1.csv",
            "LQI",True, False)
```

```
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
-----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d ar
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a
  y = column_or_1d(y, warn=True)
**********************************************************************
{'Errores modelo RF : 5': (1.178, 0.729, 18.415, 0.465, 15.359, 0.614)}
{'Errores modelo SVR : 5': (0.831, 0.704, 18.284, 0.457, 3.97, 0.728)}
{'Errores modelo ARIMA : 5': (0.382, 0.444, 3.781, 0.354, 3.0, 0.875)}
{'Errores modelo LSTM : 5': (1.683, 1.129, 18.873, 0.988, 4.017, 0.448)}
----------------------------------------------------------------------
{'Tiempo modelo RF : 5': 1.6646480560302734}
{'Tiempo modelo SVR :  5': 0.16831326484680176}
{'Tiempo modelo ARIMA : 5': 4089.862053871155}
{'Tiempo modelo LSTM : 5': 693.2197227478027}
**********************************************************************
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_matplotlib/misc.py:411: UserWarning: Requested projection is different fro
  ax = plt.gca(xlim=(1, n), ylim=(-1.0, 1.0))
-----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d ar
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a
  y = column_or_1d(y, warn=True)
**********************************************************************
{'Errores modelo RF : 10': (2.565, 0.834, 19.3, 0.457, 19.47, 0.16)}
{'Errores modelo SVR : 10': (0.843, 0.699, 18.001, 0.503, 4.119, 0.724)}
{}
{'Errores modelo LSTM : 10': (0.484, 0.457, 17.532, 0.193, 3.038, 0.841)}
----------------------------------------------------------------------
{'Tiempo modelo RF : 10': 3.596153020858764 6}
{'Tiempo modelo SVR :  10': 0.23330330848693848}
{}
{'Tiempo modelo LSTM : 10': 1065.9264781475067}
**********************************************************************
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
```

Figura 23. *Retorno de prueba de autocorrelación. LQI*

```
automatizar("/content/drive/MyDrive/PROYECTO -- EXPERIMENTOS/Script limpiar datasets/Datset Limpios LQI/10m/raw_data_run3--1.csv",
            "LQI",True, False)
```

```
---------------------- DICKEY - FULLER ----------------------
ADF Estadistico: -1.528370
Valor p: 0.519434
Valores críticos
        1%: -3.43116
        5%: -2.86190
        10%: -2.56696
-------------------------------------------------------------
/usr/local/lib/python3.6/dist-packages/pandas/plotting/_matplotlib/misc.py:411: UserWarning: Requested projection is different fro
  ax = plt.gca(xlim=(1, n), ylim=(-1.0, 1.0))
-----------------------  SERIE DE DIFERENCIAS -----------------
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:200: DataConversionWarning: A column-vector y was passed when a 1d ar
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a
  y = column_or_1d(y, warn=True)
**************************************************************************
{'Errores modelo RF : 20': (1.647, 0.767, 18.636, 0.49, 18.677, 0.461)}
{'Errores modelo SVR : 20': (1.031, 0.784, 18.448, 0.607, 4.259, 0.663)}
{}
{'Errores modelo LSTM : 20': (0.679, 0.639, 17.836, 0.365, 3.391, 0.778)}
--------------------------------------------------------------------------
{'Tiempo modelo RF : 20': 7.351617813110352}
{'Tiempo modelo SVR :  20': 0.28081679344177246}
{}
{'Tiempo modelo LSTM : 20': 1744.9791107177734}
**************************************************************************
```

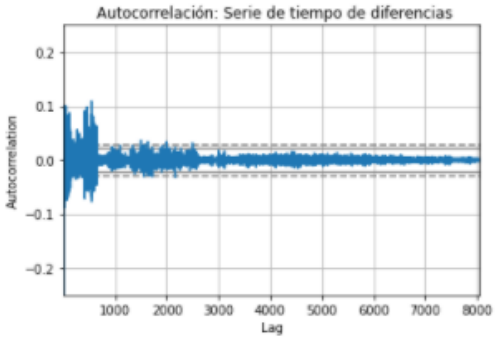
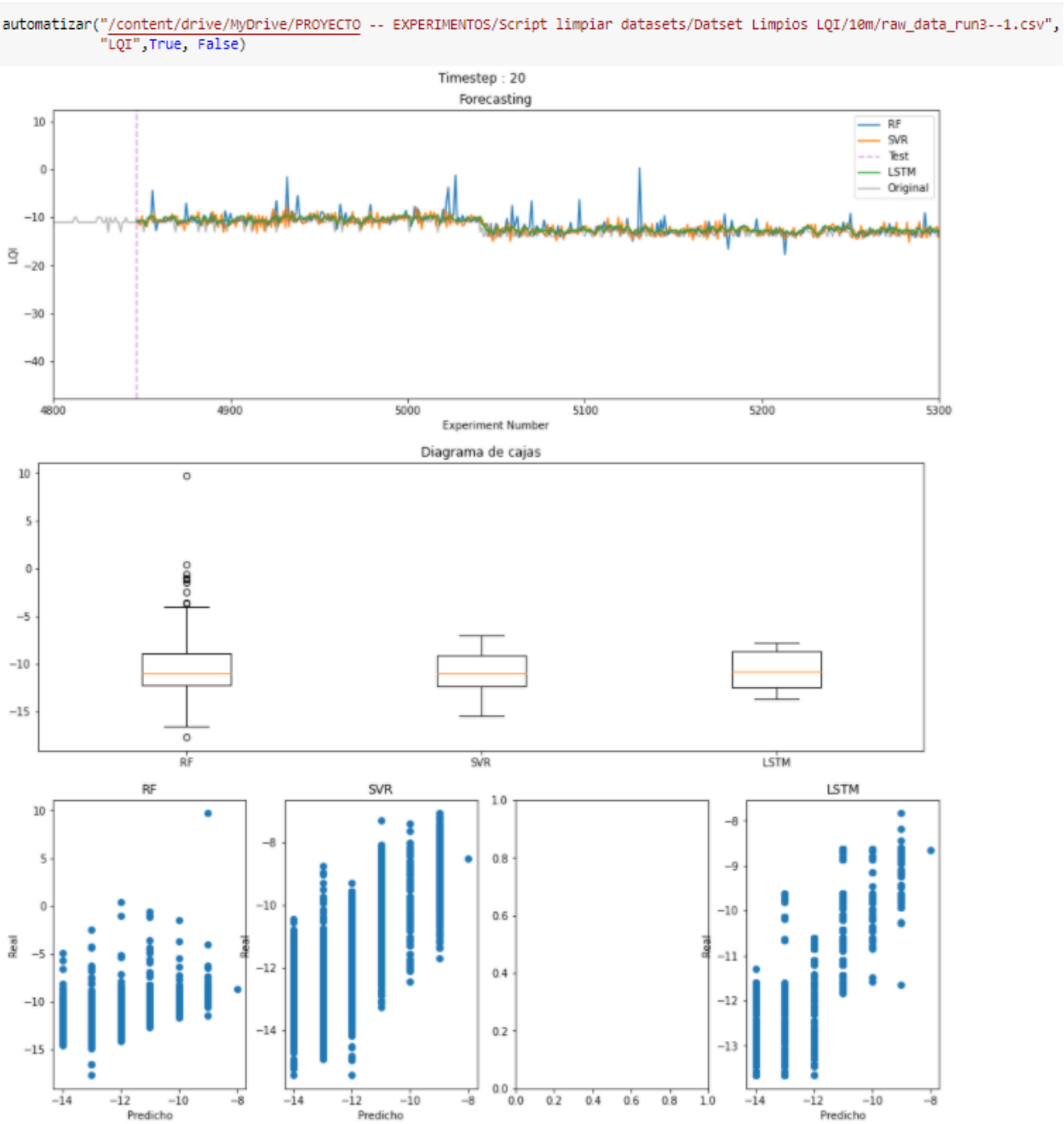


Figura 24. *Retorno de visualización, diagrama de cajas y correlación de las predicciones.*



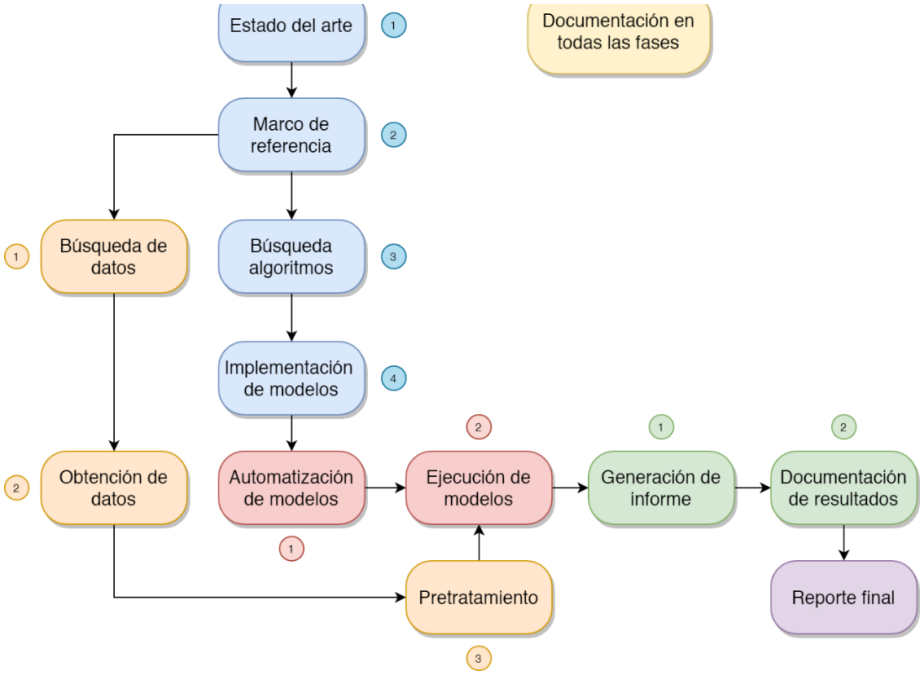
Nota: Las visualización de las predicciones se realizan sobre la sección de prueba de la serie temporal. El eje vertical del diagrama de cajas corresponde a la medida de valores en LQI.

2.2.4 Diagramas operacionales

En esta sección se ilustra la metodología general del proyecto y la metodología de cada uno de los modelos, su forma de operación, generación de predicciones y reportes. Estos diagramas representan con gran precisión la implementación de software que se desarrolló para esta investigación.

2.2.4.1 Metodología. El diagrama en la *figura 25* ilustra todas las fases del proyecto investigativo, la sección representada por el color azul indica el reconocimiento del estado del arte, su parte teórica, aplicación e implementación tecnológica. La sección naranja menciona el proceso de búsqueda de datos aptos para el proyecto, al igual que los procesos de pretratamiento para el ingreso a los modelos. La sección roja implica la automatización, como también la ejecución con los datos pretratados anteriormente. La sección verde describe el proceso de recopilación de resultados, obtención de métricas de error, diagramas de proyecciones y tiempos de ejecución. Finalmente, la sección violeta constituye la generación del reporte final, análisis y conclusiones del trabajo investigativo.

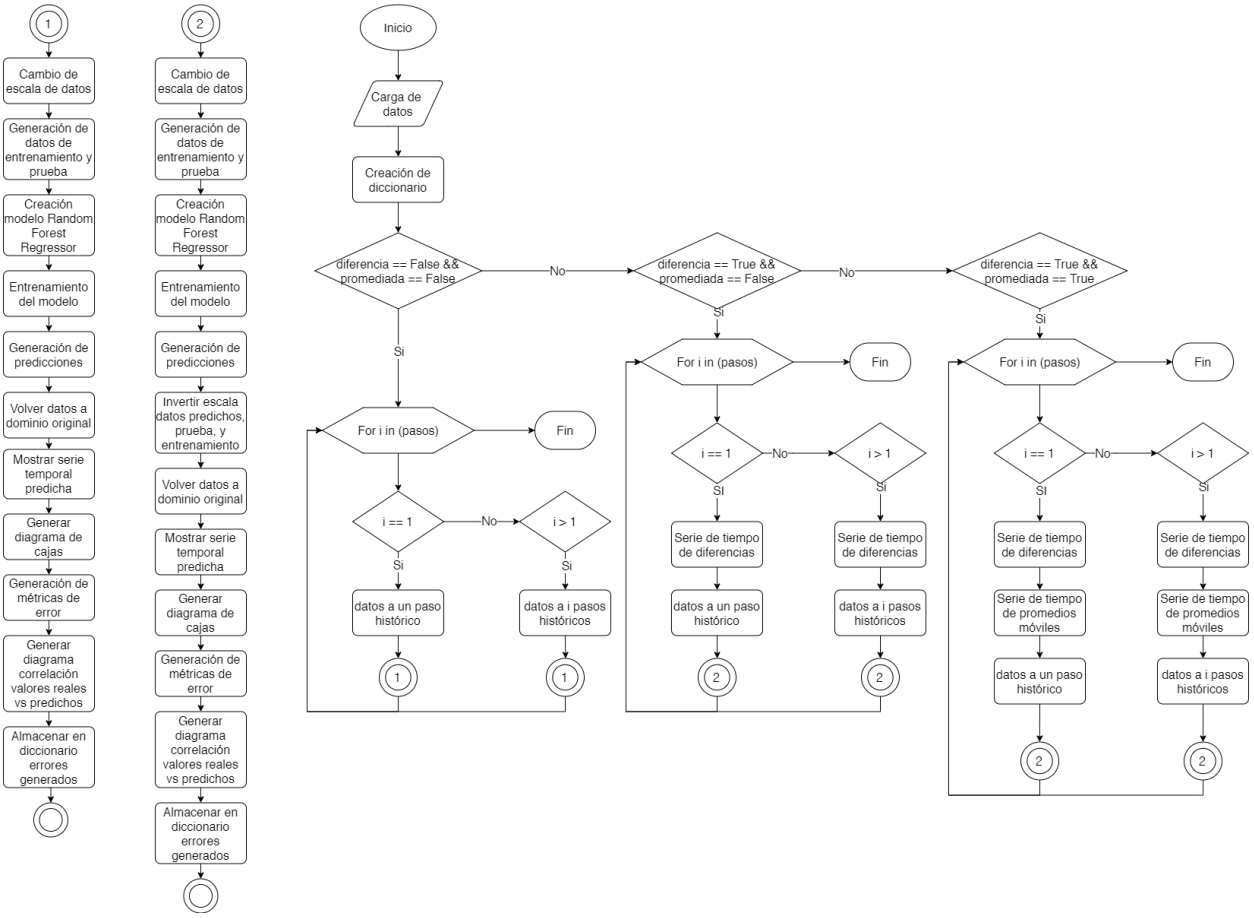
Figura 25. *Diagrama metodológico del proyecto*



2.2.4.2 Metodología de modelos. Los siguientes diagramas mostrarán todo el procedimiento para el inicio, carga y pretratamiento de datos, ejecución de los modelos, generación de predicciones y generación de reportes. Para evitar confusiones, la palabra ‘diccionario’ contenida en ciertas operaciones sólo representa una estructura de datos en donde se almacenan el conjunto de datos de prueba y entrenamiento, las predicciones generadas y el tiempo de ejecución del algoritmo indicado.

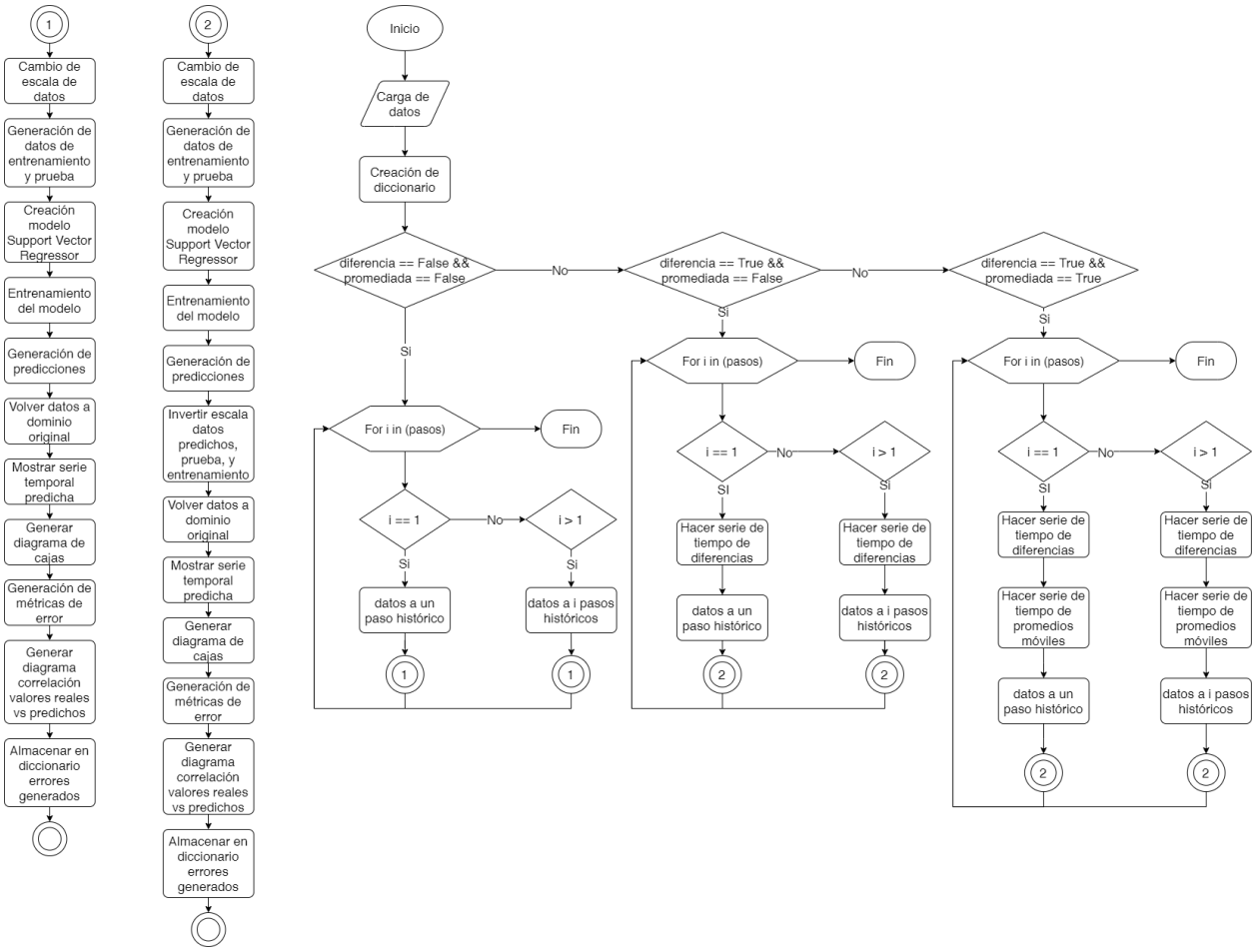
- **Diagrama operacional RF.**

Figura 26. *Diagrama operacional RF.*



- **Diagrama operacional SVR**

Figura 27. *Diagrama operacional SVR*



- **Diagramas operacionales ARIMA**

Figura 28. *Diagrama operacional ARIMA con WFV*

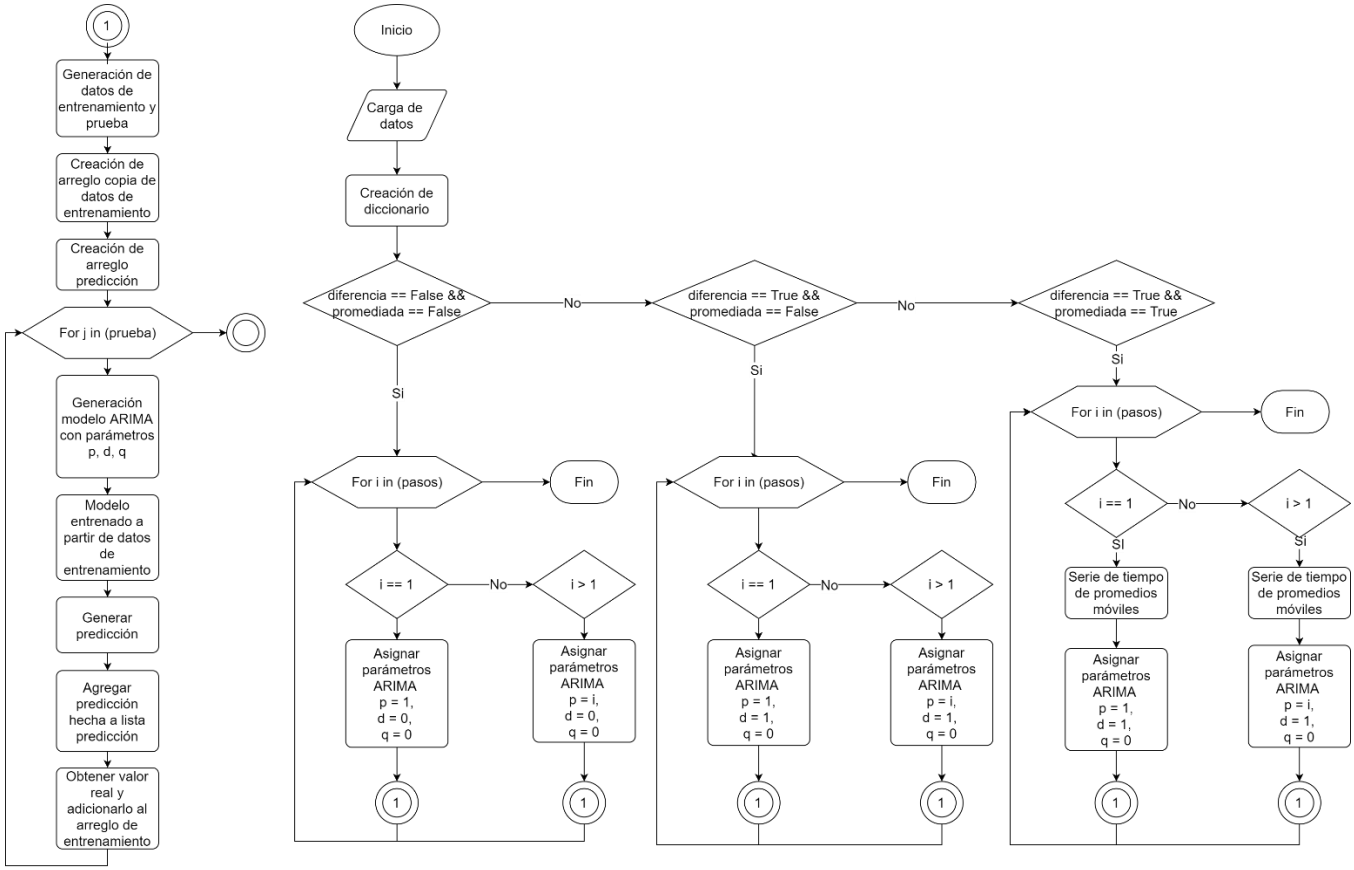
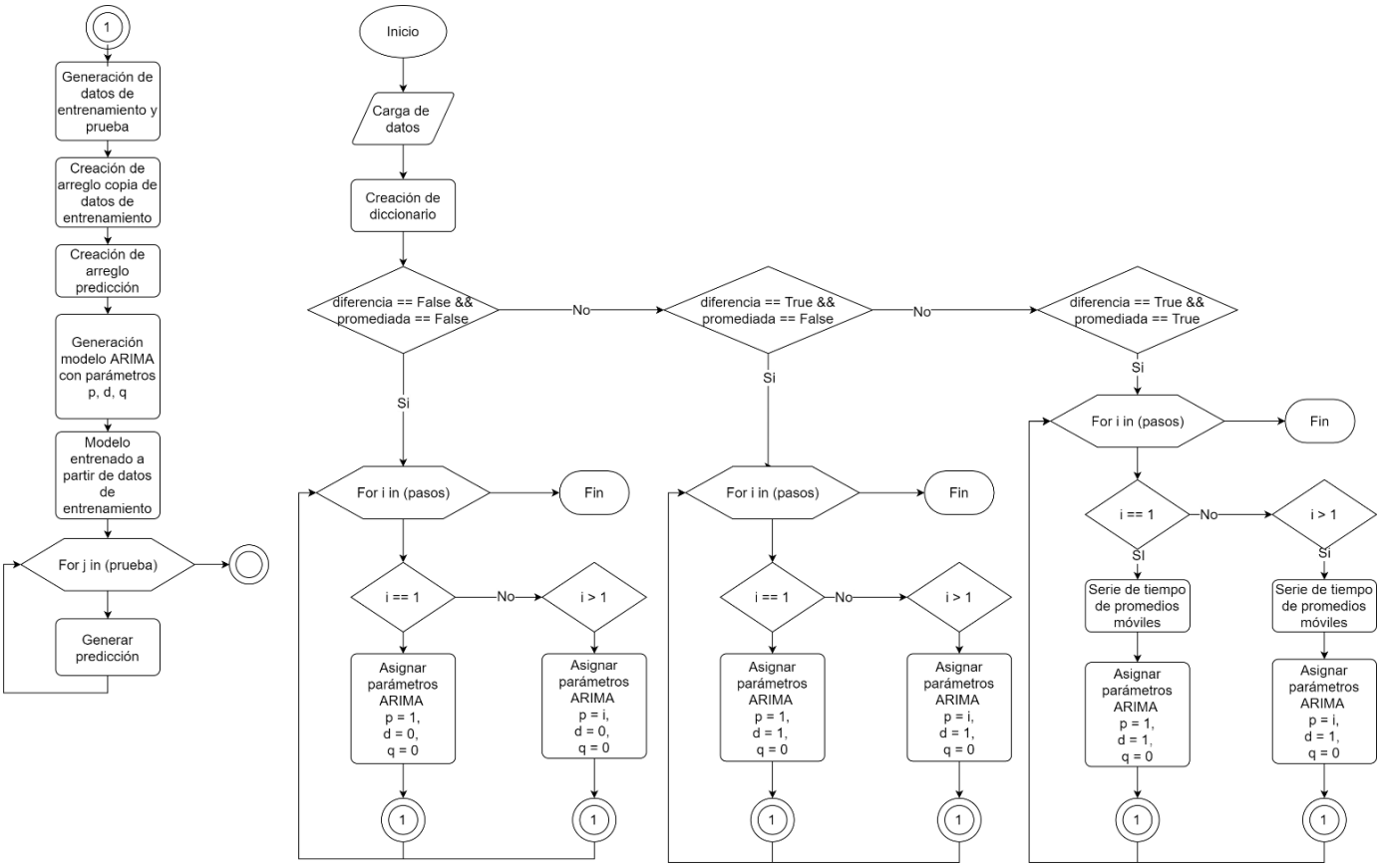
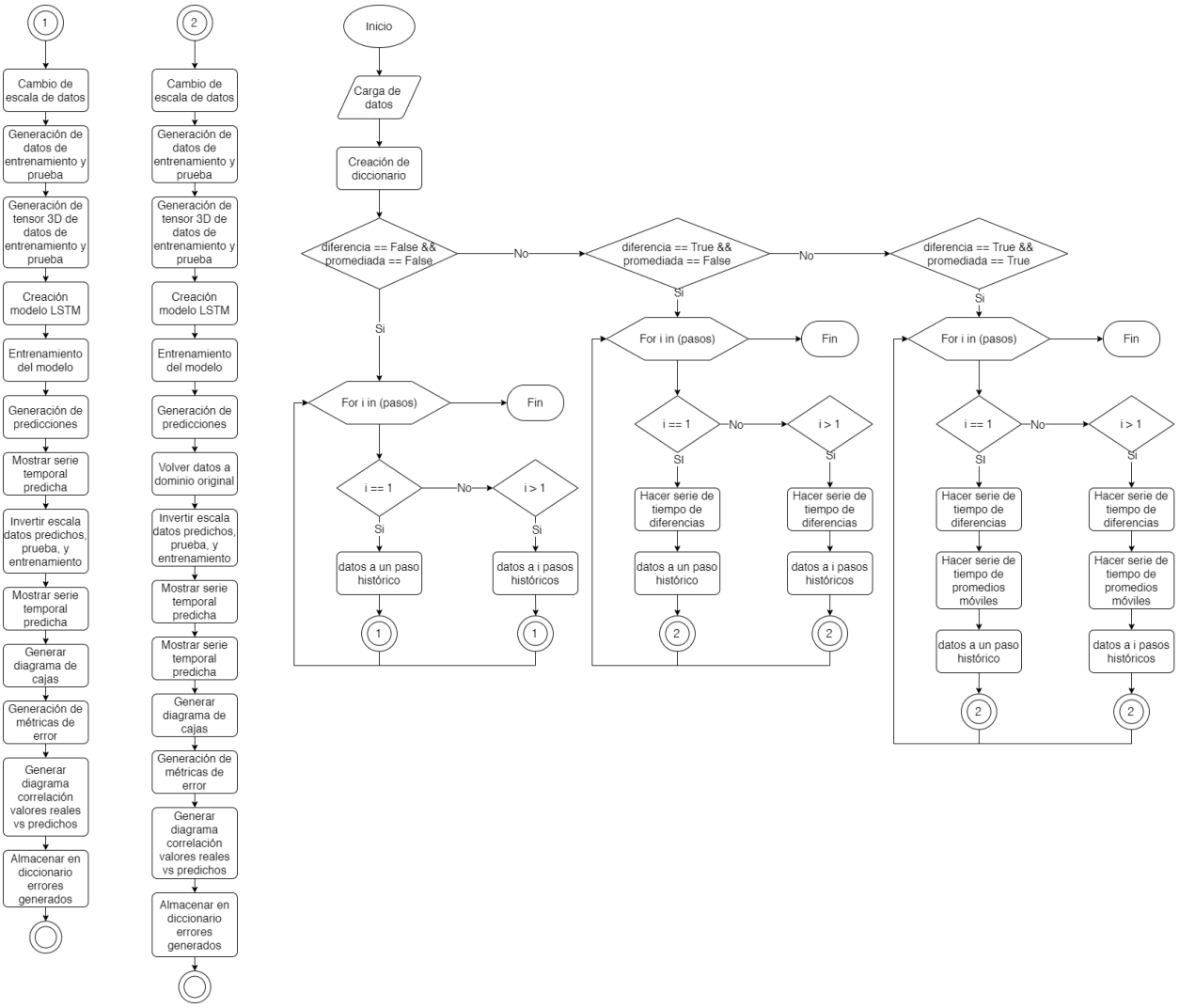


Figura 29. *Diagrama ARIMA sin WFV*



- **Diagrama operacional LSTM**

Figura 30. *Diagrama operacional LSTM*



2.2.5 Tecnologías usadas

2.2.5.1 Entornos y lenguaje de programación.

- **Python:**

  Python es el lenguaje predilecto para la implementación total de este proyecto, caracterizado por ser altamente documentado, versátil, simplificado y rápido, al igual que permite implementar los algoritmos de aprendizaje automático de forma

sencilla. La versión del lenguaje corresponde a la 3.5 ejecutado bajo una arquitectura de procesador *x86/ x64.* Mucha de la implementación fue segmentada en forma de funciones, esto quiere decir que éstas contenían la resolución de un problema determinado acoplándose o realizando llamados a otras funciones, logrando una automatización parcial para la carga, pretratamiento, creación de modelos y generación de resultados.

- **Google Colaboratory:**

  Esta plataforma se caracteriza por dar acceso gratuito a máquinas virtuales con soporte de GPU y TPU. Además de acoplarse con la tecnología de Jupyter Notebook, sus funcionalidades para la programación en Python y la fácil integración con librerías de aprendizaje automático y modelos estadísticos.

- **Jupyter Notebook:**

  Su tecnología incorporada a Google Colaboratory permiten un entorno de programación versátil a través de celdas de código fácilmente implementables y didácticas al observar la salida, gráficos y métricas de error de forma organizada.

2.2.5.2 Tratamiento y visualización de datos. Durante la implementación tecnológica se hace necesario corroborar visualmente los resultados de los algoritmos y la forma de las estructuras de datos por medio de las siguientes tecnologías.

- **Pandas:**

  De gran utilidad para leer los conjuntos de datos en formato *txt* y *csv*, filtrar y reemplazar valores en los registros y generar las series temporales univariantes listas para el ingreso a los modelos.

- **Numpy:**

Esta biblioteca es de bastante uso en este proyecto, dado que es bastante versátil en el manejo de datos con múltiples dimensiones, además de ofrecer una amplia maniobrabilidad y operatividad en los arreglos de datos.

- **Matplotlib:**

  Matplotlib se usó para la generación de los gráficos de las predicciones, autocorrelación, diagramas de cajas, correlación de valores verdaderos y predichos de los cuatro modelos implementados.

2.2.5.3 Aprendizaje automático. Se emplearon diferentes bibliotecas de uso libre para la ejecución propiamente de los algoritmos de aprendizaje de máquina. Todos los aquí mencionados tienen la ventaja de presentar documentación robusta conforme a cada de una de las funcionalidades aplicadas.

- **Statsmodels:**

  Mencionado como “modulo”, *statsmodels* permite acoplar modelos estadísticos a implementaciones de Python. De este módulo se empleó el modelo ARIMA y la prueba de *Augmented Dickey-Fuller*.

- **Scikit-Learn:**

  Scikit-Learn es una librería ampliamente usada para la implementación de modelos de aprendizaje automático, pretratamiento de datos (cambio de escala y análisis de componentes principales *PCA*), métricas de error, tanto para problemas de clasificación como de regresión.

- **Keras:**

Keras es una biblioteca enfocada al uso de rede neuronales y bastante conocida por su facilidad de implementación. De esta biblioteca se resalta el uso de redes recurrentes LSTM y redes densas.

2.2.6 Resultados

Para esta fase se categorizan los experimentos según la métrica de calidad de enlace y metodología, igualmente se muestran los resultados para el fichero *distance_20m*, ya que este presenta la mayor cantidad de series temporales filtradas y experimentos para LQI y RSSI. Se optó por generalizar los resultados realizando la media de los resultados individuales del fichero *distance_20m*.

Las tablas de resultados de la 3 a la 18, muestran las diversas métricas de error que se tuvieron en cuenta para evaluar los modelos, además se muestran los resultados para los experimentos de 5, 10 y 20 pasos temporales. Se optó por una precisión de dos decimales aplicando redondeo a las cifras, esto a fin de mejorar la presentación y análisis de los resultados.

La métrica r^2 se omite en estas tablas por ser impertinentes en este contexto ya que sólo representa la capacidad de predicción de un modelo en una única serie temporal, no obstante, esta métrica está presente en los resultados de los 104 experimentos hechos. Sin embargo, r^2 permite contextualizar la precisión de cada uno de los modelos con respecto a la serie temporal univariante de prueba. En el apéndice A y B podrá notar que r^2 difiere entre los modelos por cada experimento, logrando evidenciar aproximaciones en las predicciones de los modelos en algunas series temporales.

Para las series de tiempo de diferencias se establece el acrónimo *STD,* y para las series de tiempo de diferencias con promedios móviles *STD PM*, esto a fin de simplificar los títulos de las tablas o figuras.

2.2.6.1 RSSI.

2.2.6.1.1 WFV.

Tabla 3. *Resultados para STD con WFV RSSI*

Media de resultados para fichero 20M STD						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	2,77	0,88	18,92	0,61	12,86	1,06
SVR	7,35	2,40	23,86	2,34	12,36	0,16
ARIMA	1,21	0,34	5,93	0,08	12,17	4371,43
LSTM	1,95	0,71	18,16	0,44	12,01	678,75
10 pasos temporales						
RF	3,13	0,89	19,19	0,58	14,98	1,93
SVR	7,45	2,41	23,92	2,36	12,18	0,20
LSTM	2,44	0,93	18,79	0,70	11,90	1011,38
20 pasos temporales						
RF	4,49	0,97	19,86	0,55	16,77	3,88
SVR	7,11	2,36	23,65	2,39	12,18	0,25
LSTM	2,04	0,77	18,31	0,52	11,90	1662,03

Figura 31. *MAPE para STD con WFV RSSI*

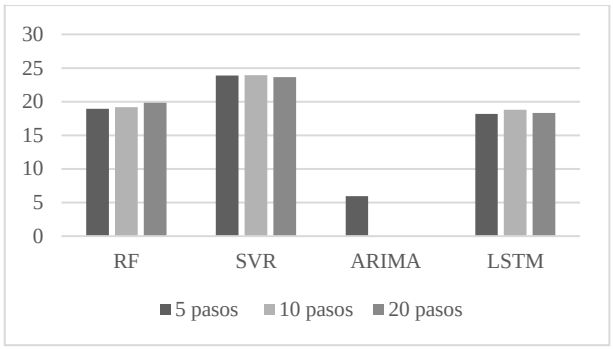


Tabla 4. *Resultados para STD y PM con WFV RSSI*

Media de resultados para fichero 20M STD PM						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	2,10	0,80	18,83	0,57	11,49	1,34
SVR	4,13	1,52	21,03	1,40	11,65	0,17
ARIMA	1,18	0,34	5,56	0,10	11,15	4524,86
LSTM	2,00	0,71	18,53	0,43	11,19	701,10
10 pasos temporales						
RF	2,16	0,80	18,84	0,55	12,67	2,36
SVR	4,11	1,50	20,93	1,36	11,71	0,19
LSTM	1,96	0,70	18,58	0,45	11,41	1069,21
20 pasos temporales						
RF	2,17	0,78	18,79	0,53	12,31	4,78
SVR	4,03	1,48	20,87	1,37	11,84	0,23
LSTM	2,03	0,75	18,68	0,50	11,48	1714,62

Figura 32. *MAPE para STD y PM con WFV RSSI*

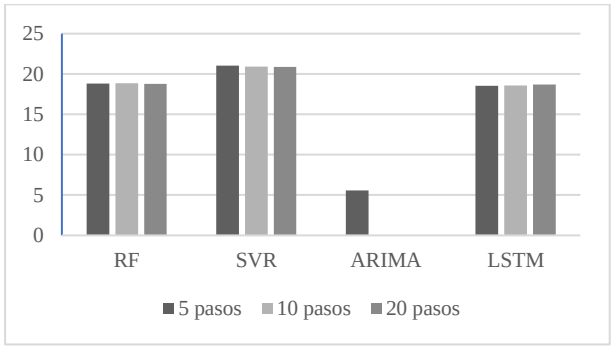


Figura 33. *Predicciones para STD con WFV RSSI*

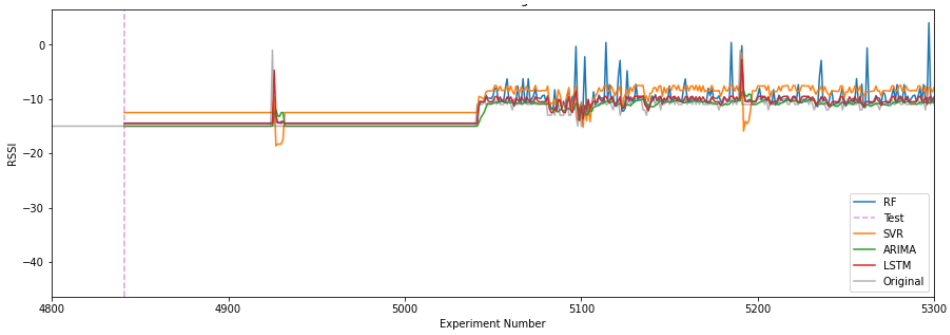


Figura 34. *Diagrama de cajas para predicciones de STD con WFV RSSI*

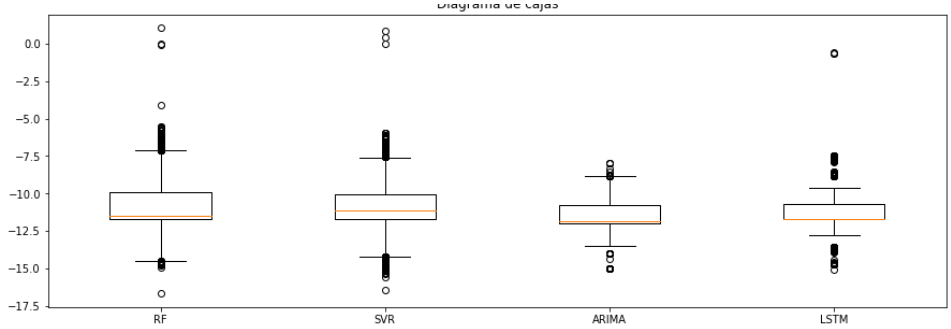


Figura 35. *Predicciones para STD y PM con WFV RSSI*

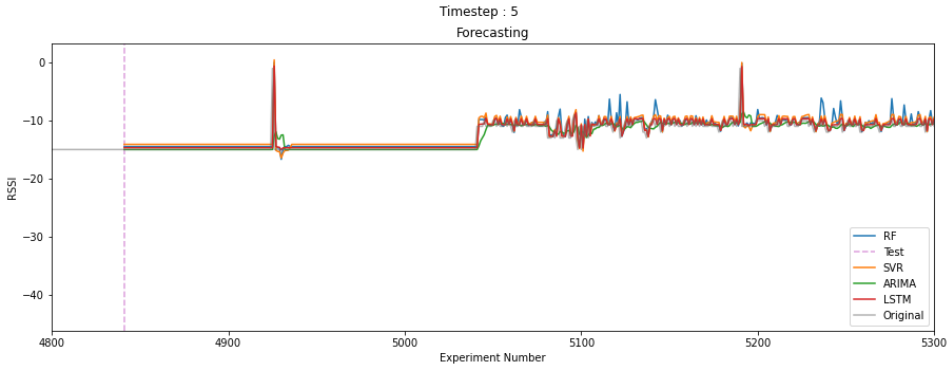


Figura 36. *Diagrama de cajas para predicciones de STD y PM con WFV RSSI*

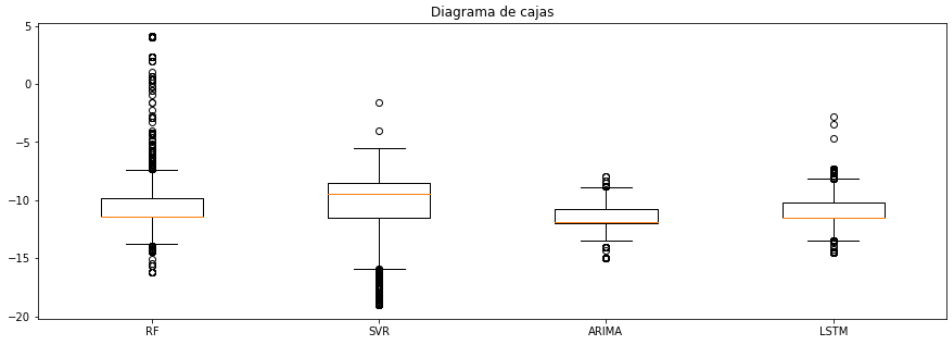


Tabla 5. *Diferencia de MAPE entre STD y STD PM*

	5 pasos	10 pasos	20 pasos	Total
RF	0,09	0,35	4,46	4,91
SVR	2,83	2,99	0,34	6,17
ARIMA	0,37			0,37
LSTM	-0,37	0,22	0,42	0,26

2.2.6.1.2 Sin WFV.

Tabla 6. *Resultados para STD sin WFV RSSI*

Media de resultados para fichero 20M STD						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	2,89	1,02	22,83	0,80	13,19	0,97
SVR	8,43	2,73	28,44	2,72	11,69	0,17
ARIMA	12,05	2,71	31,87	2,14	11,84	2,07
LSTM	2,39	0,88	22,40	0,61	11,05	608,85
10 pasos temporales						
RF	2,97	0,97	22,99	0,69	15,61	1,79
SVR	8,02	2,65	28,14	2,68	10,87	0,21
ARIMA	12,12	2,72	31,97	2,15	11,85	10,88
LSTM	2,12	0,73	22,38	0,43	11,16	906,79
20 pasos temporales						
RF	4,47	1,05	25,10	0,66	18,27	3,67
SVR	7,76	2,60	29,15	2,70	11,54	0,31
ARIMA	13,12	2,88	34,61	2,26	12,70	167,25
LSTM	2,70	0,93	24,15	0,63	11,82	1462,91

Figura 37. *MAPE para STD sin WFV RSSI*

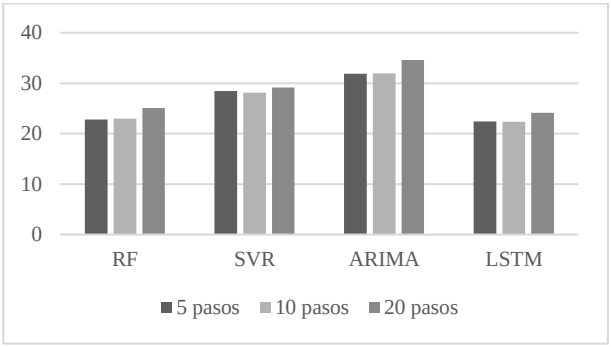


Tabla 7. *Resultados para STD y PM sin WFV RSSI*

Media de resultados para fichero 20M STD PM						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	3,80	0,94	25,58	0,55	12,46	1,33
SVR	4,61	1,37	26,64	1,12	12,21	0,21
ARIMA	12,85	2,79	36,06	2,22	12,03	2,05
LSTM	3,49	0,81	25,47	0,39	11,93	613,23
10 pasos temporales						
RF	3,83	0,95	25,59	0,55	12,61	2,37
SVR	4,58	1,35	26,55	1,09	12,25	0,24
ARIMA	12,92	2,80	36,14	2,23	12,04	10,91
LSTM	3,54	0,88	25,35	0,49	12,06	929,36
20 pasos temporales						
RF	4,03	0,96	25,64	0,53	13,34	5,05
SVR	4,49	1,33	26,46	1,09	12,32	0,29
ARIMA	12,80	2,79	31,90	2,21	12,03	171,43
LSTM	3,49	0,86	25,29	0,46	12,10	1539,54

Figura 38. *MAPE para STD y PM sin WFV RSSI*

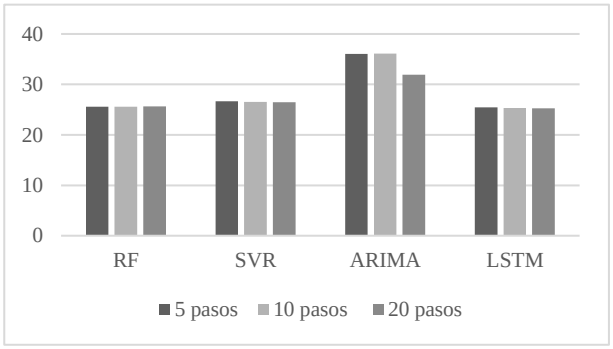


Figura 39. *Predicciones para STD sin WFV RSSI*

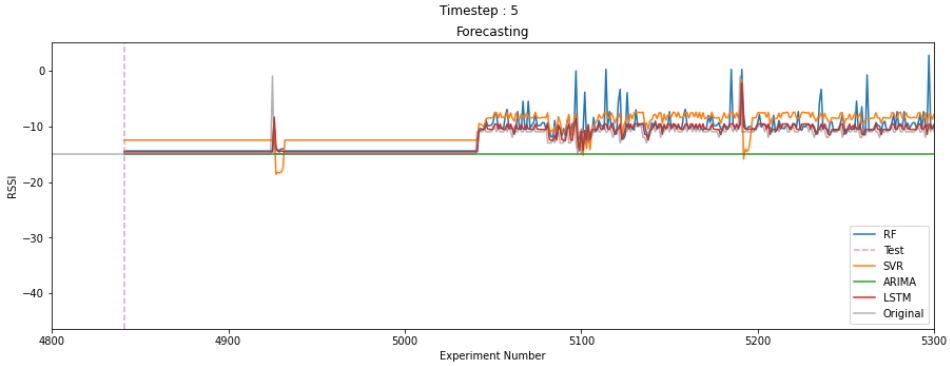


Figura 40. *Diagrama de cajas para predicciones de STD sin WFV RSSI*

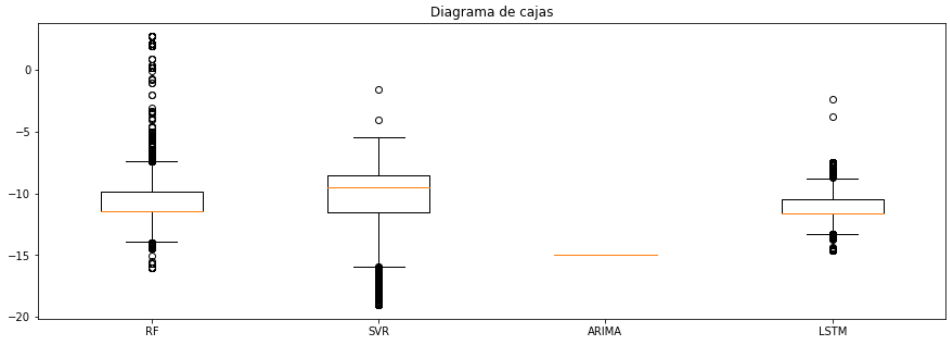


Figura 41. *Predicciones para STD y PM sin WFV RSSI*

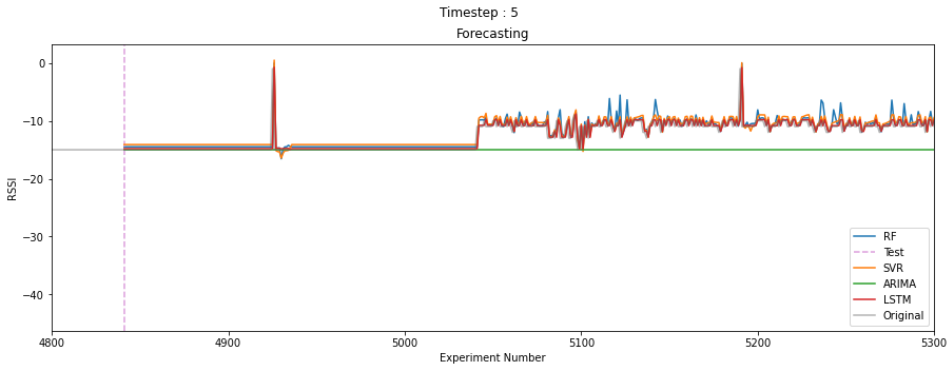


Figura 40. *Diagrama de cajas para predicciones de STD y PM sin WFV RSSI*

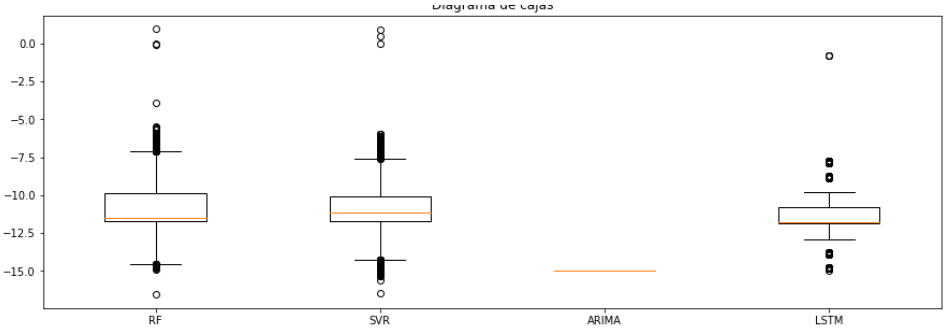


Tabla 8. *Diferencia de MAPE entre STD y STD PM*

	5 pasos	10 pasos	20 pasos	Total
RF	-2,75	-2,60	-0,54	-5,89
SVR	1,80	1,60	2,69	6,09
ARIMA	-4,18	-4,17	2,71	-5,65
LSTM	-3,08	-2,97	-1,14	-7,18

Tabla 9. *Diferencias de tiempo de ejecución RSSI*

	WFV	Sin WFV	Diferencia (s)
STD	4371,43	2,07	4369,36
STD PM	4524,86	2,05	4522,81

*Nota**:** Esta tabla se crea a partir de los tiempos de ejecución del modelo ARIMA con y sin WFV para la métrica RSSI.*

2.2.6.2 LQI.

2.2.6.2.1 WFV.

Tabla 10. *Resultados para STD con WFV LQI*

Media para resultados para fichero 20M STD						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	4,36	1,19	25,78	0,85	14,34	1,02
SVR	9,01	2,79	30,97	2,78	11,98	0,19
ARIMA	1,98	0,48	12,18	0,15	11,72	4269,91
LSTM	3,50	1,14	25,19	0,84	11,05	692,30
10 pasos temporales						
RF	4,22	1,12	25,96	0,71	16,26	1,87
SVR	8,30	2,66	30,38	2,71	11,13	0,23
LSTM	3,04	1,07	25,50	0,74	11,22	1041,41
20 pasos temporales						
RF	5,50	1,17	26,54	0,68	16,99	3,72
SVR	8,08	2,62	30,05	2,73	11,11	0,30
LSTM	2,80	1,01	24,75	0,71	10,90	1720,22

Figura 41. *MAPE para STD con WFV LQI*

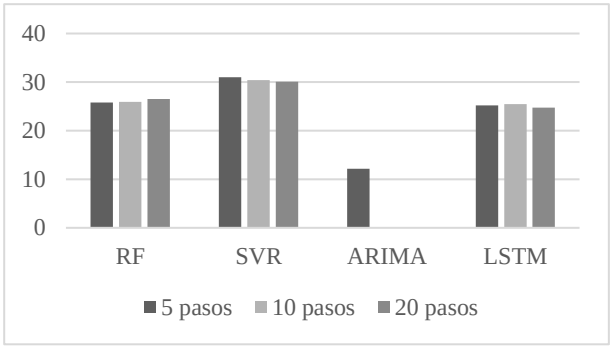


Tabla 11. *Resultados para STD y PM con WFV LQI*

Media de resultados para fichero 20M STD PM						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	3,62	0,94	25,58	0,54	12,59	1,40
SVR	4,06	1,23	26,19	0,96	12,59	0,22
ARIMA	1,88	0,48	12,18	0,15	11,72	4270,68
LSTM	3,40	0,81	25,25	0,38	12,22	702,67
10 pasos temporales						
RF	3,68	0,94	25,58	0,54	12,45	2,50
SVR	4,06	1,21	26,10	0,94	12,66	0,24
LSTM	3,51	0,89	25,40	0,50	12,32	1049,55
20 pasos temporales						
RF	3,64	0,93	25,53	0,52	12,40	5,24
SVR	4,03	1,20	26,04	0,94	12,70	0,29
LSTM	3,49	0,89	25,50	0,49	12,36	1731,72

Figura 42. *MAPE para STD y PM con WFV LQI*

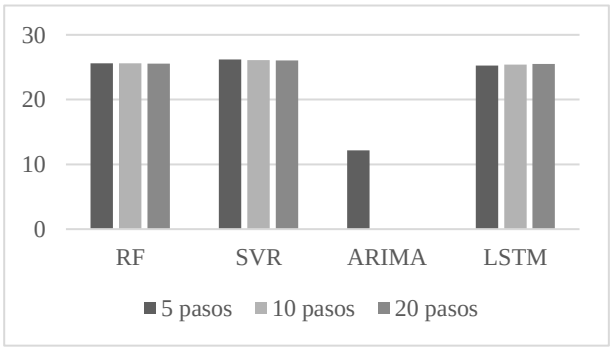


Figura 45. *Predicciones para STD con WFV LQI*

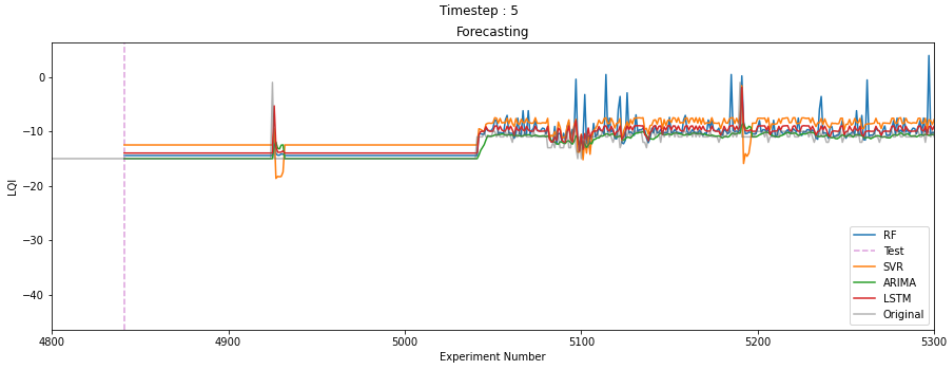


Figura 48. *Diagrama de cajas para predicciones de STD con WFV LQI*

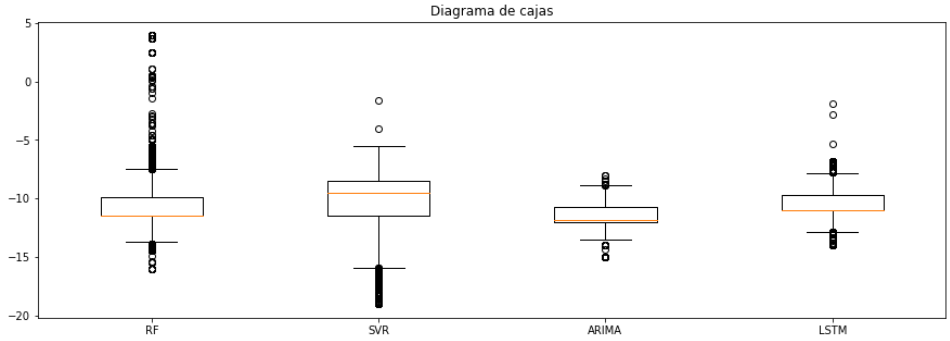


Figura 47. *Predicciones para STD y PM con WFV LQI*

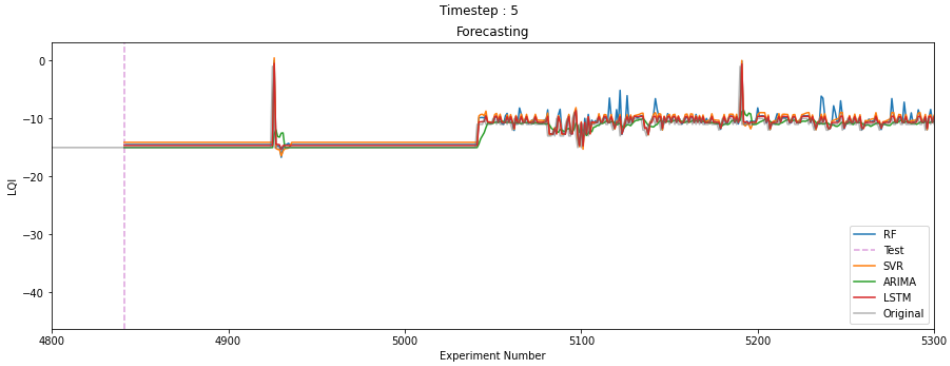


Figura 46. *Diagrama de cajas para predicciones de STD y PM con WFV LQI*

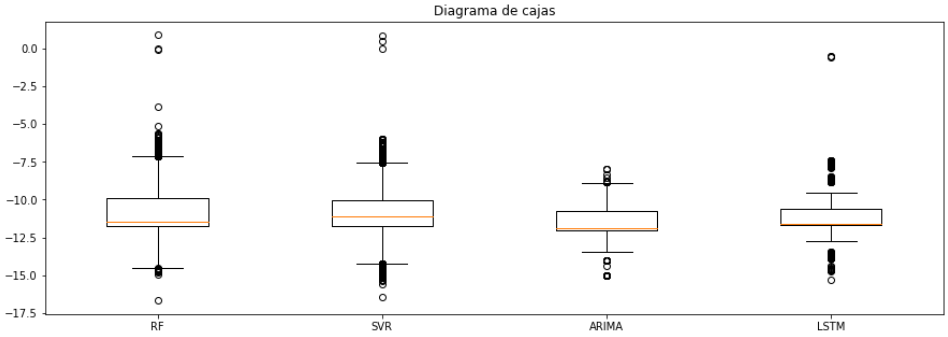


Tabla 12. *Diferencia de MAPE entre STD y STD PM*

	5 pasos	10 pasos	20 pasos	Total
RF	0,19	0,37	1,01	1,58
SVR	4,78	4,28	4,01	13,07
ARIMA	0,00			0,00
LSTM	-0,06	0,00	-0,74	-0,80

2.2.6.2.2 Sin WFV.

Tabla 13. *Resultados para STD sin WFV LQI*

Media de resultados para fichero 20M STD						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	4,19	1,18	25,72	0,85	14,05	0,99
SVR	9,01	2,79	30,97	2,78	11,98	0,26
ARIMA	12,85	2,79	36,06	2,22	12,03	2,09
LSTM	3,14	0,95	25,11	0,60	11,31	632,00
10 pasos temporales						
RF	4,24	1,12	25,96	0,71	16,14	1,86
SVR	8,30	2,66	30,38	2,71	11,13	0,23
ARIMA	12,92	2,80	36,14	2,23	12,04	10,10
LSTM	3,48	1,11	25,15	0,79	11,22	913,41
20 pasos temporales						
RF	5,52	1,17	26,57	0,69	17,66	3,62
SVR	8,12	2,63	30,12	2,73	11,25	0,31
ARIMA	12,80	2,79	35,99	2,21	12,03	169,72
LSTM	3,07	0,94	25,17	0,58	11,87	1471,36

Figura 47. *MAPE para STD sin WFV LQI*

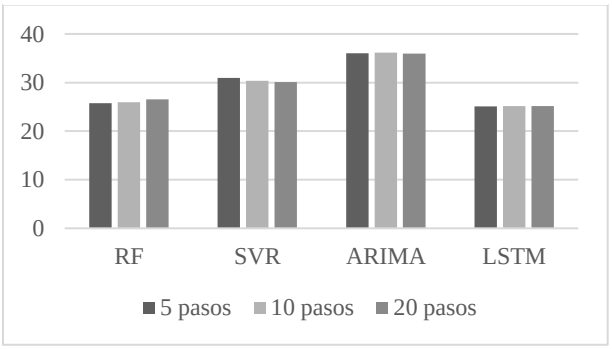


Tabla 14. *Resultados para STD y PM sin WFV LQI*

Media de resultados para fichero 20M STD PM						
Error	**MSE**	**MAE**	**MAPE**	**MedianAE**	**Max Error**	**Tiempo de ejecución (s)**
5 pasos temporales						
RF	0,89	0,94	25,58	0,54	12,57	1,35
SVR	1,26	1,23	26,19	0,96	12,59	0,22
ARIMA	16,58	2,79	36,05	2,22	12,03	2,10
LSTM	0,80	0,82	25,38	0,40	12,11	634,26
10 pasos temporales						
RF	0,93	0,94	25,57	0,54	12,47	2,41
SVR	1,30	1,21	26,10	0,94	12,66	0,24
ARIMA	16,71	2,80	36,13	2,23	12,04	11,27
LSTM	0,87	0,90	25,40	0,51	12,30	915,21
20 pasos temporales						
RF	0,93	0,93	25,53	0,53	12,42	5,12
SVR	1,30	1,20	26,04	0,94	12,70	0,28
ARIMA	16,17	2,79	36.01	2,21	12,03	169,48
LSTM	0,86	0,88	25,43	0,49	12,30	1482,44

Figura 48. *MAPE para STD y PM sin WFV LQI*

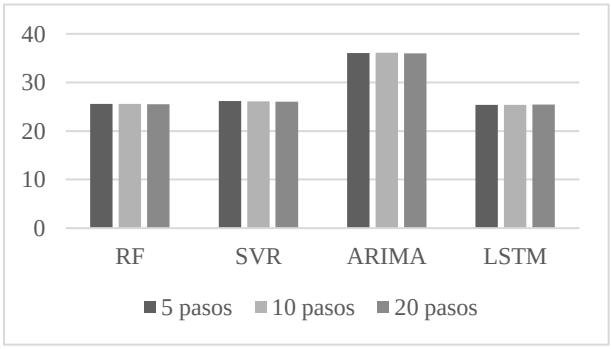


Figura 51. *Predicciones para STD sin WFV LQI*

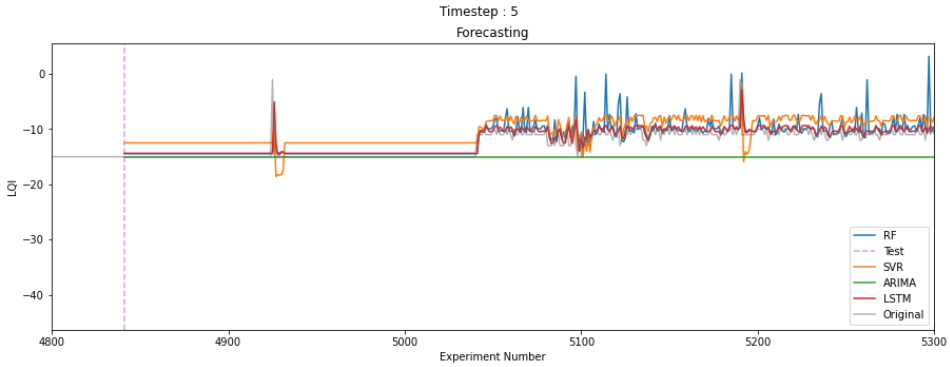


Figura 54. *Diagrama de cajas para predicciones de STD sin WFV LQI*

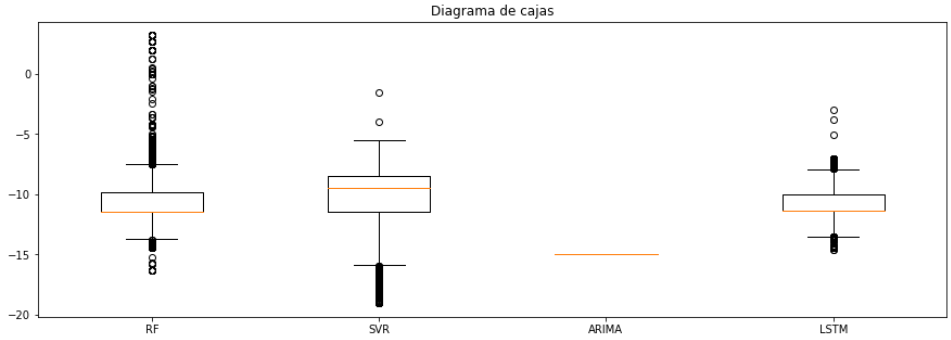


Figura 53. *Predicciones para STD y PM sin WFV LQI*

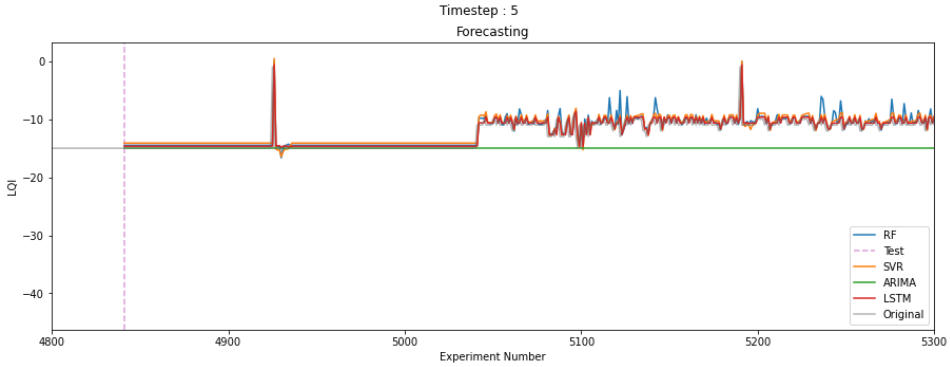


Figura 52. *Diagrama de cajas para predicciones de STD y PM sin WFV LQI*

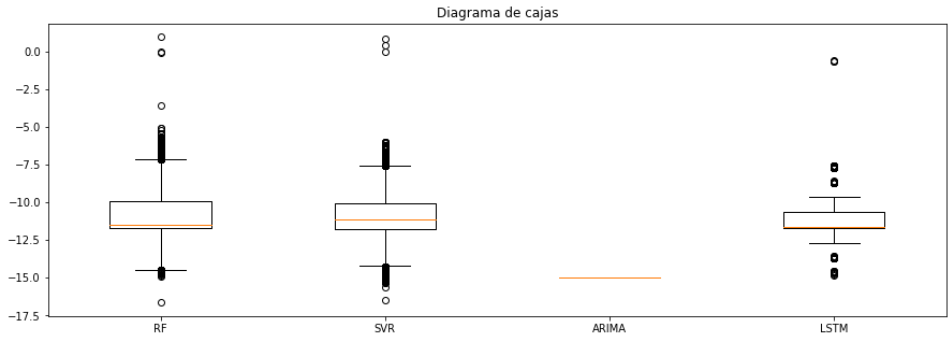


Tabla 15. *Diferencia de MAPE entre STD y STD PM*

	5 pasos	10 pasos	20 pasos	Total
RF	0,14	0,39	1,04	1,57
SVR	4,78	4,28	4,07	13,07
ARIMA	-0,06	0,01	-0,02	-0.07
LSTM	-0,27	-0,24	-0,26	-0.77

Tabla 16. *Diferencias de tiempo de ejecución LQI*

	WFV	Sin WFV	Diferencia (s)
STD	4269,91	2,09	4267,82
STD PM	4270,68	2,05	4268,63

*Nota**:** Esta tabla se crea a partir de los tiempos de ejecución del modelo ARIMA con y sin WFV para la métrica LQI.*

2.2.6.3 Discusión. Es importante mencionar la implicación de la consideración de más pasos temporales para generar predicciones, esto quiere decir que, a mayor cantidad de pasos temporales tenidos en cuenta, mayor es el tiempo de entrenamiento por la cantidad de parámetros o componentes que se deben entrenar en los modelos. Esto lo reflejan las tablas 3, 4, 6, 7, 10 y 11, donde aumentan los tiempos de ejecución de los modelos cuando se consideran más pasos temporales.

Una de las implicaciones experimentales que se evidenciaron en la fase de resultados, fue la inclusión de la metodología *WFV* en el modelo ARIMA, siendo éste el principal factor para que la precisión en las predicciones fuera notable ante los otros modelos propuestos, esto lo evidencian las tablas 3, 4, 10 y 11, en donde los errores MAPE, MAE y MSE son bajos comparados a los otros modelos propuestos.

Experimentalmente se obtuvo una mejora en la atenuación del error, en términos de MAPE, para SVR al aplicar promedios móviles a las series temporales. Las tablas 5, 8, 12 y 17 demuestran que SVR presenta una reducción del error MAPE de más del 6% para todas las tablas de diferencias.

Por el lado de las redes LSTM, éstas presentaron un alto tiempo de ejecución para las series temporales dadas como entrada. Si bien es cierto que presentan una diferencia pequeña del error con respecto a RF, no es significativa en términos porcentuales de MAPE ni en unidades de error cuadrático medio MSE. En adición, este modelo no se ve beneficiado al aplicar promedios móviles a las series temporales, de hecho, en las tablas 8, 12 y 17 se evidencia el incremento del error porcentual para los cinco, diez y veinte pasos temporales.

La experimentación con RF infiere que este modelo llega a nivelarse en términos de error porcentual a LSTM y a ser menor con respecto a SVR para métricas de error MAPE, MAE y

Median Absolute Error. A su vez, este modelo presentó un tiempo de ejecución bastante menor que LSTM y ARIMA con y sin *WFV*, esto se evidencia en las tablas 3 a la 14 donde el modelo no supera los 6 segundos durante el entrenamiento.

WFV es una metodología favorable, en este caso, para el modelo ARIMA. Sin embargo, este concepto y forma de realizar predicciones puede ser aplicado a los demás modelos, ver sus implicaciones en la precisión de las predicciones y los tiempos de ejecución asociados a cada modelo.

3 Conclusiones

La metodología *WFV* para el modelo ARIMA permite un bajo error en términos de MSE, MAE, MAPE y Median Absolute Error en las métricas de calidad de los enlaces RSSI y LQI. Esto se evidencia en las tablas 3, 4, 10 y 11.

Sin *WFV* el modelo ARIMA presenta errores superiores, tanto en métricas de error MSE, MAE, MAPE, y Median Absolute Error, que a cualquiera de los otros modelos evaluados. Las tablas 6, 7, 14 y 15 así lo demuestran.

Las tablas 9 y 18 muestran una diferencia de más de 4000 segundos (66.66 minutos) entre los experimentos en donde se aplicó *WFV* para ARIMA y en donde no, lo que implica que esta metodología presenta un tiempo de ejecución considerable.

La metodología de aplicación en LSTM no presenta una ventaja significativa comparado a los otros dos modelos propuestos, siendo el tiempo de ejecución la característica que presenta mayor desventaja.

Aplicar el tratamiento de promedios móviles reduce el error de MAPE para el modelo SVR, así lo señalan las figuras32, 38, 44 y 50. Cabe hacer mención que también aplica para otras métricas de error como MSE, MAE y Median Absolute Error.

RF presentó un rendimiento equiparable a LSTM, esto se observa en las tablas de resultados para las dos métricas. De igual manera se destaca el tiempo de ejecución, que es inferior a LSTM.

4 Recomendaciones

- Uso de series temporales multivariantes para los algoritmos de SVR, RF y LSTM.
- Análisis de la correlación en las series temporales multivariantes para comprender la relación o impacto que presentan sobre las observaciones.
- Integración de metodología *WFV* en modelos como SVR, RF o LSTM.
- Uso de series temporales que posean variabilidad en el tiempo y no presenten cambios repentinos en instantes de tiempo cortos.
- Modificación en la operatividad de *WFV*, restringiendo o definiendo el tamaño del conjunto de entrenamiento a fin de controlar el tiempo de ejecución.
- Aumentar la cantidad de celdas de memoria, capas ocultas y número de épocas en redes LSTM.

Referencias Bibliográficas

Breiman, L. (1999). *Random Forests - Random Features, Tecnical Report 567, Statistic Department, University of California, Berkeley, (https://www.stat.berkeley.edu/~breiman/random-forests.pdf, 08.10.2018'de erişildi)*. 1–29.

Brownlee, J. (2018). *How to Develop LSTM Models for Time Series Forecasting*. Https://Machinelearningmastery.Com/. https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/

Brownlee, J. (2020). *How to Create an ARIMA Model for Time Series Forecasting in Python*. Www.Machinelearningmastery.Com. https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

Cao, J., Mao, K., Wu, J., & Lendasse, A. (2015). Proceedings of ELM-2015 Volume 1, Theory,Algorithms and Application(I). *Proceedings in Adaptation, Learning and Optimization*, *1*. https://doi.org/10.1007/978-3-319-28397-5

Chang, C. C., & Lin, C. J. (2011). LIBSVM: A Library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1–39. https://doi.org/10.1145/1961189.1961199

Fu, S., & Zhang, Y. (2015). *{CRAWDAD} dataset due/packet-delivery (v. 2015-04-01)*. https://doi.org/10.15783/C7NP4Z

Fu, S., Zhang, Y., Jiang, Y., Hu, C., Shih, C. Y., & Marrón, P. J. (2015). Experimental Study for Multi-layer Parameter Configuration of WSN Links. *Proceedings - International Conference on Distributed Computing Systems*, *2015-July*(June), 369–378. https://doi.org/10.1109/ICDCS.2015.45

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term-Memory. *Neural Computation*, *9*(8),

1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hofmann, D., & Dinges, P. (2006). *A First Course on Time Series Analysis*.

Mills, T. C. (2019). Time Series and Their Features. *Applied Time Series Analysis*, 1–12. https://doi.org/10.1016/b978-0-12-813117-6.00001-6

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Rey Graña, C., & Ramil Diaz, M. (2011). Series temporales. *Introduccion a La Estadistica Descriptiva. Segunda Edicion*, 85–105. https://doi.org/10.4272/978-84-9745-167-3.ch4

Seabold, S., & Perktold, J. (2010). statsmodels: Econometric and statistical modeling with python. *9th Python in Science Conference*.

Segal, M. R. (2004). Machine Learning Benchmarks and Random Forest Regression. *Biostatistics*, 1–14. http://escholarship.org/uc/item/35x3v9t4.pdf

Seymour, L., Brockwell, P. J., & Davis, R. A. (1997). Introduction to Time Series and Forecasting. In *Journal of the American Statistical Association* (Vol. 92, Issue 440, p. 1647). https://doi.org/10.2307/2965440

Tilgner, M. (2019). *Time series forecasting with random forest*. Www.r-Bloggers.Com. https://www.r-bloggers.com/2019/09/time-series-forecasting-with-random-forest/

Yuan, D., Kanhere, S. S., & Hollick, M. (2017). Instrumenting Wireless Sensor Networks — A survey on the metrics that matter. *Pervasive and Mobile Computing*, *37*, 45–62. https://doi.org/10.1016/j.pmcj.2016.10.001

Zhang, Z. (2020). Time series: a data analysis approach using R. In *Journal of Quality Technology*.

https://doi.org/10.1080/00224065.2020.1714517

Apéndices