UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

# MODELO DE PREDICCIÓN DE FALLOS A EQUIPO CRÍTICO EN UNA PLANTA DE CELULOSA

Trabajo de Titulación para optar al Título de Ingeniero en Mantenimiento Industrial

Alumno:
Pablo Andrés Reinoso Órdenes.

Profesor Guía:
Mg. Ing. Andrés Eduardo Aránguiz Garrido.

**2022**

## DEDICATORIA

*A mi familia: Sylvia, Ramón y Ermelinda, quienes siempre me han brindado su apoyo incondicional, cariño sincero y confianza.*
*A mi hermano y sobrino, quienes fueron una luz en momentos que más lo necesitaba.*
*A mis amigos, quienes a pesar de estar en diferentes caminos y etapas, su cariño ha sido de vital importancia en este proceso.*
*A mis compañeros Tiaren, Carlos y José, con quienes compartimos buenos y malos momentos, y nunca nos permitimos caer.*

# RESUMEN

Existe antecedentes empíricos, que al combinar un modelo Autorregresivo (AR) para series temporales de datos, se logran buenos resultados; y que al combinarlos con herramientas de machine Learning, se consigue una mejoría en los datos obtenidos. Específicamente al aplicar el un modelo Autorregresivo (AR), sobre una histórico de fallas, se obtiene una nueva serie de datos, la cual se ingresa en un algoritmo de Redes Neuronales Artificiales (RNA), consiguiendo minimizar en gran amplitud el error de predicciones con respecto a los datos reales.

Se aplica el modelo de vida Weibull, a un histórico de fallas de un equipo crítico en una planta de celulosa, obteniendo predicciones de estos eventos, y determinando errores con respecto a los tiempos reales. Posteriormente se estudia un modelo autorregresivo integrado de media móvil (ARIMA), obteniendo una serie de datos que se ingresarán a un algoritmo de redes neuronales artificiales, generando estimaciones y comparándolas con los tiempos reales de funcionamiento. Posteriormente se propondrá una mejora en el modelo, con la finalidad de minimizar los errores de pronóstico derivados del modelo.

Como primer ítem se emplea el modelo Weibull, indicando un MTBF de 111 horas, y que el activo se encuentra en una etapa de rodaje. Posteriormente se aplica el modelo Weibull de una forma distinta, agrupando los 8 primeros valores de la muestra real, a estos se le aplica Weibull y se obtiene el valor 9; luego se agrupan los 9 primeros valores de la data, se aplica Weibull y se obtiene el 10, este paso se repite hasta obtener la totalidad de la muestra. Se aplica el modelo ARIMA (1.0.2), parámetros obtenidos mediante un algoritmo de iteración. Al obtener las predicciones de ARIMA, se ingresan los datos obtenidos a una red neuronal artificial. Como propuesta de mejora, se ingresan los datos directos desde el histórico de fallos, a la red neuronal artificial, iterando los hiperparámetros de esta, logrando así una notoria mejoría en el proceso. Para la estimación de errores se emplean métricas estadísticas: RMSE, MAD y MAPE.

El modelo Weibull ajustado obtiene, un valor RMSE de 342,39, mientras que el modelo ARIMA(1.0.2) obtiene un valor RMSE de 337,8, mejorando así la métrica de Weibull. Desde el modelo ARIMA(1.0.2) + RNA(1.50.40.30.15.1), se obtiene un RMSE de 50,24, mejorando los obtenidos en los otros modelos. Los resultados obtenidos por el modelo propuesto, son aún más satisfactorios con un valor RMSE de 4,75, un error absoluto medio de 2,4 horas y un error porcentual medio de 7,14%, mejorando los tres modelos antes analizados.

# ÍNDICE

# ÍNDICE DE FIGURAS

# INTRODUCCIÓN

En un mundo que evoluciona constantemente, tratando de generar mejoras tanto para la sociedad como para el medio ambiente, es casi imposible pensar que el mantenimiento industrial no lo haría; y es que pasamos desde simples rutinas de lubricación y aseo, a complejos planes de mantenimiento, en donde se consideran todo tipo de recursos necesarios para ejecutarlos.

Ahora bien, el mantenimiento predictivo ha ido tomando mayor relevancia con el pasar del tiempo, y es que su eficacia está a comprobada, pero el alto costo que conlleva el monitoreo de activos, lo hace ser poco eficiente. El mantenimiento predictivo es clave, es decir, el poder anticipar eventos no deseados, dentro de los activos que cuentan las organizaciones de manufactura. Cuando los activos fallan, no sólo se ve afectada la capacidad de generar riqueza, además se suman interrupciones de servicios, desastres que afectan a la población, e incluso eventos que afectan a la humanidad.

La predicción aplicada en el mantenimiento, nos sitúa delante de eventos inesperados que implican fallas funcionales parciales y totales de los equipos.

Pero… ¿cómo validamos si realmente se logran predecir los eventos de falla en una industria?

Este proyecto de título, se basa en elaborar una propuesta de mejora en la anticipación de eventos, para un equipo crítico en una planta de celulosa, evaluando diferentes alternativas de modelamiento para la vida de este.

Utilizando el error medio cuadrático como instrumento de validación, el error absoluto de cada predicción, para medir la imprecisión en cada una de estas y el error porcentual para medir la calidad del dato obtenido.

# OBJETIVOS

## OBJETIVO GENERAL

Elaborar una propuesta de mejora en la predicción de eventos de falla, empleando un modelo para series de datos temporales y entrenando un algoritmo de machine learning, para un equipo crítico en una planta de celulosa.

## OBJETIVOS ESPECÍFICOS

Analizar base de datos histórica de fallas mediante el modelo de uso de vida Weibull, para obtener predicciones de eventos y determinar errores de estimación, con respecto a los tiempos reales de funcionamiento.

Diseñar un algoritmo óptimo de aprendizaje supervisado, a través de un modelo autorregresivo integrado de media móvil y modelo de redes neuronales artificiales, para obtener predicciones y determinar errores de estimación con relación a los tiempos reales de funcionamiento

Generar una propuesta de mejora en el modelo base, mediante la iteración de parámetros e hiperparámetros con la finalidad de minimizar los errores de estimación con relación a los otros modelos y a los tiempos reales de funcionamiento.

# CAPÍTULO 1: MODELO DE USO DE VIDA WEIBULL

# 1. MODELO DE USO DE VIDA WEIBULL

La predicción precisa de los eventos de falla en el mantenimiento, ha ido tomando mayor relevancia a lo largo del tiempo. Si bien el mantenimiento preventivo ha mostrado ser eficaz, a la hora de generar tareas de mantenimiento, sus altos costos conllevan una baja significativa en la rentabilidad de los sistemas productivos. El modelo de uso de vida Weibull, diseñado en el año 1951 (Baptista, 2017) ha sido uno de los modelos de falla preferidos dentro del mantenimiento.

## 1.1. GENERALIDADES DEL MODELO

Una de las principales preguntas en ingeniería de mantenimiento es: ¿Cuándo fallarán mis activos críticos? El modelo de uso de vida Weibull, trata de responder a esta interrogante, generando estimaciones dependientes del comportamiento de falla de los equipos. A grandes rasgos, el modelo toma una gran muestra de datos históricos de fallos; de estos obtiene tiempo medio entre fallas (MTBF), mediante α que corresponde al parámetro de escala de la distribución y β, que representa el factor de forma. Ver Figura 1-1 para comprender el comportamiento de parámetro de forma.

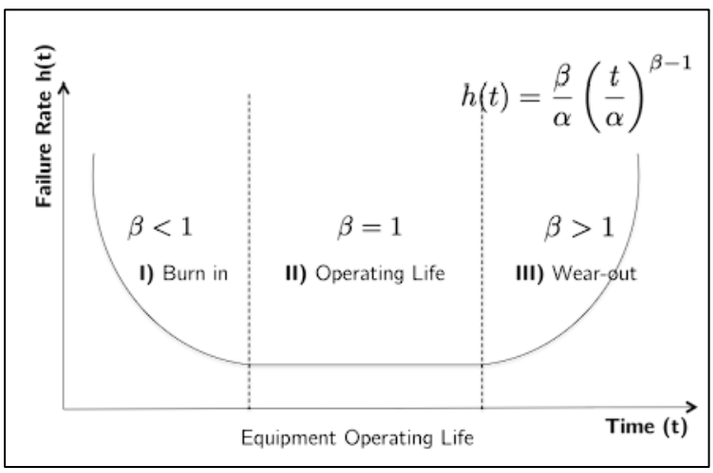


Figura 1-1. Curva de la bañera típica de eventos de fallos

Fuente: Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA Modeling (Baptista, Sankararaman, Medeiros, Nascimento, Prendinger y Henriquesa, 2017)

La Figura 1-1 representa la curva de la bañera, en donde la vida útil de un activo se ve modelada. Cuando β es menor que 1, el equipo se encuentra con una alta tasa de fallo, pero decrece en el tiempo; comúnmente se denomina “etapa de mortalidad infantil”. Cuando β es igual a 1, la tasa de fallo se refleja como constante; lo que indica un periodo de fallas aleatorias. Mientras β sea mayor que 1, la tasa de fallo tiene un comportamiento creciente a través del tiempo; se denomina etapa de desgaste, y generalmente los malos eventos ocurren de manera seguida.

## 1.2.DESARROLLO DEL MODELO DE USO DE VIDA WEIBULL

La base de datos histórica de fallos de un equipo, representa el inicio del desarrollo del modelo de uso de vida Weibull. La selección de esta, hace hincapié en cómo se recopilan y almacenan los datos; generando una selección de datos valiosos. Posteriormente se deben procesar los datos seleccionados, es decir, generar un tratamiento que consiste en regularizar datos faltantes y eliminar valores atípicos.

Weibull define dos parámetros relevantes para su desarrollo: α, β. Con estos se logra obtener la función de densidad de falla, (ecuación 1); la cual consiste en la probabilidad de que un componente o sistema falle dentro de un periodo predefinido como instante delta t. La función acumulada de fallo o probabilidad acumulada de fallo (ecuación 2) consiste en la probabilidad de que un componente o sistema falle dentro de un periodo de tiempo establecido, es decir, que no sobreviva. La confiabilidad (Ecuación 3) se define como la probabilidad de que un componente o sistema esté en buen funcionamiento en un periodo de tiempo dado, bajo condiciones ambientales dadas; dichas condiciones no permiten un mantenimiento genérico a ciertos componentes o sistemas, por lo tanto, los planes de mantenimiento son específicos, entregando gran relevancia a esta variable para la toma de decisiones.

$$f(t) = \frac{\beta}{\alpha} \cdot \left(\frac{t}{\alpha}\right)^{\beta-1} \cdot e^{-\left(\frac{t}{\alpha}\right)^{\beta}} \quad (1)$$

$$F(t) = 1 - e^{-\left(\frac{t}{\alpha}\right)^{\beta}} \quad (2)$$

$$R(t) = e^{-\left(\frac{t}{\alpha}\right)^{\beta}} \quad (3)$$

$$MTBF = \alpha \cdot \Gamma \cdot \left(1 + \frac{1}{\beta}\right) \quad (4)$$

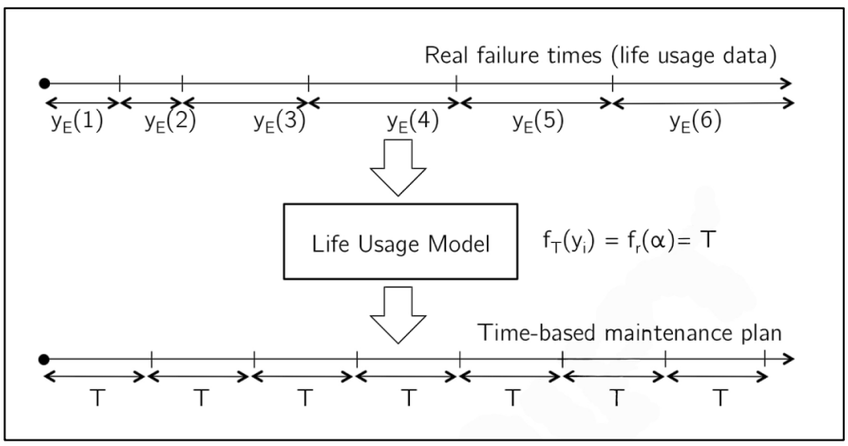


Figura 1-2. Modelo de uso de vida Weibull

Fuente: Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA Modeling (Baptista, 2017) (Thyago P. Carvalho, 2019)

La Figura 1-2 indica los tiempos entre fallas resultantes de un equipo, se aprecia que cada intervalo es distinto al otro. Mediante la distribución Weibull, como dato final obtenido se calcula el tiempo medio entre fallas, es decir, el promedio generalizado de los tiempos de falla del equipo. El modelo de uso de vida Weibull podría generar predicciones erróneas sobre los eventos de falla a futuro al comparar estas diferencias. Si se lleva a mayores intervalos de tiempo, el error podría ser significativo; e inclusive, desestimar la información obtenida con Weibull.

## 1.3.DESARROLLO DEL CASO MEDIANTE MODELO WEIBULL

EL equipo 3050 de la planta de celulosa reúne un total de 369 eventos de falla desde el año 2013, hasta el año 2018, considerando distintas duraciones e intervalos de funcionamiento hasta la ocurrencia del fallo. Esto hace referencia a que los tiempos entre fallas son distintos entre cada uno de los periodos (no son constantes). El modelo Weibull, si bien es uno de los más usados en ingeniería de mantenimiento, predice una media para los tiempos entre falla.

Antes de ejecutar el modelamiento de datos en base a Machine Learning es importante introducir algunos conceptos sobre el diseño de un modelo (Thyago P. Carvalho, 2019). Estos conceptos hacen referencia al procedimiento de tratamiento de datos mediante herramientas estadísticas y de machine learning.

El procedimiento es:

Paso 1: selección de datos.
Paso 2: Procesamiento de histórico de datos.
Paso 3: Selección de modelos.
Paso 4: Entrenamiento de modelos.
Paso 5: Validación de modelos.

Para la realización del modelamiento de uso de vida Weibull, se decidió abarcar los pasos uno y dos expuestos anteriormente. El paso número uno consiste en la selección de una base de datos relevante para el análisis, considerando el impacto del activo en la organización. Para este caso, la planta de celulosa proporciona información sobre equipos que consideran críticos dentro de su proceso productivo. Dentro del paso dos: Procesamiento de histórico de datos, se identifica la transformación de la base de datos. Esto incluye la normalización, tratamiento de datos faltantes y eliminación de valores atípicos (valores iguales a cero). En cuanto a la relevancia de este paso, cualquier dato atípico provocará un error en el análisis del modelamiento.

### 1.3.1. Base de datos

La selección de la base de datos a analizar consiste en siete variables: Área, Causa, Fecha del evento, Equipo, duración del evento en minutos, duración del evento en horas y TBF, que representa el tiempo de funcionamiento hasta la falla.

Como software de apoyo para la obtención de información estadística y grafica se emplea Python, el cual, solo diseñando un algoritmo simple, es capaz de entregar la información deseada. Para el cálculo de MTBF se modificó la base de datos considerando solo la información relevante para el software. Se generaron dos columnas, una con los tiempos hasta la falla o TBF, y otra con el ranking global de eventos o EVENTOS. Los datos representan los eventos de falla ocurridos entre el abril del 2013 y julio 2018.

| Area | Causa | Fecha | Equipos | Total (minutos) | Total (horas) | TBF |
|---|---|---|---|---|---|---|
| Mec | Atochamiento sector decarga | 26-04-2013 | 3050 | 35 | 0,58 | |
| Mec | Cambio Polín de Carga | 08-05-2013 | 3050 | 11 | 0,18 | 288 |
| Mec | Otros | 06-07-2013 | 3050 | 60 | 1,00 | 1416 |
| Mec | Otros | 14-08-2013 | 3050 | 125 | 2,08 | 936 |
| Mec | Otros | 15-08-2013 | 3050 | 240 | 4,00 | 24 |
| Mec | Activación ZS | 30-11-2013 | 3050 | 60 | 1,00 | 2568 |
| Mec | Metal | 20-03-2014 | 3050 | 42 | 0,70 | 2640 |
| Mec | Metal | 24-03-2014 | 3050 | 35 | 0,58 | 96 |
| Mec | Metal | 25-03-2014 | 3050 | 20 | 0,33 | 24 |
| Mec | Metal | 26-03-2014 | 3050 | 34 | 0,57 | 24 |
| Mec | Metal | 27-03-2014 | 3050 | 15 | 0,25 | 24 |
| Mec | Metal | 28-03-2014 | 3050 | 26 | 0,43 | 24 |
| Mec | Metal | 01-04-2014 | 3050 | 56 | 0,93 | 96 |
| Mec | Metal | 02-04-2014 | 3050 | 49 | 0,82 | 24 |
| Mec | Metal | 03-04-2014 | 3050 | 34 | 0,57 | 24 |
| Mec | Metal | 04-04-2014 | 3050 | 63 | 1,05 | 24 |

Figura 1-3. Fragmento de base de datos equipo 3050

Fuente: Elaboración propia, basado en información entregada por Planta Celulosa

Se puede apreciar en la Figura 1-3 la existencia de valores igual a cero. Desarrollando el paso dos del procedimiento explicado con anterioridad, se calculará un promedio entre el valor anterior y posterior a la ubicación del cero, luego se reemplazará el dato cero con el valor calculado.

El comportamiento de los tiempos hasta la falla se representa gráficamente en la Figura 1-4, demostrando picos de tiempo al inicio y centro de la curva. Esto podría demostrar que las intervenciones de mantenimiento causan un aumento en los intervalos entre falla.

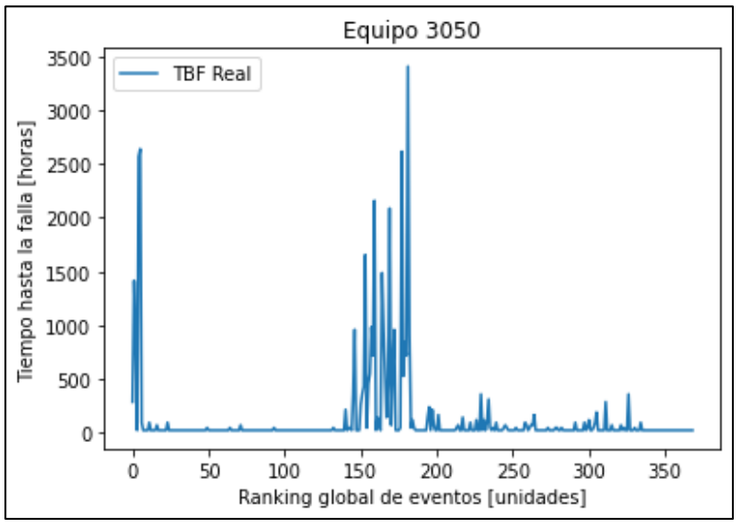


Figura 1-4. Gráfica de tiempos hasta la falla

Fuente: Elaboración propia en Python

### 1.3.2. Obtención de parámetros Weibull mediante metodología mínimos cuadrados

Para la obtención de los parámetros Weibull se utiliza el software Python, y mediante el método de mínimos cuadrados o método gráfico. Esta metodología emplea la función de distribución acumulada; los registros de eventos son interpolados por una recta. Para dicho caso se emplea la ecuación 5, que no es más que una representación de recta [y = a + bx] y la ecuación 6 que representa una recta con pendiente e intersección con el eje y.

$$y = \mathrm{Ln}\left[-Ln\left(1 - F(t)\right)\right] \quad (5)$$

$$x = Ln(TBF) \quad (6)$$

Aplicando el concepto a la obtención de parámetros Weibull, como se muestra en las ecuaciones 7, 8 y 9.

$$Ln[-\left(Ln\left(R(t)\right)\right) = -\beta \cdot Ln(\alpha) + \beta \cdot Ln(\beta) \quad (7)$$

$$a = -\beta \cdot Ln(\alpha) \quad (8)$$

$$bx = \beta \cdot Ln(\beta) \quad (9)$$

Aplicando los arreglos matemáticos, los parámetros Weibull quedan definidos en las ecuaciones 10 y 11.

$$\beta = b = pendiente \quad (10)$$

$$\alpha = e^{-\left(\frac{\alpha}{\beta}\right)} \quad (11)$$

Los parámetros Alfa (α) y Beta (β) para la base histórica de fallos quedan definidos en la Figura 1-5.

| Alpha (α) | Beta (β) |
|---|---|
| 88,3 | 0,7 |

Figura 1-5. Parámetros Alfa y Beta

Fuente: Elaboración propia mediante Excel

Esto demuestra que el activo se encuentra en una etapa de rodaje o mortalidad infantil, ya que el parámetro beta (β) es menor a uno [1].

### 1.3.3. Obtención del tiempo medio entre falla (MTBF)

La obtención del tiempo medio entre fallas (en adelante MTBF) consiste en la aplicación de los parámetros de ajuste Weibull. Mediante este valor se entrega un tiempo promedio en el cual el activo falla, es decir, el intervalo entre eventos. Para determinar el MTBF se emplea la ecuación 12.

$$MTBF = \alpha \cdot \Gamma \cdot \left(1 + \left(\frac{1}{\beta}\right)\right) \quad (12)$$

El valor obtenido considera intervalos constantes entre los eventos de falla, lo cual no se asemeja a la realidad del activo sometido al análisis. En la Figura 1-6 se representa el valor obtenido para el histórico de fallo.

| MTBF |
|---|
| 111,3 |

Figura 1-6. Tiempo medio entre fallas de histórico general de eventos

Fuente: Elaboración propia mediante modelo de uso de vida Weibull

## 1.4. METODOLOGÍA DE ANÁLISIS

Como metodología para el desarrollo de los modelos dentro de la presente investigación se seguirá el procedimiento presentado a continuación:

a. Desarrollo del histórico de fallos a través del modelo en análisis.
b. Obtención de predicciones de eventos de falla a través del modelo.
c. Determinación de la raíz del error cuadrático medio.
d. Cálculo el error absoluto en cada predicción, analizarlos y compararlos con el error absoluto medio de la muestra.
e. Obtención del error absoluto medio por intervalo estándar establecido (este paso solo se llevará a cabo si la dimensión de los datos lo permite).
f. Cálculo del error porcentual para cada dato predicho y compararlo con el error porcentual medio de la muestra.

### 1.4.1. Raíz del error cuadrático medio

Como parámetro de validación se utilizará el error cuadrático medio, conocido por sus siglas RMSE (Root Mean Squared Error), se obtiene mediante la ecuación 13.

$$RMSE = \frac{1}{n} \cdot \sqrt{\sum (Y - \mathrm{y})^2} \qquad (13)$$

El error cuadrático medio representa la desviación estándar de los errores de predicción. Los valores residuales representan la distancia entre los datos reales, y los obtenidos mediante el modelo ejecutado; es decir, la dispersión de los valores entregados.

### 1.4.2. Error absoluto y error absoluto medio

Para el análisis de los datos obtenidos de forma unitaria se empleará el error absoluto, y para la medición por intervalo generado se utilizará la desviación absoluta media. Véase la ecuación 14 para error absoluto y ecuación 15 para la desviación absoluta media.

$$EA = Dato\ Real - Predicción \qquad (14)$$

$$MAD = \frac{\sum |dato\ real - pronóstico|}{cantidad\ de\ datos} \qquad (15)$$

### 1.4.3. Error de porcentual y error porcentual medio absoluto

El error porcentual corresponde al cociente entre el error absoluto y el dato real dentro de la muestra. Representa un indicador de calidad del dato obtenido; a menor error de porcentaje, mejor es la predicción obtenida. La ecuación 17 representa el error porcentual.

El error porcentual medio absoluto (MAPE de ahora en adelante por sus siglas en inglés) corresponde a una medida de error relativa en la que se emplean valores absolutos. Permite determinar la calidad de los valores obtenidos en las predicciones a través de la media de estos. La ecuación 16 representa esta métrica.

$$\text{MAPE} = \frac{1}{n} \cdot \sum_{t=1}^{n} \left| \frac{dato\ real - pronóstico}{dato\ real} \right| \tag{16}$$

$$EP = \left| \frac{dato\ real - pronóstico}{dato\ real} \right| \tag{17}$$

## 1.5. DESARROLLO DEL CASO MEDIANTE WEIBULL AJUSTADO

Como metodología para proyectar los tiempos entregados por el modelo Weibull, se agrupan los datos de “Tiempos hasta la falla” en 8 valores inicialmente; con esto se calcula cuando ocurrirá el evento nueve (en horas). Una vez obtenido el valor, se suma el noveno tiempo real de funcionamiento con el fin de determinar cuándo sucederá el décimo. Siguiendo este ciclo repetitivo se logra proyectar cada evento de falla con todos los datos, y así verificar el error que entrega el modelo de uso de vida. En la Figura 1-7 se puede apreciar gráficamente los valores obtenidos mediante el modelo y los tiempos reales hasta la falla.

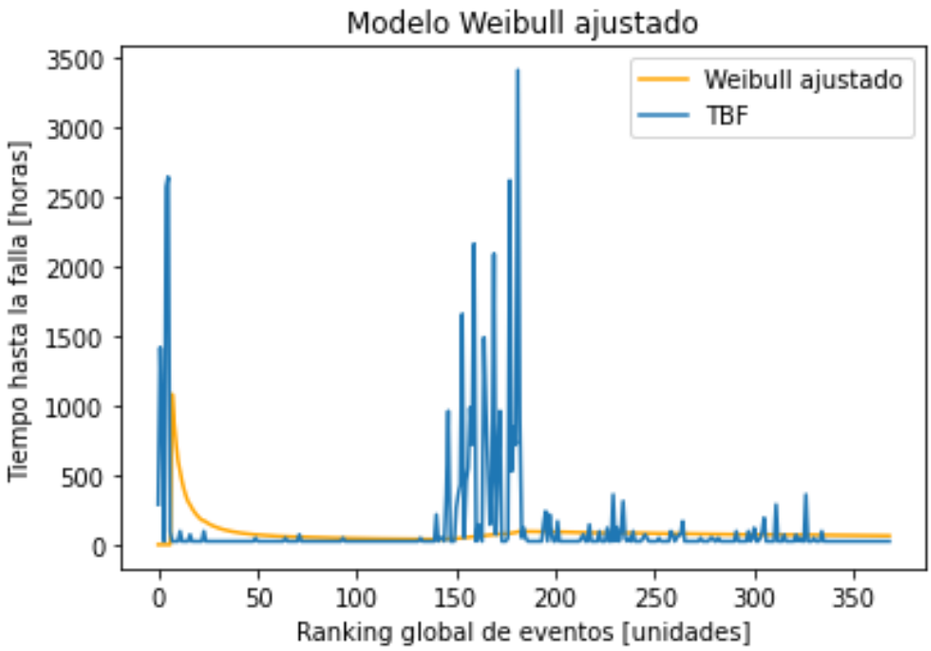


Figura 1-7. Gráfica de tiempo real hasta la falla y predicción de modelo Weibull

Fuente: Elaboración propia en Python

Para ver los resultados, ver subsección 1.5.1.

### 1.5.1. Resultados del modelo Weibull ajustado

La aplicación del modelo de uso de vida Weibull ajustado representa menores errores en relación con la aplicación tradicional de la metodología.

El parámetro beta (β) tuvo un comportamiento en un rango de 0,61 a 1,12; con un total de 369 datos, esto quiere decir que el equipo se encuentra en una etapa de rodaje o mortalidad infantil al inicio de la curva, a medida que el tiempo avanza, los eventos ocurren con una mayor frecuencia, aumentando el grado del parámetro beta. Con anterioridad se mencionó que el activo se encontraba en una fase de rodaje, según modelo Weibull general se corrobora la etapa en la que se encuentra el activo (ver subsección 1.3.2); cabe destacar que los valores predichos por Weibull ajustado se asemejan a la realidad, por lo tanto, se tomará como etapa real los valores de beta (β) definidos en Weibull ajustado (etapa de rodaje).

Para el caso general total de los datos estimados, el valor de RMSE corresponde a 342,39. Se calcula que existen 312 datos menores que el promedio general del error absoluto, y 49 valores mayores que este indicador. La media corresponde a 123,71 horas. En la Figura 1-8 se aprecia el error absoluto de cada dato predicho por el modelo Weibull.

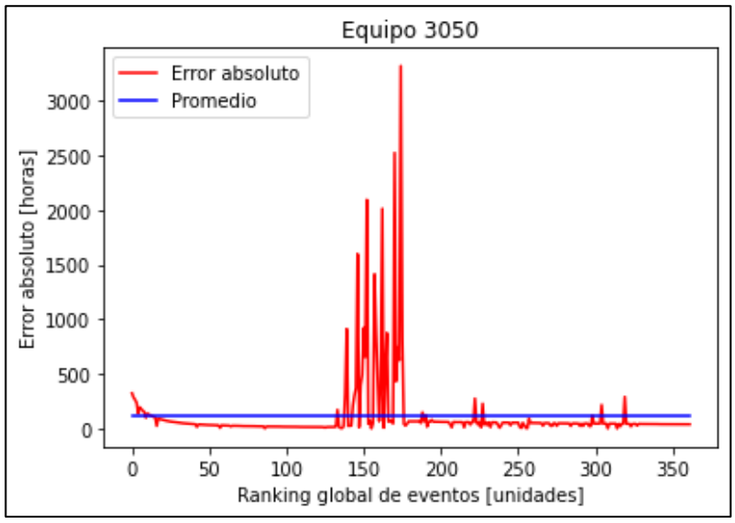


Figura 1-8. Error absoluto en la predicción de eventos con modelo Weibull

Fuente: Elaboración propia mediante Python

El máximo valor de error absoluto queda definido como 3320 horas; llevándolo a gran escala, representa un desacierto en la predicción del evento de casi 5 meses. Por otro lado, el mínimo error se diferencia en 1 horas con respecto al tiempo real.

Se genera un análisis entre intervalos de tiempo, promediando los errores absolutos con el fin de conocer en que rango de este se producen los mayores desaciertos en la predicción. Los resultados se pueden apreciar en la Figura 1-9 y gráficamente se representan en la Figura 1-10.

| Intervalo de analisis [hrs] | Media error [hrs] | Tamaño muestra [unidades] |
|---|---|---|
| Entre 0 y 500 horas | 54 | 344 |
| Entre 500 y 1000 horas | 694 | 12 |
| Entre 1000 y 1500 horas | 1412 | 1 |
| Mayor a 1500 horas | 2307 | 5 |

Figura 1-9. Promedios de error absoluto entre intervalos de análisis

Fuente: Diseño propio mediante Excel

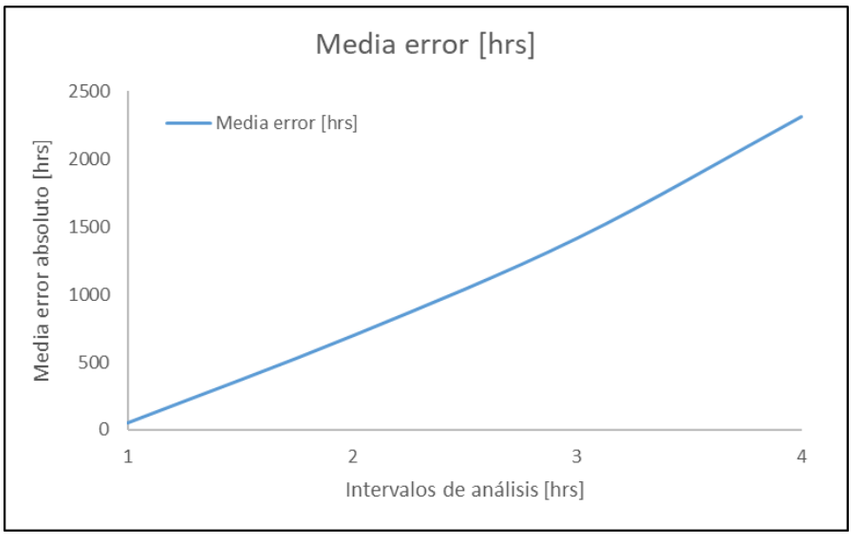


Figura 1-10. Gráfica de comportamiento media error absoluto

Fuente: Diseño propio mediante Python

El error porcentual medio corresponde a 215,30%, mientras que el máximo error es de un 4399,14%; el mínimo error de los valores obtenidos se define como un 1,52%. Existen 105 valores obtenidos en que su error es sobre la media, mientras que bajo la media son 256.

# CAPÍTULO 2: DESARROLLO DE REDES NEURONALES ARTIFICIALES CON BASE EN MODELO ARIMA

# 1. MODELO AUTORREGRESIVO DE MEDIA MOVIL

Se puede definir como un modelo estadístico para series temporales de datos en donde existe una dependencia entre estos. Esto quiere decir que el modelamiento de la data en el futuro se hace en función de valores pasados o existentes. Intenta encontrar patrones dentro de la serie de datos con el fin de obtener el comportamiento futuro de estos.

## 1.1. GENERALIDADES DEL MODELO

El acrónimo ARIMA se expresa de la unión de sus tres componentes: AR, derivado de la palabra autorregresivo; I, de integrado y MA, que hace referencia a medias móviles. Sus parámetros se delimitan como p, d, q.

Para el desarrollo del modelo, se aplicará un modelo ARMA(p,d,q)

- Autorregresiva (AR): Asume que el valor de la serie tiempo en “t” corresponde a una combinación lineal de valores perteneciente a los “p” periodos anteriores (Christiansen, 2020)

- Integrada (I): Aplica la diferenciación en los datos con el objetivo de obtener una serie estacionaria. El parámetro “d” corresponde a la cantidad de diferenciaciones realizadas (Christiansen, 2020)

- Medias móviles (MO): Asume que el error de regresión observado corresponde a una combinación lineal de “q” errores aleatorios previos (Christiansen, 2020).

## 1.2. DESARROLLO DEL CASO MEDIANTE MODELO ARIMA

Para lograr el desarrollo deseado del modelo. es necesario seguir un proceso sistemático de trabajo, el cual se verá explicado a continuación.

a. Se debe estabilizar la varianza muestral, eliminando la tendencia y estacionalidad en los datos, con el fin de obtener una serie estacionaria. Se pueden utilizar gráficas de función de autocorrelación y función de autocorrelación parcial para determinar “p” y “q”
b. Una vez obtenida la serie estacionaria se debe estimar un modelo capaz de expresar la estructura de correlación de esta.
c. Para la expresión obtenida en la parte b, se debe determinar la variabilidad, tendencia y estacionalidad de los datos originales.
d. El modelo obtenido se debe validar a través de las correlaciones de los valores residuales.

En la Figura 1-1 se expresa de manera más sencilla los puntos antes descritos.

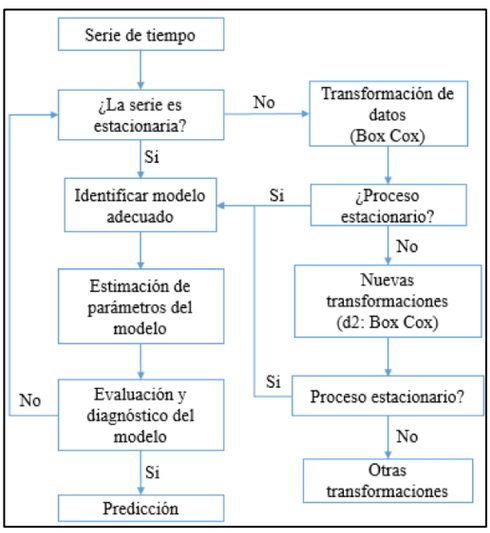


Figura 1-1. Diagrama de flujo para modelo ARIMA(p,d,q)

Fuente: (Uribe, Ortiz, Romero, & Valbuena, 2017)

### 1.2.1. Prueba de Dickey Fuller

La prueba de Dickey Fuller indica si la serie de datos no es estacionaria, decidiendo si es que es necesario diferenciarla. Es decir, ***d*** = *0*.

Consiste en que, si el valor de ***p*** es menor que el nivel de significancia 5% o 0,05, la serie se cataloga como estacionaria

$$H0 > 5\% = Serie\ no\ es\ estacionaria$$

$$H0 < 5\% = Serie\ estacionaria$$

Mediante el software Python se logra aplicar la prueba de Dickey Fuller, obteniendo como resultado que el valor de ***p*** representa un 1%. Por lo tanto, no se deberá realizar una diferenciación entre los datos, ya que la serie es estacionaria. De esta metodología se discierne el parámetro ***d***, el cual corresponde a la cantidad de diferenciaciones que se tuvieron que realizar para que la serie tuviese un comportamiento estacionario. Para este caso, es 0.

El cálculo de la diferenciación se determina mediante la fórmula:

$$Dif = y - y_{-1} \tag{18}$$

### 1.2.2. Orden del término AR(p)

Se debe establecer si existe la necesidad de que el modelo requiera un término ***AR***. Una forma de hacerlo es verificando el grafico de autocorrelación parcial; para el análisis arroja valores incorrectos respecto a los resultados esperados. Se emplea un algoritmo de iteración que modifica los parámetros P y Q requeridos y entrega como resultado aquel que optimiza el proceso de predicción.

Durante el proceso de iteración mediante el software Python, se determina que el valor óptimo para el termino p es 1.

La autocorrelación parcial corresponde a la correlación parcial entre la serie y su error.

La autocorrelación parcial para del error ***k*** queda definida como:

$$Y(t) = \alpha 0 + \alpha 1 \cdot Yt - 1 + \alpha 2 \cdot Yt - 2 + \alpha 3 \cdot Yt - 3 \tag{19}$$

### 1.2.3. Ordene del término MA(q)

Se debe establecer si existe la necesidad de que el modelo requiera un término ***MA***. De acuerdo con el algoritmo de iteración, el valor óptimo para el término MA es 2. Este valor indica cuantos términos de MA son necesarios para borrar cualquier autocorrelación en la data en calidad estacionaria.

### 1.2.4. Prueba de Ljung – Box

La prueba de Ljung – Box demuestra el comportamiento del error. Si Ho > 0,05, indica una media igual a cero y la existencia de ruido blanco, por ende, el modelo se ajusta bien. Si Ho < 0,05, existe una varianza constante y no presenta ruido blanco, es decir, el

modelo no se ajusta bien. Para el caso del modelo ARIMA 1.0.2, la prueba de Ljung – Box da como resultado un 0,1, es decir que la media tiende a cero y existe ruido blanco.

### 1.2.5. Medición de Curtosis

El coeficiente de Curtosis indica el grado de amplitud que posee la curva de frecuencia en relación con la concentración de datos. Como interpretación general, si es mayor que 1 corresponde a una curva leptocúrtica, si es menor que 1 representa una curva platicúrtica, y si es igual a cero se define como mesocúrtica. Para el modelo ARIMA 1.0.2, el valor estimado es de 20,60; esto se puede interpretar como una curva leptocúrtica ya que es notablemente mayor que 1.

Mediante el gráfico de los valores residuales del modelo respecto a los datos reales, queda expresado gráficamente en la Figura 1-2.

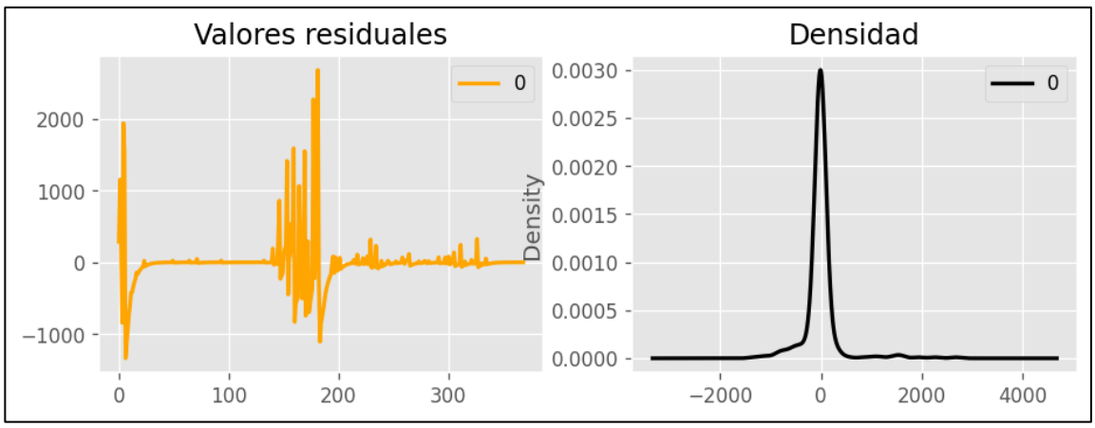


Figura 1-2. Curvas de valores residuales y densidad de datos modelo ARIMA 1.0.2

Fuente: Diseño propio elaborado mediante Python

### 1.2.6. Resultados del modelo ARIMA (1.0.2)

Una vez desarrollado el modelo y obtenidos todos los parámetros que determinan su desempeño en el caso de análisis, se procede a realizar la predicción de los tiempos de funcionamiento obtenidos mediante ARIMA (1.0.2).

El valor de RMSE corresponde a 337,8, el cual es menor en relación con el entregado por Weibull ajustado. A continuación, en la Figura 1-3 se representa

gráficamente las predicciones del modelo indicado con respecto a los tiempos de funcionamiento hasta la falla.

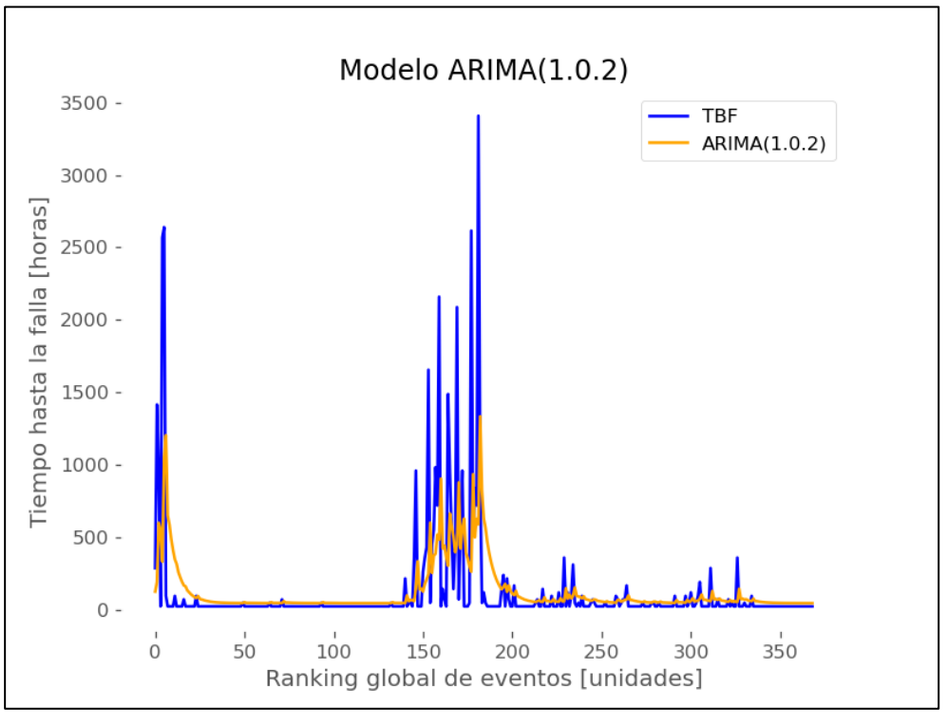


Figura 1-3. Gráfica de tiempo real hasta la falla y predicción modelo ARIMA(1.0.2)

Fuente: Diseño propio mediante Python

Al igual que los resultados obtenidos mediante el modelo de uso de vida Weibull, se demuestra una mayor amplitud en el error para aquellos valores mayores a 1000 horas. Se determinan los errores absolutos para cada tiempo dentro de la base de datos, dando como resultado la existencia 302 datos que se encuentran bajo el promedio del error absoluto, mientras que 67 valores están por sobre este indicador. La media es definida como 129,24 horas. El error máximo obtenido es 2818 horas, y el mínimo 2 horas. En la Figura 1-4 demuestra gráficamente la amplitud de los errores obtenidos por cada dato pronosticado.

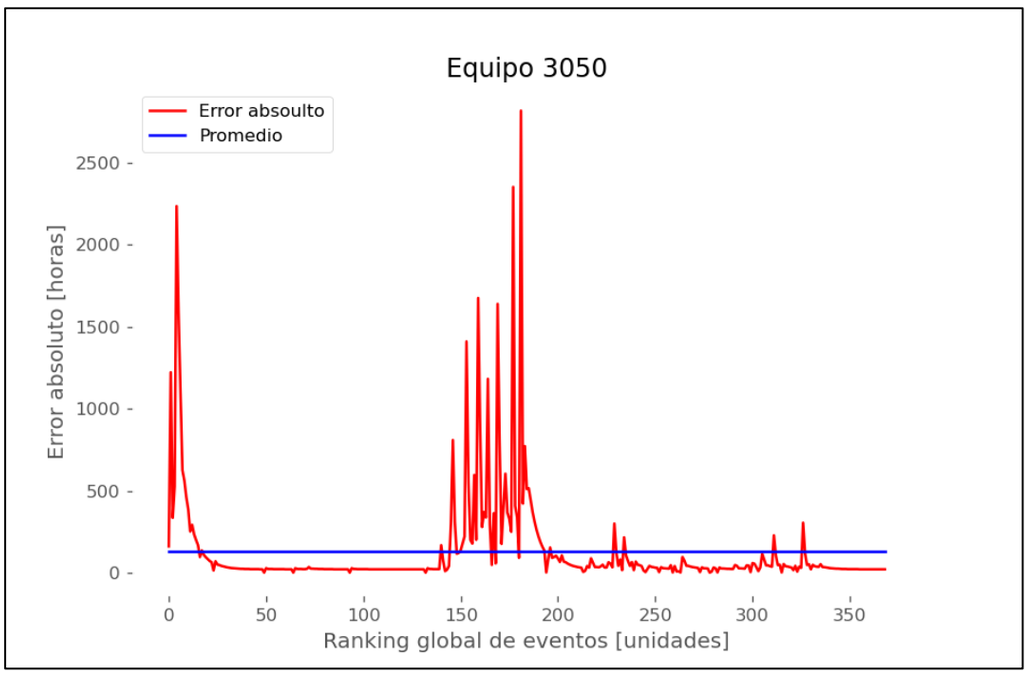


Figura 1-4. Error absoluto en datos con modelo ARIMA

Fuente: Elaboración propia mediante Python

Se calcula la media del error absoluto por intervalo de tiempos de funcionamiento; los resultados quedan expresados en la Figura 1-5:

| Intervalo de análisis [hrs] | Media error [hrs] | Tamaño muestra [unidades] |
|---|---|---|
| Entre 0 y 500 horas | 81,6 | 346 |
| Entre 500 y 1000 horas | 256,8 | 13 |
| Entre 1000 y 1500 horas | 1106,5 | 2 |
| Mayor a 1500 horas | 1852,9 | 7 |

Figura 1-5. Promedios de error absoluto entre intervalos de análisis

Fuente: Diseño propio elaborado mediante Excel

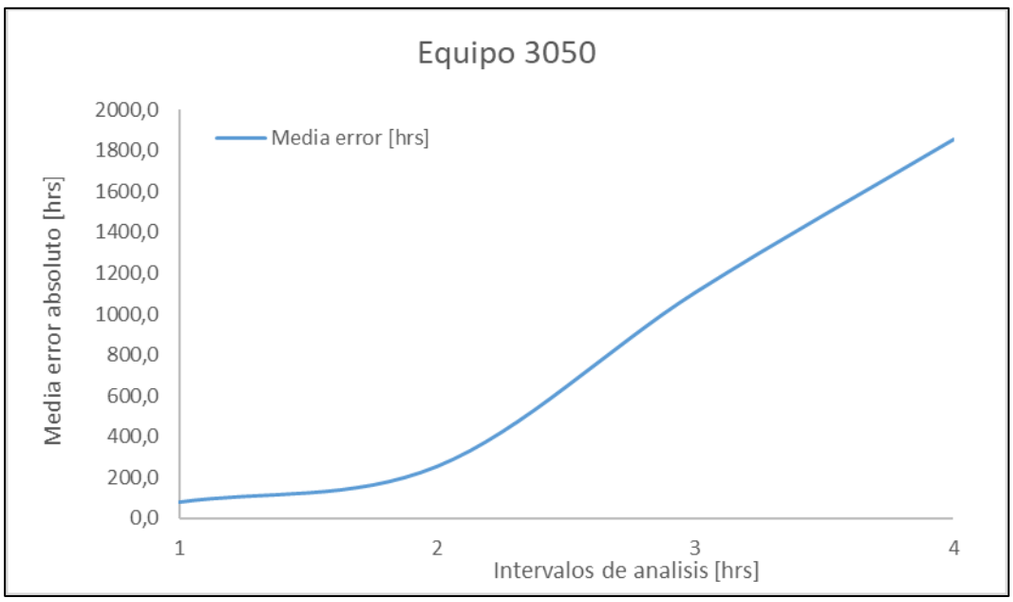


Figura 1-6. Gráfica comportamiento media error absoluto

Fuente: Diseño propio mediante Python

Para este caso, desde el segundo intervalo de análisis se presenta una clara tendencia al aumento del error absoluto; en donde el máximo lo obtiene en aquellos datos mayores a 1500 horas, mientras que el mínimo error se presenta entre 0 y 500 horas.

El error porcentual medio corresponde a 228,97%, mientras que el máximo error es de un 3670,83%; el mínimo error de los valores obtenidos se define como un 2,08%. Existen 62 valores obtenidos en que su error es sobre la media, mientras que bajo la media son 307 datos.

## 1.3.REDES NEURONALES ARTIFICIALES

Las redes neuronales artificiales son técnicas computacionales inteligentes, las cuales hacen referencia a neuronas biológicas. La red en si está compuesta por una serie de nodos o neuronas, que si se analizan de manera individual tienen una metodología de funcionamiento sencilla. Se conectan con otras neuronas a través de canales, los cuales tienen un peso asignado (interfiere directamente con el resultado).

Existe una serie de definiciones para denominar a una red neuronal, pero aquella que las representa con mayor exactitud sería… una red neuronal se puede definir como un sistema que permite establecer una relación entre entradas y salidas inspiradas en el

sistema nervioso y diferenciándose de la computación tradicional, ya que estos no utilizan una algoritmia secuencial. Las redes neuronales artificiales se comportan como un cerebro humano, en donde se procesa la información en paralelo, con la posibilidad de aprender y generalizar situaciones no incluidas en procesos de entrenamiento (Serna, 2017).

La constitución general de una neurona viene dada por una o más entradas, llamadas dendritas; el cuerpo y una salida denominada como axón. Su similitud con una neurona biológica se puede ver en la Figura 1-7.


Figura 1-7. Neurona biológica y neurona artificial

Fuente: Redes neuronales Conceptos básicos y aplicaciones (Matich, 2001)

### 1.3.1. Neuronas biológicas

La neurona es una clasificación de célula que tiene como función la transmisión de información a través de impulsos nerviosos, los cuales son de tipo químico y eléctrico. La velocidad de transmisión de información queda limitada a 120 m/s, en el caso de las neuronas con mayor tamaño.

Conforman una extensa red a través de todo el cuerpo, siendo ésta el medio transmisor de un impulso nervioso en forma de mensaje químico y eléctrico. El impulso viaja en un mismo sentido a través de toda la red neuronal. Si se ve de una manera macro, el impulso llega a la neurona a través de las dendritas (entradas), se procesa en el soma o núcleo del nodo y luego sale a través del axón (salida) con dirección hacia otra neurona, repitiendo el proceso en esta otra.

Las interacciones entre las neuronas no involucran el contacto directo entre ellas, más bien se encuentran separadas por un espacio llamado espacio sináptico. El impulso nervioso al llegar al extremo del axón, se liberan neurotransmisores a la sinapsis (espacio sináptico), convirtiendo la señal eléctrica en una señal química, la cual entra a través de la dendrita en otra neurona; es decir, el impulso entra en forma de señal eléctrica y sale como

señal química. En la Figura 1-8 se aprecia la forma de una neurona y el espacio sináptico en donde ocurre la interacción.

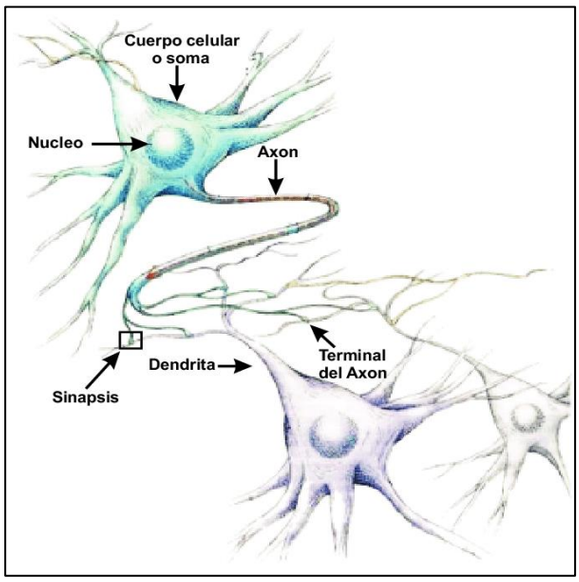


Figura 1-8. Interacción entre neuronas biológicas

Fuente: (Cedeño, 2022)

### 1.3.2. Elementos que componen una red neuronal artificial

La composición de una red neuronal artificial se caracteriza por tener una capa de entrada, en donde se ubican los nodos (la cantidad de neuronas depende del resultado deseado) que reciben los datos de interés, cabe destacar que estos valores provienen desde el exterior de la red; luego viene la capa oculta, la cual puede estar constituida por una o más capas de neuronas, estas no tienen contacto con el exterior de la red, y pueden estar conectadas de distintas maneras, definiendo así el tipo de red neuronal; al final se encuentra la capa de salida, la que se encarga de entregar los valores deseados.

El funcionamiento se basa en la recepción de los parámetros de entrada por parte de la neurona artificial, se realiza una ponderación de los datos a través del peso, el cual puede activar o desactivar el nodo; se ejecuta una suma ponderada de las entradas, si el valor obtenido es mayor al umbral matemático, esta se activa y entrega una salida de datos.

En la Figura 1-9 se puede apreciar una red neuronal artificial con una capa de entrada, dos capas ocultas y una capa de salida.

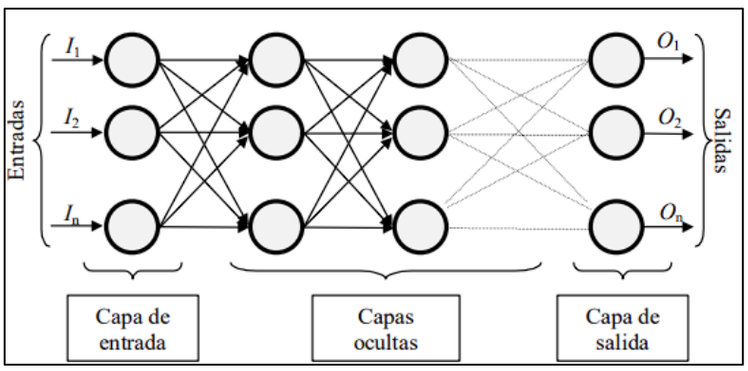


Figura 1-9. Ejemplo representativo de una red neuronal artificial

Fuente: Redes neuronales Conceptos básicos y aplicaciones (Matich, 2001)

### 1.3.3. Tipos de redes neuronales artificiales

La principal diferencia entre los tipos de redes neuronales artificiales es la forma en que están hechas las conexiones de estas. Se pueden clasificar en:

a. Redes de propagación hacia adelante (feed – foward): el flujo de la información a través de la neurona es hacia adelante, extendiéndose por las capas; se caracteriza por no poseer conexiones de retroalimentación.

b. Redes recurrentes: A diferencia de las redes de propagación hacia adelante, las redes recurrentes poseen conexiones de retroalimentación, lo que ayuda a la evolución y estabilidad de la red. Estabilidad hace referencia a que no haya cambios en el estado de activación de las neuronas.

### 1.3.4. Funciones de entrada

La neurona artificial trata a muchos valores de entrada como si fueran uno solo; tal proceso se conoce como entrada global. La problemática surge cuando se quieren combinar las entradas simples dentro de la entrada global.

La función entrada (ecuación 20) se encarga de resolver la problemática presentada con anterioridad.

$$Input = (in_{i1} \cdot W_{i1}) * (in_{i2} \cdot W_{i2}) * (in_{in} W_{in}) \quad (20)$$

El operador * representa la operación adecuada. Dependiendo del caso podría ser sumatoria, productoria, máximo, mínimo, etc. n representa al número de entradas en la neurona y Wi Ni al peso.

Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños (Matich, 2001).

Algunas funciones de entrada son:

a. Sumatoria de las entradas pesadas: Corresponde a la suma de todos los valores de ingreso en la neurona, multiplicados por sus pesos (ecuación 21).

$$\sum_j (n_{ij} w_{ij}), \ \ con\ j = 1,2,3, \dots, n \tag{21}$$

b. Productoria: Corresponde al producto de todos los valores que entran a la neurona, multiplicados por sus pesos (ecuación 22).

$$\prod_j (n_{ij} w_{ij}), \ \ con\ j = 1,2,3, \dots, n \tag{22}$$

c. Máximo: Corresponde al máximo valor de entrada, es decir, el más (Ponce Cruz, 2010)fuerte; multiplicado por su peso correspondiente (ecuación 23).

$$Max_j (n_{ij} w_{ij}), \ \ con\ j = 1,2,3, \dots, n \tag{23}$$

### 1.3.5. Funciones de activación

Como definición se puede decir que la función de activación es la regla que logra establecer el efecto de la entrada total *u(t)* en la activación de la unidad *k* (Ponce Cruz, 2010)

Las neuronas artificiales, pueden tener diferentes estados de activación, algunas se limitan a solo dos, pero otras pueden definir cualquier valor dentro de un grupo de datos.

El estado de actividad de una neurona se determina mediante la función de activación de esta; convirtiendo la entrada global en un número de activación. El rango queda definido de ***0 a 1*** o de ***-1 a 1,*** en donde el ***0*** representa la inactividad para el primer caso, al igual que ***-1*** en el segundo caso; el ***1*** define a la neurona como activa.

La función de activación se define a grandes rasgos como la función de entrada menos el umbral.

Algunas de las funciones de activación más conocidas son:

a. Función lineal: Generalmente utilizada cuando el resultado deseado en la salida es una regresión lineal; es decir, cuando ***a = 1***, la salida es igual a la entrada. La ecuación 24 representa una función de activación lineal, y la figura 2-12 es su representación gráfica en el intervalo ***-1 a 1*** para ordenadas y abscisas. La Figura 1-10 muestra como es la gráfica de la función lineal.

$$F(x) = a \cdot x \tag{24}$$

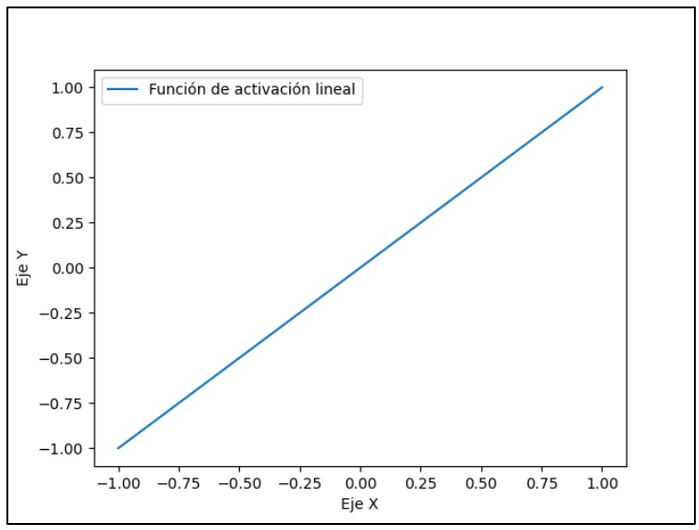


Figura 1-10. Representación gráfica de una función de activación lineal

Fuente: Elaboración propia mediante Python

b. Función escalón: Este tipo de función es asociada a neuronas binarias, es decir, que solo pueden tomar dos valores. Cuando la sumatoria de la entrada global es mayor o igual al umbral, el valor de activación es ***1***; cuando es menor, es ***-1 o 0***. La Figura 1-11 demuestra la gráfica de una función tipo escalón, y la ecuación 25 la representa matemáticamente.

$$F_k(x) = \begin{cases} 1; x \geq 0 \\ 0; x < 0 \end{cases} \qquad F_k(x) = \begin{cases} 1; x \geq 0 \\ -1; x < 0 \end{cases} \tag{25}$$

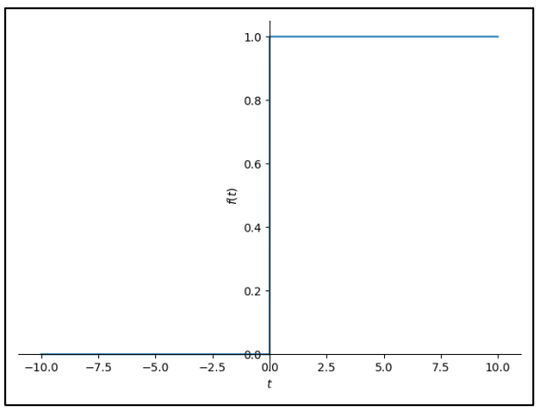


Figura 1-11. Representación gráfica de función de activación escalón

Fuente: Diseño propio mediante Python

c. Función tangente hiperbólica: Generalmente su uso es para situaciones en donde las variaciones en valores positivos y negativos de la señal son suaves. Se utiliza principalmente en casos de entrenamiento supervisado (en la sección 1.4 se explicará la fase de entrenamiento de un modelo de RNA). En la Figura 1-12 se puede ver la representación gráfica de la función. La fórmula queda expresada en la ecuación 26.

$$F_k(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}} \tag{26}$$

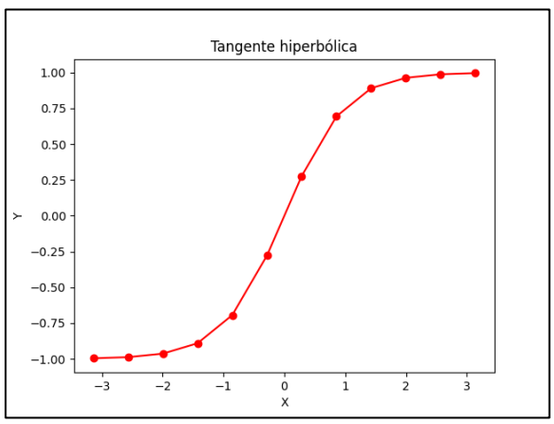


Figura 1-12. Representación gráfica de función de activación tangente hiperbólica

Fuente: Elaboración propia mediante docs.scipy.org

d. Función sigmoidal: Los valores de salida que entrega la función comprenden el rango desde 0 a 1; aquellos valores cercanos a 1 tienden a ser asintóticos, al igual que los cercanos a 0. En la Figura 1-13 se puede apreciar la representación gráfica de la función sigmoide. La ecuación 27 representa esta función.

$$F_k(u) = \frac{1}{1+e^{-u}} \quad (27)$$

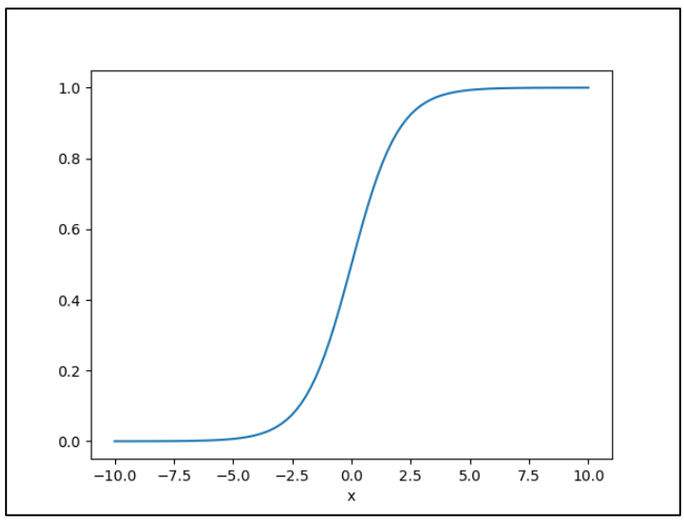


Figura 1-13. Representación gráfica de función de activación sigmoide

Fuente: Elaboración propia mediante Acervolima

## 1.4.ENTRENAMIENTO DE UNA RED NEURONAL

Se denomina entrenamiento al proceso de configuración de una red neuronal para que las entradas produzcan las salidas deseadas a través del fortalecimiento de las conexiones (Ponce Cruz, 2010).

Una de las formas características de lograr esto es entregándole los pesos a las conexiones; otra metodología conlleva utilizar formas de retroalimentación para lograr el aprendizaje automático de los pesos y adaptación de estos por parte de la red.

Se pueden definir dos tipos de aprendizaje: aprendizaje supervisado y aprendizaje no supervisado. Para el caso del presente trabajo de título, se interiorizará en aprendizaje supervisado. La característica principal de este tipo de aprendizaje es que la fase de

entrenamiento es controlada por un supervisor, el cual entrega la entrada al algoritmo y conoce la salida que este debe determinar. En el caso que la red no entregue los valores o decisiones deseadas, el supervisor puede variar los pesos para obtener los resultados deseados.

Se pueden definir tres tipos de aprendizajes supervisados:

- Aprendizaje por corrección de error: Este método de aprendizaje consiste en ajustar los pesos de las conexiones en función del error entre los valores de entrada y los de salida.
- Aprendizaje por refuerzo: Es un tipo de aprendizaje supervisado más lento; se basa en no disponer de un valor concreto al que se quiere llegar en la fase de entrenamiento.
- Aprendizaje estocástico: En este tipo de aprendizaje supervisado se realizan variaciones a los valores de los pesos con número aleatorios, luego se evalúa el desempeño de la red de acuerdo con los valores deseados.

### 1.4.1. Aprendizaje por corrección de error (Backpropagation)

Representa una generalización del algoritmo de mínimos cuadrados. Se actualizan los pesos de acuerdo con el error cuadrático medio obtenido. Al ser del tipo aprendizaje supervisado, la metodología de entrenamiento Backpropagation necesita que le entreguen lineamientos de cada salida, y el valor que se quiere obtener.

Para “enseñarle” a la red neural es necesario entrenar un conjunto de datos, el cual consiste en señales de entradas ***x1*** y ***x2*** asignadas con objetivos correspondientes (salidas deseadas) denominados ***z*** (Ponce Cruz, 2010). El proceso de entrenamiento es iterativo, en cada paso del bucle, los pesos se ven modificados; son calculados a través del proceso de retropropagación del error.

La Figura 1-14 muestra el proceso de entrenamiento mediante retropropagación del error.

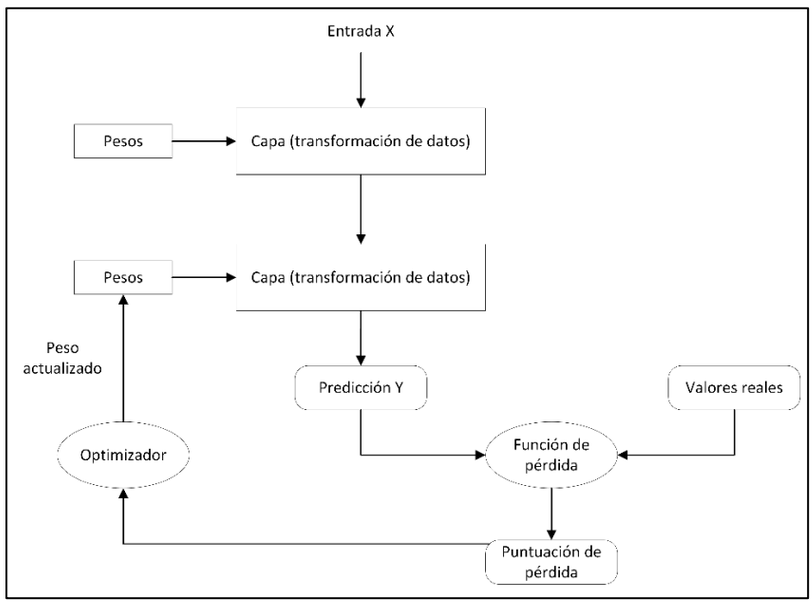


Figura 1-14. Diagrama proceso de retropropagación de error

Fuente: Elaboración propia mediante Visio con base en (Chollet, 2018)

Al inicio se le asignan valores de pesos aleatorios a la red, por lo que esta implementa una serie de transformaciones aleatorias. Cuando comienza la ejecución del programa, los valores entregados están lejos de aquellos a los que se desea llegar; por lo tanto, la puntuación de error es muy alta. Por cada recorrido que ejecuta la red, los pesos se van ajustando y el error va disminuyendo. Tal proceso repetido un número suficiente de veces se llama entrenamiento.

### 1.4.2. Descenso de gradiente estocástico

Dada una función diferenciable, es teóricamente posible encontrar su mínimo analíticamente: se sabe que el mínimo de una función es un punto donde la derivada es 0, así que todo lo que se debe hacer es encontrar todos los puntos donde la derivada tiende a 0 y comprobar para cuál de estos puntos la función tiene el valor más bajo (Chollet, 2018). Se debe encontrar analíticamente la combinación de valores de peso que minimicen la función de pérdida.

El descenso de gradiente estocástico por pequeños lotes corresponde a la extracción de pequeños grupos de datos dentro de la muestra general; se puede definir como la cantidad de muestras que se propagaran en la red, es decir, la red toma el mini lote y entrena al algoritmo, luego toma el segundo mini lote y repite el proceso. Representa el proceso con mayor eficiencia, en comparación con la metodología que calcula el

gradiente para la totalidad de los datos; o el caso opuesto, seleccionar unitariamente los datos de la muestra general. La estructura del procedimiento para llevar a cabo la metodología queda definida a continuación.

1. Dibujar un lote de la muestra de entrenamiento X y los valores de Y correspondientes.
2. Hacer correr la red en X para obtener los valores de predicción y.
3. Calcular la pérdida de la red en el lote, y medir error entre el valor real y la predicción.
4. Calcular la gradiente de la pérdida con respecto a los valores de la red (con paso hacia atrás).
5. Mover los parámetros en dirección opuesta al gradiente, con el fin de reducir la pérdida en el lote.

Una problemática que se presenta en la fase de entrenamiento surge por la selección de una tasa de aprendizaje sea demasiado pequeña; esto provocará que el descenso de la curva necesite muchas iteraciones y podría atascarse en un mínimo local. En la Figura 1-15, step representa el paso o tasa de aprendizaje. Al posicionarse en el punto t=3, se alcanza el mínimo global de la pérdida.

En la Figura 1-16, al continuar con una tasa de aprendizaje pequeña, el algoritmo definirá el punto mínimo local como el mínimo global; sin embargo, al continuar con la curva de la función, se puede apreciar un valor mínimo al definido por el programa, valor que no se tomará en cuenta. Dicho evento ocurre por el aumento de la amplitud al lado derecho del punto mínimo local (es decir, aumento de la pérdida). Esta problemática se puede resolver mediante Momentum.

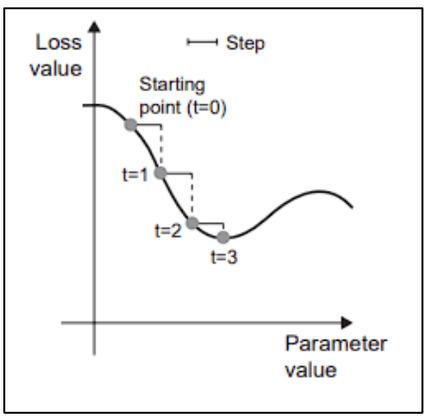


Figura 1-15. Curva de pérdida

Fuente: Deep Learning with Python (Chollet, 2018)

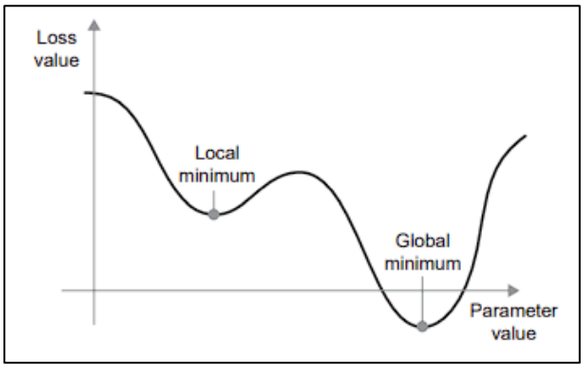


Figura 1-16. Curva mínimo local y mínimo global

Fuente: Deep Learning with Python (Chollet, 2018)

### 1.4.3. Momentum o impulso

Una Buena manera de interpretar el Momentum es imaginar el proceso de optimización del gradiente como una bola de acero que rueda por la curva de pérdida; el impulso se implementa moviendo la pelota en cada paso basándose no solo en el valor de pendiente actual (aceleración actual) sino también en la velocidad actual (resultante de la aceleración pasada) (Chollet, 2018).

### 1.4.4. Optimizador Adam

Combina la metodología de Momentum y RMSProp, calculando una combinación lineal entre el gradiente y el incremento anterior; considera los gradientes recientemente aparecidos en las actualizaciones para mantener diferentes tasas de aprendizaje por variable. (Burrueco, 2009). El cálculo de un momento que se adapta corresponde a un tipo de algoritmo que optimiza el descenso de gradiente. Su eficiencia va más allá de una pequeña cantidad de datos, ya que su uso para grandes cantidades de datos o parámetros demuestra una simplificación respecto a los algoritmos comunes. Utiliza menos memoria.

# CAPÍTULO 3: DISEÑO DE UN MODELO ÓPTIMO DE REDES NEURONALES ARTIFICIALES PARA LA PREDICCIÓN DE EVENTOS DE FALLA

# 2. DESARROLLO DEL MODELO DE REDES NEURONALES

A diferencia del modelo propuesto en (Baptista, 2017), en donde se aplica un modelo AR(1) como modelo base, para posteriormente aplicar Random Forest o Redes Neuronales; se aplicará un modelo base ARIMA(1.0.2), para luego ejecutar el algoritmo de redes neuronales artificiales, analizando indicadores respecto a los dos modelos propuestos con anterioridad (Weibull y ARIMA).

## 2.1. MODELO BASE PROPUESTO

Se diseña la arquitectura del algoritmo de red neuronal artificial, considerando una entrada por la cual, ingresarán los datos obtenidos mediante ARIMA(1.0.2). Se emplearán 4 capas ocultas, respectivamente la primera con 50 neuronas, la segunda con 40 neuronas, la tercera con 30 neuronas, la cuarta con 14; una capa de salida con 1 neurona.

Las funciones de activación corresponden a la tangente hiperbólica en todas las capas de la red, incluyendo la de salida, la tasa de aprendizaje queda definida como 0,0002; los mini lotes para el entrenamiento por descenso de gradiente estocástico, se agrupan en 20 datos. Se definen 1000 iteraciones para el proceso de entrenamiento (Epoch=1000). Para el entrenamiento de la red se emplea el 66% de la base histórica, es decir, 264 datos; mientras que para la prueba y validación se utiliza el 33% restante.

### 2.1.1. Resultados del modelo base

Una vez que se desarrolla el modelo RNA(1.50.40.30.15.1), con parámetros de entrada los valores obtenidos en modelo ARIMA(1.0.2), se ejecutan las predicciones obtenidas a través del algoritmo. El valor de RMSE corresponde a 50,2, el cual es bastante menor en relación con el entregado por Weibull ajustado y modelo ARIMA(1.0.2).

A continuación, en la Figura 2-1 se representa gráficamente las predicciones del modelo indicado con respecto a los tiempos de funcionamiento hasta la falla.

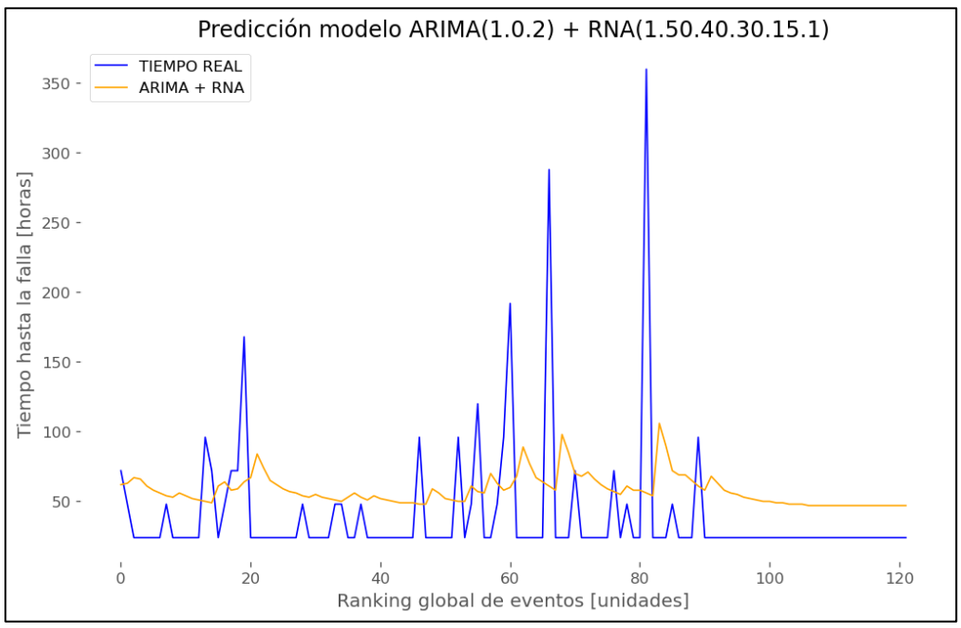


Figura 2-1. Gráfica de tiempo real hasta la falla y predicción modelo ARIMA + RNA

Fuente: Elaboración propia mediante Python

Se puede apreciar visualmente que el modelo es capaz de predecir aquellos eventos menores a 50 horas. Para un análisis con mayor profundidad, se calcula el error absoluto entre cada predicción con respecto al valor real. Para el caso, no se podrán generar intervalos de análisis debido a que todos los datos son menores a los rangos estudiados en los casos anteriores. En la Figura 2-2 se representa gráficamente obtenida.

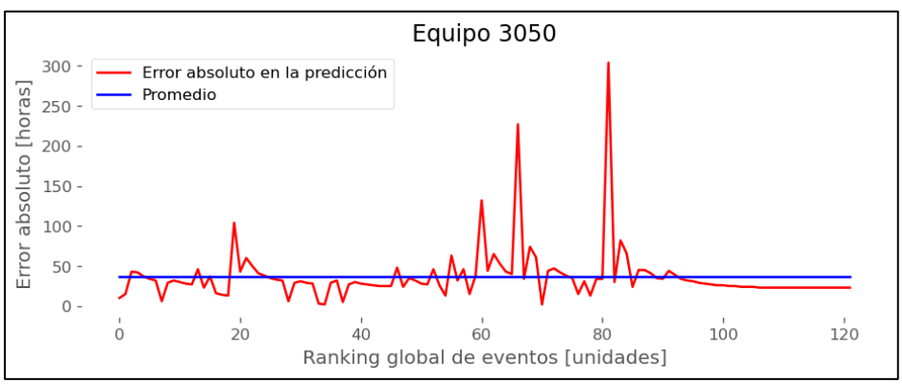


Figura 2-2. Error absoluto en la predicción de eventos modelo ARIMA + RNA

Fuente: Elaboración propia mediante Python

El error absoluto promedio queda definido como 36,4 horas, el error máximo en 304 horas y el mínimo error en 2 horas. Existen 37 errores que se encuentran por sobre el promedio, mientras que 86 se encuentran bajo este indicador. El error medio es considerablemente menor al valor obtenido en los modelos desarrollados con anterioridad. En la Figura 2-3 se pueden observar las comparativas de los modelos propuestos.

El error porcentual medio corresponde a 117,07%, mientras que el máximo error es de un 339,95%; el mínimo error de los valores obtenidos se define como un 3,05%. Existen 57 valores obtenidos en que su error es sobre la media, mientras que bajo la media son 66.

| Modelo | Valor RMSE | Error absoluto medio [hrs] | Error porcentual medio [%] |
|---|---|---|---|
| Weibull ajustado | 342,39 | 123,71 | 215,30% |
| ARIMA(1.0.2) | 337,8 | 129,24 | 228,97% |
| ARIMA(1.0.2) + RNA(1.50.40.30.15.1) | 50,24 | 36,47 | 117,07% |

Figura 2-3. Resultados RMSE y Error absoluto medio

Fuente: Elaboración propia mediante Excel

## 2.2. CAMBIOS EN EL MODELO ARIMA + RNA

Para el desarrollo óptimo de un modelo de redes neuronales artificiales se iteran ciertos parámetros y valores; número de neuronas en capas ocultas, tasa de aprendizaje, mini lote para el descenso de gradiente estocástico. La cantidad de datos en el grupo de entrenamiento consiste en el 66% de los registros de falla, es decir 244 datos; para la validación se emplean los valores restantes. La función de activación corresponde a tangente hiperbólica, al igual que el modelo anterior.

Para la entrada de la red se utilizarán los valores reales de tiempo hasta la falla.

### 2.2.1. Iteraciones sobre modelo base

Se genera una planilla de iteración, en donde cada variabilidad al modelo se plasma en esta, en conjunto con el resultado RMSE obtenido. Se consideran 11 columnas dentro de la planilla; Modelo representa la indexación que recibe dicho modelo, Entradas hace referencia a la cantidad de neuronas en la capa de entrada de la red, RNA indica la

cantidad de neuronas en cada capa oculta de la red, RMSE ARIMA representa al valor RMSE obtenido por el modelo ARIMA(1,0,2), RMSE WEIBULL hace referencia al valor RMSE obtenido por el modelo Weibull ajustado, RMS RNA + ARIMA indica el valor RMSE obtenido en el modelo base, RMSE entrenamiento representa al valor RMSE en el entrenamiento de la red usando el histórico real de tiempos hasta la falla, RMSE RNA validación se refiere al valor obtenido en la etapa de validación (prueba) de la red usando el histórico real de tiempos hasta la falla, mini lote corresponde a la cantidad de datos que entrena el modelo por cada iteración; tasa de aprendizaje indica la variabilidad que se da a dicho parámetro por cada modelo, Mejora ARIMA(1,0,2) + RNA representa una variable binaria que indica si el modelo presentado es mejor al modelo base en relación al valor RMSE.

En la Figura 2-4 se demuestra una parte del proceso de iteración en el modelo, el cual se realizó de forma manual.

| Modelo | Entradas | RNA | | | | | RMSE ARIMA | RMSE WEIBULL | RMSE RNA + ARIMA | RMSE RNA entrenamiento | RMSE RNA validación | Mini lote | Tasa de aprendizaje | Mejora ARIMA(0,1,1) + RNA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 70 | 35 | 20 | 0 | 1 | 337,8 | 342,39 | 50,24 | 425,72 | 52,28 | 15 | 0,00001 | No |
| 2 | 1 | 70 | 35 | 20 | 0 | 1 | 337,8 | 342,39 | 50,24 | 203,12 | 51,86 | 15 | 0,0005 | No |
| 3 | 1 | 70 | 35 | 20 | 0 | 1 | 337,8 | 342,39 | 50,24 | 363,95 | 51,92 | 20 | 0,00005 | No |
| 4 | 1 | 70 | 35 | 20 | 0 | 1 | 337,8 | 342,39 | 50,24 | 427,19 | 51,57 | 10 | 0,00001 | No |
| 5 | 1 | 70 | 35 | 20 | 0 | 1 | 337,8 | 342,39 | 50,24 | 437,32 | 51,17 | 30 | 0,00001 | No |

Figura 2-4. Fragmento de tabla iteraciones sobre modelo base

Fuente: Elaboración propia en Excel

### 2.2.2. Modelo óptimo

Una vez ejecutados los cambios al modelo base, se opta por aquel que posee un menor valor de RMSE, con respecto a sus pares. Para este caso de análisis, el modelo que obtuvo un mejor comportamiento con las predicciones fue el número 15, es decir, el RNA(1.90.35.20.10.1). Su configuración se representa en la Figura 2-5.

| Modelo 15 | |
|---|---|
| **Entradas** | 1 |
| **RNA** | 90 |
| | 35 |
| | 20 |
| | 10 |
| | 1 |
| **RMSE validación** | 4,75 |
| **Mini lote** | 20 |
| **Tasa de aprendizaje** | 0,0002 |
| **Mejora ARIMA(1.0.2) + RNA** | Si |

Figura 2-5. Modelo RNA(1.90.35.20.10.1)

Fuente: Elaboración propia mediante Excel

Posee una capa entrada, la cual recibe los datos puros desde el histórico de fallos, tiene 90 neuronas en la primera capa oculta, 35 neuronas en la segunda capa oculta, 20 neuronas en la tercera capa oculta, 10 neuronas en la cuarta capa oculta y 1 neurona en la capa de salida, la cual entrega las predicciones. El valor RMSE queda definido como 4,75, significantemente menor al determinado en los otros modelos; los mini lotes para el entrenamiento se agrupan en 20 datos, con una tasa de aprendizaje de 0,0002.

La arquitectura de la red queda representada en la Figura 2-6, en donde la primera capa oculta visualmente solo refleja el 50% de la cantidad total de neuronas que posee.

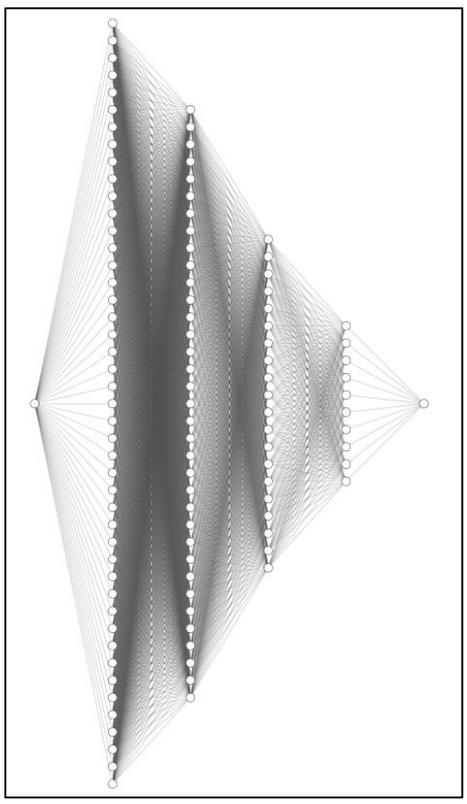

Figura 2-6. Representación visual de red neuronal

Fuente: Elaboración propia mediante Alexlenail

### 2.2.3. Resultados del modelo RNA(1.90.35.20.10.1)

Tras ejecutar las diferentes iteraciones dentro de la red neuronal, se decide que el modelo que representa mejor la vida útil del activo 3050 de la planta de celulosa es el RNA(1.90.35.20.10.1). En total trabaja con 335.741 parámetros, y el valor obtenido en RMSE es de 4,75. Las predicciones obtenidas en los datos de prueba se ven plasmados en la Figura 2-7.

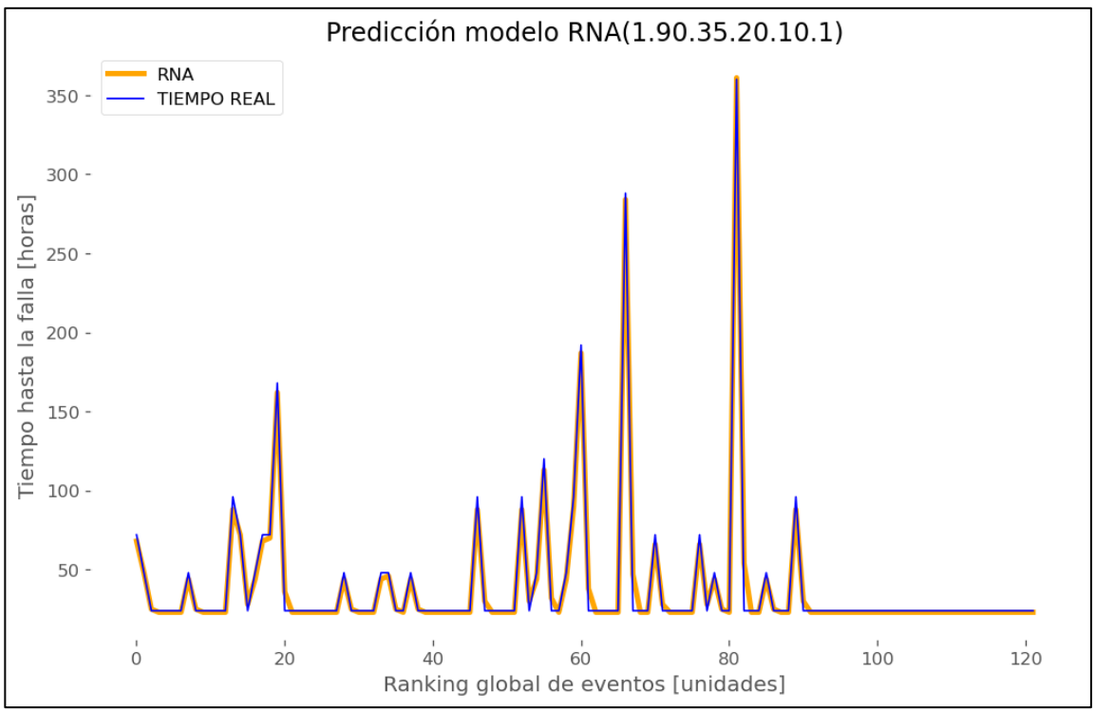


Figura 2-7. Gráfica de tiempo hasta la falla y predicción modelo RNA(1.90.35.20.10.1)

Fuente: Elaboración propia mediante Python

Se puede ver que las predicciones del modelo son ciertamente certeras en comparación con los otros modelos. Para aquellos eventos ocurridos sobre las 250 horas de funcionamiento, las diferencias son levemente mayores.

Para conocer objetivamente el comportamiento de las predicciones, se grafica el error absoluto de estas, con relación al tiempo real. El resultado se puede ver en la Figura 2-8

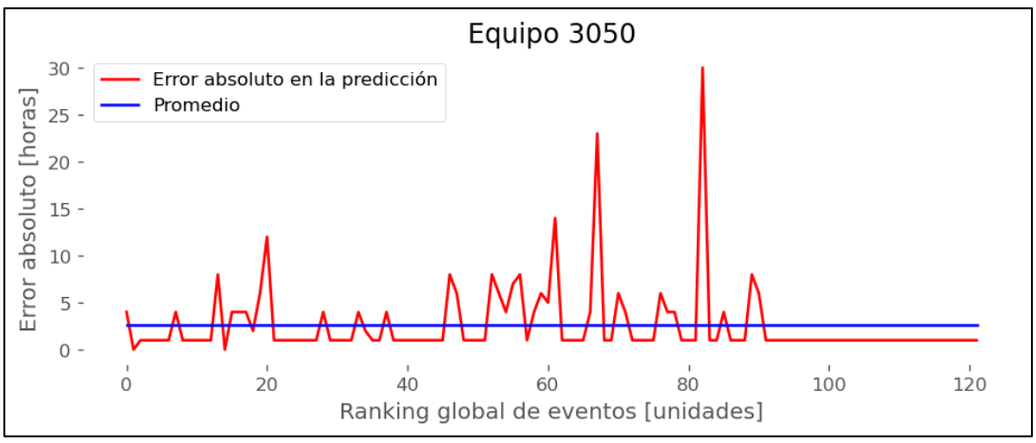


Figura 2-8. Error absoluto medio modelo RNA(1.90.35.20.10.1)

Fuente: Elaboración propia mediante Python

Existen 89 errores menores que el promedio, mientras que 33 se encuentran por sobre este valor. El error máximo obtenido es de 30 horas, el menor es de 0 horas. El promedio de error absoluto queda definido como 2,4 horas. Debido a la baja dimensión de los errores obtenidos, resulta innecesario analizar los resultados mediante intervalos de tiempo.

El error porcentual medio corresponde a 7,14%, mientras que el máximo error es de un 126,46%; el mínimo error de los valores obtenidos se define como un 0,17%. Existen 26 valores obtenidos en que su error es sobre la media, mientras que bajo la media son 96. A continuación, en la Figura 2-9 se muestran los valores obtenidos de RMSE y Error absoluto medio en los distintos modelos desarrollados a lo largo de este trabajo.

| Modelo | Valor RMSE | Error absoluto medio [hrs] | Error porcentual medio [%] |
|---|---|---|---|
| Weibull ajustado | 342,39 | 123,71 | 215,30% |
| ARIMA(1.0.2) | 337,8 | 129,24 | 228,97% |
| ARIMA(1.0.2) + RNA(1.50.40.30.15.1) | 50,24 | 36,47 | 117,07% |
| RNA(1.90.35.20.10.1) | 4,75 | 2,4 | 7,14% |

Figura 2-9. Resultados modelos analizados

Fuente: Elaboración propia mediante Excel

### 2.2.4. Comparación entre modelos

Al realizar la comparativa entre las métricas estadísticas obtenidas de los diferentes modelos vistos dentro del presente trabajo de título, se obtiene que el valor RMSE obtenido por ARIMA(1.0.2), ARIMA(1.0.2) + RNA(1.50.40.30.15.1) y RNA(1.50.40.30.15.1); todos mejoran el valor obtenido por Weibull ajustado. Los errores absolutos no consiguen el mismo comportamiento, al igual que los errores porcentuales obtenidos. En la Figura 2-10 se pude ver las mejoras o descensos entre los errores absolutos y errores porcentuales obtenidos de cada modelo, en comparación con el modelo Weibull ajustado.

| Modelo | Error absoluto medio [hrs] | Mejora Weibull | Error porcentual medio [%] | Mejora Weibull |
|---|---|---|---|---|
| **Weibull vs ARIMA(1.0.2)** | -4,47% | No | -6,35% | No |
| **Weibull vs ARIMA(1.0.2) + RNA(1.50.40.30.15.1)** | 70,52% | Si | 48,87% | Si |
| **Weibull vs RNA(1.90.35.20.10.1)** | 98,06% | Si | 93,90% | Si |

Figura 2-10. Comparación de errores entre modelos

Fuente: elaboración propia mediante Excel

El modelo ARIMA(1.0.2) aumenta el error medio absoluto en un 4,47% en comparación con Weibull ajustado; el error porcentual medio sigue el mismo comportamiento, aumentado en un 6,35%. En lo que respecta al modelo ARIMA(1.0.2) + RNA(1.50.40.30.15.1), el error absoluto medio disminuye en un 70,52% en comparación con Weibull ajustado; el error porcentual medio disminuye en un 48,87%. Cabe destacar la notoria mejoría en los indicadores de error al aplicar herramientas de Machine Learning a un modelo estadístico de series temporales.

El modelo RNA(1.90.35.20.10.1) supera toda expectativa en sus resultados, logrando una disminución del error absoluto medio de 98,06% en comparación con Weibull ajustado; mientras que el error porcentual medio disminuye en un 93,90%.

En la Figura 2-11 se representa gráficamente los resultados obtenidos mediante los modelos ARIMA(1.0.2) + RNA(1.50.40.30.15.1) y RNA(1.90.35.20.10.1), comparándolos con los tiempos reales de funcionamiento.

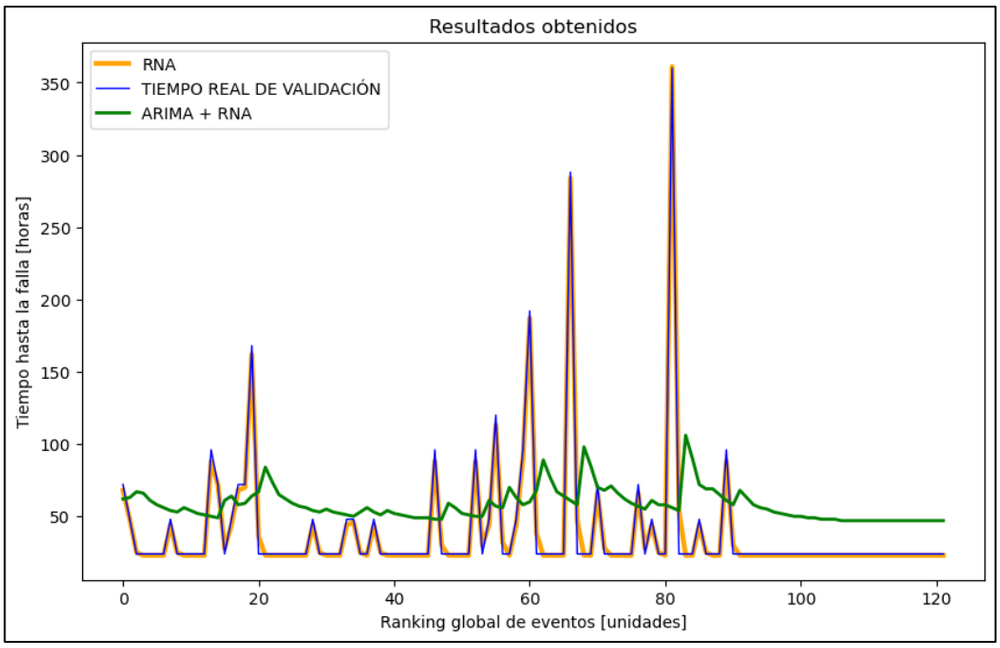


Figura 2-11. Comparación de resultados obtenidos

Fuente: Elaboración propia mediante Python

# CONCLUSIONES

El modelo de uso de vida Weibull, es capaz de modelar el ciclo de un activo, adquiriendo indicadores claves y permitiendo generar estimaciones, en base a los recursos monetarios de las empresas. Si bien es una herramienta útil, el margen de error que conlleva con respecto a la realidad abre espacio al riesgo de error.

Mediante el modelo Weibull se obtienen 49 errores mayores a 123 horas, es decir, el desacierto es mayor a 5 días de funcionamiento. Con un valor RMSE de 342,3, el modelo se posiciona como el que genera mayores errores en los pronósticos.

En relación con la propuesta generada en (Baptista, Sankararaman, de Medeiros, Nascimiento Jr., & Henriquesa, 2017), se itera sobre esta, cambiando el modelo AR(1) por el modelo ARIMA(1.0.2). Los resultados fueron óptimos, ya que el valor RMSE disminuyó en 1,34% respecto al valor obtenido mediante Weibull ajustado.

La aplicación de herramientas de inteligencia artificial, como el Machine Learning a tareas de mantenimiento, suena prometedor para representar los procesos en el futuro, y queda demostrada su eficiencia a la hora de modelarlos.

Al desarrollar el modelo base ARIMA(1.0.2) + RNA(1.50.40.30.15.1) se logra minimizar el RMSE obtenido por Weibull Ajustado en un 85,33%, y por ARIMA(1.0.2) en 85,13%. Si bien la mejora es positiva, el modelo base representa un máximo error de 304 horas, (casi 13 días de diferencia).

Para finalizar este trabajo de título, se propone una mejora al modelo base sugerido en (Baptista, Sankararaman, de Medeiros, Nascimiento Jr., & Henriquesa, 2017), utilizando como entrada los datos directamente desde el histórico.

La iteración sobre los parámetros de la red neuronal artificial, logra resultados sumamente positivos en las predicciones de eventos de falla. La estructura queda definida como una RNA(1.90.35.20.10.1). Logra superar en la métrica RMSE al modelo Weibull ajustado en un 98,61%, a ARIMA(1.0.2) en 98,59% y al modelo base ARIMA(1.0.2) + RNA(1.50.40.30.15.1) en 90,55%. El error máximo queda definido como 30 horas.

Para tareas de mantenimiento predictivo, tales como análisis vibratorios, ultrasonidos, etc. El modelo es capaz de definir con cierta sensibilidad, cuando ocurrirá el fallo, por lo tanto, las tareas de mantenimiento que se programen a partir del modelo, serán exactas con respecto al tiempo de ejecución de estas.

# BIBLIOGRAFÍA


Baptista, S. D. (2017). *Forecasting Fault Events for Predictive Maintenance using Data-driven Tech - niques and ARMA Modeling.*

Burrueco, D. (2009). *Interactive Chaos*. Obtenido de https://interactivechaos.com/es/manual/tutorial-de-machine-learning/adam

Cedeño, A. M. (2022). *ResearchGate.* Obtenido de https://www.researchgate.net/publication/47900975_Un_modelo_neuronal_basado_en_la_metaplasticidad_para_la_clasificacion_de_objetos_en_senales_1-d_y_2-d

Chollet, F. (2018). *Deep Learning with Python.*

Christiansen, M. A. (2020). *Desarrollo de un modelo de predicción de tráfico de clientes para optimizar la dotación de personal en tiendas físicas del retail.* Santiago, Chile.

Matich, D. (2001). *Redes Neuronales: Conceptos Básico y Aplicaciones* . Rosario.

Ponce Cruz, P. (2010). *Inteligencia Artificial con Aplicaciones a la Ingeniería* . Ciudad de México.

Serna, E. (2017). *Desarrollo e innovación en ingeniería.* Medellín.

Thyago P. Carvalho, F. A. (2019). A systematic literature review of machine learning methods applied to predictive maintenance.

Uribe, J. F., Ortiz, E. J., Romero, H., & Valbuena, H. L. (agosto de 2017). *ResearchGate*. Obtenido de https://www.researchgate.net/publication/318940178_Incidencia_de_las_politicas_publicas_de_empleo_sobre_la_desocupacion_en_Colombia_un_analisis_de_intervencion_para_el_periodo_2002_-_2014

# ANEXO A

- **TBF: Tiempo Entre Fallas:** Representa el tiempo transcurrido, hasta la ocurrencia del evento de falla.

$$TBF = (X_1 - X_{x-1}) \cdot 24$$