

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

**PROPUESTA DE METODOLOGÍA DE PREDICCIÓN DE EVENTOS PARA LA
MEJORA EN LA GESTIÓN DE PROCESOS DE MANTENIMIENTO
APLICANDO MODELOS DE REGRESIÓN Y MACHINE LEARNING A
EQUIPO CRÍTICO EN PLANTA DE CELULOSA**

Trabajo de Titulación para optar al
Título profesional de Ingeniero en
Mantenimiento Industrial con
Licenciatura en Ingeniería de
Mantenimiento

Renzo Ariel Gallardo Almonacid

Profesor Guía:

Mg. Ing. Sr. Andrés Aránguiz Garrido

2022

RESUMEN

KEYWORDS: MACHINE LEARNING, MANTENIMIENTO PREDICTIVO, GESTIÓN DE ACTIVOS, CONFIABILIDAD, WEIBULL, ARMA, SVR

Para la obtención de productos derivados de la madera en una planta de celulosa, previamente se requiere que ésta sea tratada en diferentes procesos. El sistema Power Feed de una planta celulosa, es el primero dentro de la producción y cumple la función de alimentar a todo el proceso con troncos de madera cortada en bruto mediante un transportador de tambor, hacia el proceso de descortezado. Las detenciones no programadas debido a fallas imprevistas en el transportador de tambor generaron costos de ineficiencia cercanos a los 7.694,4 [USD/h], durante el periodo de producción en el año 2018. Sin embargo, para controlar esta situación, los procesos de gestión de mantenimiento dentro de la planta utilizaron registros de los tiempos de funcionamiento diarios del transportador de tambor, considerado como equipo crítico, para así, aplicar el modelo de Uso de vida de Weibull a estos datos e intentar predecir el próximo evento de falla, y posteriormente poder realizar acciones de mantenimiento preventivo de forma anticipada. Sin embargo, la gran desventaja del modelo de uso de vida de Weibull es que sus resultados se basan en la probabilidad de que un equipo se encuentre operativo a intervalos de tiempos fijos, sin considerar el comportamiento variable de los procesos, lo que puede resultar en algunos casos en un exceso de intervenciones y, por lo tanto, de recursos. A raíz de esto, distintos estudios actuales han comprobado que es posible mejorar la precisión con la que se realizan estas predicciones mediante el uso de modelos de regresión y de Machine Learning, ya que, estos últimos tienen la capacidad de manejar datos multivariados de gran dimensión, y de extraer relaciones ocultas dentro de los datos en entornos complejos y dinámicos, como lo son los entornos industriales. Por tal motivo, el principal objetivo del presente proyecto de título es proponer la implementación de una nueva metodología de predicción, la cual se basa en un modelo de regresión SARIMA, junto con un modelo basado en datos (o de ML), llamado: Support Vector Regression (SVR), para la mejora en el desempeño de los procesos de gestión del mantenimiento industrial. En el estudio realizado, se logra determinar que el modelo de regresión SARIMA de parámetros: $(p, d, q)(P, D, Q, s) = (4, 1, 0)(2, 0, 0, 5)$, junto con un modelo basado en datos SVR, de hiperparámetros: ($kernel = rbf$, $C = 10000$, $epsilon = 0.05$, $gamma = 0.1$), supera al modelo de Uso de vida de Weibull de parámetros: ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$), en cuanto a las métricas de rendimiento utilizadas para

evaluar errores de bias, exactitud y precisión, logrando aumentar el MTBF estimado, pasando de 22,72 a 23,56 [h] entre intervenciones de mantenimiento, con una mínima desviación estándar. Por lo tanto, debido a su capacidad de superar al modelo de Weibull al disminuir los errores de predicción, aumentar en un 3,7% la cantidad de horas de funcionamiento medio (MTBF), con una mínima desviación, se selecciona y propone este modelo predictivo como el mejor candidato para predecir fallas en el equipo en estudio, ya que así, es posible proyectar y programar de manera óptima las intervenciones de mantenimiento. De esta manera, se comprueba que, al aplicar un modelo de regresión junto con un modelo de ML, es posible superar en rendimiento al modelo clásico de predicción utilizado actualmente por la industria.

ÍNDICE

INTRODUCCIÓN 1

OBJETIVOS GENERAL Y ESPECÍFICOS 3

OBJETIVO GENERAL 3

OBJETIVOS ESPECÍFICOS..... 3

1 ANTECEDENTES GENERALES..... 5

1.1 MACHINE LEARNING Y SU APLICACIÓN EN EL MANTENIMIENTO PREDICTIVO 5

1.1.1 Definición de Machine Learning 6

1.1.2 Machine Learning y su aplicación en el Mantenimiento Predictivo 8

1.2 CASO DE ESTUDIO..... 9

1.2.1 Descripción del proceso..... 9

1.2.2 Descripción del equipo..... 9

1.2.3 Descripción de la serie de datos de entrada 12

1.3 MODELO CLÁSICO DE PREDICCIÓN 13

1.3.1 Modelo de Uso de vida de Weibull 13

1.3.2 Aplicación del modelo de uso de vida de Weibull 16

2 METODOLOGÍA PROPUESTA PARA LA MEJORA EN LA PREDICCIÓN 23

2.1 MARCO DE TRABAJO PROPUESTO 23

2.1.1 Exploración y preprocesamiento de datos..... 24

2.1.2 Entrenamiento y prueba de un modelo..... 26

2.2 MODELOS PROPUESTOS PARA LA PREDICCIÓN 27

2.2.1 Autorregresive Integrated Moving Average 27

2.2.2 Support Vector Regression..... 30

3 EVALUACIÓN DEL MODELO DE PREDICCIÓN PROPUESTO 38

3.1 MÉTRICAS DE RENDIMIENTO 39

3.2 RESULTADOS 42

3.2.1 Exploración de la serie de datos 42

3.2.2 Datos de entrenamiento y prueba..... 48

3.2.3 Resultados de entrenamiento 49

3.2.4 Resultados de prueba 56

3.2.5 Resumen de resultados 59

CONCLUSIONES 61

BIBLIOGRAFÍA	62
ANEXOS	64
ANEXO A: ALGORITMO PARA ENTRENAMIENTO Y PRUEBA DE UN MODELO DE REGRESIÓN SARIMA	64
ANEXO B: ANÁLISIS COSTO/BENEFICIO DE IMPLEMENTAR EL MODELO PROPUESTO	66
ANEXO C: VENTAJAS Y DESVENTAJAS DEL MODELO DE PREDICCIÓN PROPUESTO.....	72

ÍNDICE DE FIGURAS

- Figura 1-1. La inteligencia artificial y algunas subcategorías
- Figura 1-2. Proceso de producción de celulosa Kraft.
- Figura 1-3. Disposición sistema de alimentación Power Feed. A la izquierda se muestra el equipo en estudio, el cual consiste en un transportador de tambor transportando rollizos de Pino radiata mediante cadenas con zapata, hacia el proceso de descortezado y astillado.
- Figura 1-4. Plano detalle y dimensiones generales del sistema de alimentación Power Feed. El equipo tiene un largo y un ancho aproximados de 32 por 6 metros respectivamente.
- Figura 1-5. Vista interior, sistema de alimentación Power Feed. Acá se pueden apreciar las 9 cadenas con zapata encargadas de transportar los rollizos en bruto de Pino radiata mediante un transportador de tambor hacia el proceso de descortezado y astillado.
- Figura 1-6. Vista extendida de las cadenas con zapata del sistema de alimentación Power Feed.
- Figura 1-7. Representación de la curva de la bañera
- Figura 1-8. Representación gráfica del funcionamiento del modelo LU de Weibull
- Figura 1-9. Gráfico “Ln(tfun) vs Ln(-Ln(R(t)))” para la obtención de los parámetros de Weibull
- Figura 1-10. Densidad de falla o probabilidad instantánea de fallas f(t) del equipo 1001, obtenida según parámetros $\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$ del modelo LU de Weibull

Figura 1-11. Confiabilidad y Mantenibilidad equipo 1001, obtenidas según parámetros $\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$ del modelo LU de Weibull

Figura 1-12. Tasa de fallas de equipo 1001, obtenida según parámetros $\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$ del modelo LU de Weibull

Figura 2-1. Diagrama del marco de trabajo propuesto.

Figura 2-2. Ejemplo de descomposición de las componentes de tendencia, estacionalidad y errores residuales de una serie de datos de entrada

Figura 2-3. Descripción gráfica de una covarianza irregular o no constante

Figura 2-4. Ejemplo de los infinitos hiperplanos posibles

Figura 2-5. Ejemplo de hiperplano y margen suave, ajustado mediante vectores de soporte

Figura 2-6. Ejemplo de aplicación de un modelo SVR con kernel RBF

Figura 2-7. Ejemplo de distintos hiperplanos creados mediante SVR, ajustados a la serie de datos con kernels lineales y no lineales

Figura 2-8. Representación del parámetro epsilon y su relación con la función de pérdida $L\epsilon$ del modelo SVR

Figura 3-1. Marco de trabajo propuesto

Figura 3-2. Arriba, se muestran los tiempos de funcionamiento del equipo 1001. En segundo, tercer y cuarto lugar respectivamente, se muestran los gráficos de descomposición de tendencia, estacionalidad y ruido residual del registro de los tiempos de funcionamiento del equipo 1001.

Figura 3-3. Histograma y densidad de la probabilidad de fallas del equipo 1001

Figura 3-4. Gráfico de normalidad QQ de la serie de tiempos de funcionamiento del equipo 1001

Figura 3-5. Gráfico de la función de autocorrelación. Permite evaluar el nivel de dependencia entre las variables de análisis

Figura 3-6. Gráfico de la función de autocorrelación parcial. Permite evaluar el nivel de dependencia entre un rango menor de variables de análisis

Figura 3-7. Gráfico de media móvil y desviación estándar para el análisis de estacionariedad de la prueba de Dickey Fuller

Figura 3-8. Resultados de la prueba de estacionariedad mediante rezagos de Dickey Fuller, obtenidos mediante lenguaje de programación Python. La prueba nos indica que cada 2 rezagos, la serie se vuelve estacionaria.

Figura 3-9. Resultados de la prueba de estacionariedad de Chi-Cuadrado, mediante lenguaje de programación Python

Figura 3-10. En azul se muestra el 80% de los valores de la serie de datos de tiempos de funcionamiento (en horas), del equipo 1001, que serán utilizados para la etapa de entrenamiento, en donde se busca definir los mejores valores para los parámetros del modelo de predicción. En amarillo se muestra el 20% restante de los datos, que serán utilizados para realizar predicciones con los parámetros determinados en la etapa de entrenamiento.

Figura 3-11. Aplicación del método de escalado a la serie de datos de entrenamiento.

Figura 3-12. Resultados de entrenamiento modelo de regresión mediante lenguaje de programación Python

Figura 3-13. Resumen de los resultados de entrenamiento del modelo de regresión mediante lenguaje de programación Python. Acá, el algoritmo optimizador da cuenta de que el modelo de regresión ARIMA que mejor se ajusta a los datos de entrenamiento corresponde a la variante SARIMA, la cual incluye parámetros estacionales. Nota: El nombre “SARIMAX” hace referencia a la variante del modelo ARIMA que utiliza variables estacionales y exógenas, las cuales no se aplican en este proyecto de título.

Figura 3-14. Predicciones de entrenamiento del modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), sobre el conjunto de datos de entrenamiento, mediante lenguaje de programación Python

Figura 3-15. Resultados de entrenamiento del modelo basado en datos SVR, mediante lenguaje de programación Python

Figura 3-16. En rojo se muestran las predicciones SARIMA aplicadas al conjunto de datos de entrenamiento de los tiempos de funcionamiento del equipo 1001 (en azul). En verde se muestran las predicciones obtenidas desde el modelo basado en datos $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5).

Figura 3-17. En rojo se muestran las predicciones SARIMA aplicadas al conjunto de datos de prueba de los tiempos de funcionamiento del equipo 1001 (en amarillo)

Figura 3-18. En este gráfico, se muestran las predicciones finales de la etapa de prueba, mostradas en verde, obtenidas desde el modelo basado en datos $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre

las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5), las cuales se deberían mostrar en rojo.

Figura 3-19. Predicciones del modelo de Uso de vida de Weibull de parámetros ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$)

Figura A. Comparación del impacto que genera el uso de distintas estrategias de mantenimiento

ÍNDICE DE TABLAS

- Tabla 1-1. Registro de tiempos de funcionamiento (en horas) de los primero cinco días de operación del equipo 1001
- Tabla 1-2. Orden ascendente de los primeros cinco valores de la serie de tiempos de funcionamiento “tfun [h]”, para el cálculo de parámetros del modelo de LU de Weibull
- Tabla 1-3. Cálculo de probabilidad acumulada de falla $F(t)$, Confiabilidad $R(t)$ y transformación logarítmica aplicado a los primeros cinco tiempos de funcionamiento para el cálculo de los parámetros de Weibull
- Tabla 1-4. Valores de los parámetros del modelo de predicción clásico: Uso de vida de Weibull, obtenidos desde el conjunto de datos de la Tabla 1-1
- Tabla 1-5. Esperanza de vida media del equipo 1001, basado en el modelo de Weibull
- Tabla 3-1. Métricas para la evaluación del rendimiento de los modelos de predicción en estudio, correspondiente a diferentes tipos de errores entre los pronósticos yEt , y el valor real yEt observado, en un tiempo t determinado.
- Tabla 3-2. Resultados de la prueba de estacionariedad de Dickey Fuller
- Tabla 3-3. Valores de los parámetros del modelo de regresión SARIMA, obtenidos desde el conjunto de datos de entrenamiento
- Tabla 3-4. Resultados de rendimiento del modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), sobre el conjunto de datos de entrenamiento
- Tabla 3-5. Valores de los hiperparámetros del modelo de ML: SVR, obtenidos desde las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5) del conjunto de datos de entrenamiento

Tabla 3-6. Resultados de rendimiento del modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre los datos del conjunto de entrenamiento

Tabla 3-7. Resultados de rendimiento del modelo SARIMA (4, 1, 0) (2, 0, 0, 5), sobre los datos del conjunto de prueba

Tabla 3-8. Resultados de rendimiento del modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre los datos del conjunto de prueba

Tabla 3-9. Resumen de los resultados finales de rendimiento del modelo clásico de predicción de Uso de vida de Weibull junto con los resultados de las fases de entrenamiento y prueba, con un modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5) y un modelo de ML: $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$

SIGLAS Y SIMBOLOGÍAS

SIGLAS

ARMA	:	Autoregressive Moving Average
ARIMA	:	Autoregressive Integrated Moving Average
IA	:	Inteligencia Artificial
AI	:	Artificial Intelligence
LU	:	Life Usage
MAD	:	Mean Absolute Deviation
MAE	:	Mean Absolute Error
MAPE	:	Mean Absolute Percentage Error
MdAE	:	Median Absolute Error
MdE	:	Median Error
ME	:	Mean Error
ML	:	Machine Learning
MSE	:	Mean Standard Error
MTBF	:	Mean Time Between Failures
MTTR	:	Mean Time to Repair
OEE	:	Overall Equipment Effectiveness
PdM	:	Predictive Maintenance
PvM	:	Preventive Maintenance
PCA	:	Principal Components Analysis
RBF	:	Radial Basis Function
RCM	:	Reliability Centered Maintenance
RMSE	:	Root Mean Square Error
ROI	:	Return on Investment
RUL	:	Remaining Useful Life
SVM	:	Support Vector Machine
SVR	:	Support Vector Regression
SARIMA	:	Seasonal Autoregressive Integrated Moving Average
SSD	:	Sample Standard Deviation

SIMBOLOGÍAS

h	:	Horas
Ton	:	Toneladas
T_{fun}	:	Tiempo de funcionamiento en horas
$f(t)$	:	Probabilidad de falla instantánea
$R(t)$	:	Confiabilidad
$F(t)$	:	Probabilidad acumulada de falla
α	:	Alfa
β	:	Beta
γ	:	Gamma
$\lambda(t)$	:	Tasa de fallas
$MTBF$	:	Tiempo medio entre fallas
$E(MTBF)$	:	Tiempo medio entre fallas esperado
$\Gamma()$	:	Función Gamma
t	:	Tiempo “t” de una serie de tiempo
Z_t, Z_{t-i}	:	Ruido blanco de la serie original y ruido pasado
ϕ_i	:	Coeficientes autorregresivos
θ_i	:	Coeficientes de media móvil
X_t, X_{t-i}	:	Valores originales y pasados de la serie
Y_t, Y_{t-i}	:	Valores originales y pasados de la serie diferenciada
p	:	Orden de coeficientes para parte autorregresiva
d	:	Orden de diferenciaciones para series no estacionarias
q	:	Orden de coeficientes para media móvil
P	:	Orden de coeficientes para parte autorregresiva con estacionalidad
D	:	Orden de diferenciaciones para series no estacionarias estacionales
Q	:	Orden de coeficientes para media móvil con estacionalidad
s	:	Orden de períodos de estaciones
∇^d	:	Operador de diferenciación
∇_s^D	:	Operador de diferenciación estacional
n	:	Número de observaciones en el conjunto de pruebas $\{y_E(t)\}_{t=1}^n$
$y_E(t)$	:	Tiempo observado hasta la próxima falla del índice de tiempo t
$\hat{y}_E(t)$	:	Tiempo predicho hasta la próxima falla del índice de tiempo t
rbf	:	Función de base radial para SVR
$linear$	:	Función linear para SVR
$poly$	:	Función polinomial para SVR

<i>sigmoid</i>	:	Función sigmoide para SVR
ε	:	Epsilon
σ	:	Gamma
ξ	:	Holgura negativa en SVR
ξ^*	:	Holgura positiva en SVR
C	:	Parámetro de penalización de error para SVR
N	:	Cantidad total de la muestra para SVR
$K()$	:	Función kernel para SVR
$f(x)$	:	Función predictiva
w	:	Vector de peso o predictor
$\phi(x)$	:	Transformación no lineal aplicada a la matriz de entrada SVR
b	:	Vector de error
L_ϵ	:	Función de perdida insensible
y	:	Valor de la variable dependiente del conjunto de entrenamiento
CG	:	Costo global
$C_{Inversión}$	:	Costo de inversión
$C_{ejercicio}$	:	Costo de ejercicio
A, A_s	:	Disponibilidad, disponibilidad del sistema
H	:	Tiempo de operación
C_{inef}	:	Costo de ineficiencia
Fa	:	Factor de actualización
η	:	Factor de rendimiento
Q'	:	Factor o tasa de calidad
H	:	Horizonte de operación
OEE	:	Coeficiente de efectividad total de un equipo

INTRODUCCIÓN

La gran cantidad de datos recopilados por los sistemas industriales actuales, contienen información sobre procesos, eventos y alarmas que ocurren a lo largo de una línea de producción. Cuando estos datos se procesan y analizan, es posible extraer información y conocimientos valiosos del proceso y la dinámica de operación de un sistema. Mediante la aplicación de un modelo de predicción basado en datos (o de Machine Learning), es posible encontrar resultados estadísticos interpretativos que favorecen la toma de decisiones estratégicas dentro de una organización, logrando anticipar eventos futuros con una alta precisión. Ésta metodología es aplicable en distintas áreas, y brinda sus respectivas ventajas como, por ejemplo: reducción de la cantidad de fallas y detenciones inesperadas en equipos industriales, reducción de inventarios de repuestos, aumento de vida útil de ítems mantenibles, reducción de costos de mantenimiento, aumento en la producción de una línea de procesos, entre otros (Carvalho, 2019). Como es posible apreciar, las ventajas mencionadas tienen una fuerte relación con los procedimientos aplicados en el área del mantenimiento predictivo (PdM), debido a que es posible programar y planificar de manera más anticipada y efectiva. Aplicar estas metodologías, permite llevar a cabo de mejor manera las funciones vitales que requiere una organización para una gestión exitosa de sus procesos. Al proyectar estas acciones en el tiempo, es posible también, promover políticas de mantenimiento adecuadas para el sistema productivo. Para obtener estas proyecciones de eventos futuros, se utilizan herramientas matemáticas que utilizan fórmulas de cálculo basadas en parámetros definidos por sus autores. En la actualidad, la distribución de Weibull es el modelo clásico de predicción utilizado por la industria. La principal desventaja de este modelo es que entrega valores de predicción a tiempos de intervalos fijos y constantes, asumiendo que las condiciones de operación son siempre las mismas, sin considerar del todo la variabilidad de los procesos en función del tiempo como una variable la cual, en realidad, tiene un comportamiento inconstante o mutable. Por tales motivos, se propone la implementación de metodologías que utilizan modelos de regresión y Machine Learning para la mejora en el desempeño de la predicción de fallas. En el Capítulo 1 se presentan definiciones clave para contextualizar el estado del arte actual del Machine Learning (ML) y su relación con el mantenimiento industrial (PdM). En el Capítulo 2, se definen metodologías de trabajo y se presentan los modelos propuestos para la predicción del próximo evento de falla de un equipo crítico, perteneciente a una planta de celulosa. Se valida la propuesta, analizando y comparando los resultados de las métricas de evaluación del modelo de predicción clásico utilizado en la industria y los modelos propuestos en el Capítulo 3. La propuesta presentada se basa en el artículo de investigación desarrollado

por (Baptista, 2017), en donde se comprueba de manera experimental que, los tiempos a intervalos fijos de cambio de una válvula de presurización perteneciente a un avión, estimados mediante el modelo de Weibull, muestran un menor rendimiento de predicción en comparación con el modelo propuesto. Por lo tanto, se inicia este estudio partiendo desde la hipótesis de que un modelo de predicción, conformado por un modelo de regresión SARIMA y, otro basado en datos SVR, supera en rendimiento al modelo de predicción clásico utilizado en la industria. La propuesta de valor es contribuir hacia la mejora del estado del arte en este tipo de problemas.

OBJETIVOS GENERAL Y ESPECÍFICOS

OBJETIVO GENERAL

Proponer la implementación de una metodología que utiliza modelos de regresión y Machine Learning para la mejora en el desempeño de los procesos de gestión del mantenimiento industrial, mediante la predicción de fallas de un equipo crítico perteneciente a una planta de celulosa.

OBJETIVOS ESPECÍFICOS

- Definir el Machine Learning y su relación con los procesos de gestión del mantenimiento industrial actual, mediante la definición del modelo de predicción clásico para la contextualización el estado del arte.
- Proponer una metodología de predicción basada en un modelo de regresión junto con un modelo de Machine Learning, mediante un marco de trabajo y la definición de los modelos a utilizar para la obtención de los valores de rendimiento generados en cada ciclo de entrenamiento.
- Evaluar la propuesta de una metodología de predicción, mediante la comparación de las métricas de rendimiento obtenidas en los distintos modelos entrenados para la mejora en el desempeño de los procesos de gestión del mantenimiento industrial.

CAPÍTULO 1: ANTECEDENTES GENERALES

1 ANTECEDENTES GENERALES

1.1 MACHINE LEARNING Y SU APLICACIÓN EN EL MANTENIMIENTO PREDICTIVO

Para comprender el Machine Learning (ML), primero es necesario entender que es una rama de la Inteligencia Artificial (IA). Por definición, la inteligencia es la facultad de la mente humana que permite aprender, entender, razonar, tomar decisiones y formarse una idea determinada de la realidad. En ciencias de la computación, la IA es la disciplina que intenta replicar y desarrollar la inteligencia y sus procesos implícitos a través de computadoras. Puede entenderse como un programa de computación diseñado para realizar determinadas operaciones que se consideran propias de la inteligencia humana, como el autoaprendizaje (Riial, 2022). De esta manera, el ML puede ser representado como una de las subcategorías de la IA, como se muestra en la Figura 1-1.

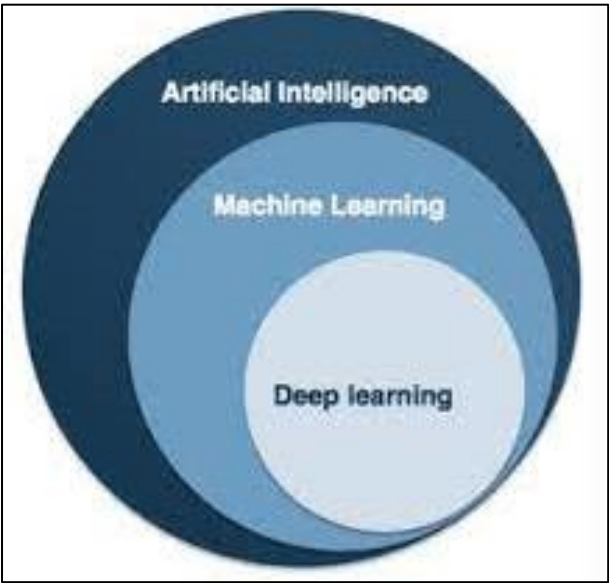


Figura 1-1. La inteligencia artificial y algunas subcategorías

Fuente: <https://www.percepcioneseconomicas.cl/data-science/>

En la actualidad, esta capacidad de aprendizaje está siendo aprovechada e incorporada en procesos y equipos industriales, incluso en el área de mantenimiento, en donde existen procesos de gestión que utilizan indicadores para tomar decisiones. Por ejemplo, se utiliza el tiempo medio entre fallas (MTBF) de un equipo para intentar

predecir eventos de fallas, el cual se determina mediante el modelo clásico de predicción utilizado por la industria, llamado modelo de uso de vida de Weibull. A lo largo del tiempo, se ha comprobado que éste entrega buenos resultados y que se logra obtener beneficio a partir de su implementación. Sin embargo, estos procesos se encuentran actualmente en plena evolución. Hoy en día, es posible utilizar modelos de regresión y de ML para mejorar la precisión con la que se realizan estas predicciones, mejorando así también, los procesos de gestión del mantenimiento logrando aumentar el beneficio obtenido al aplicarlos.

1.1.1 Definición de Machine Learning

Podemos definir el Machine Learning (ML), también conocido como Aprendizaje Automático, como la ciencia de hacer que las computadoras actúen sin estar explícitamente programadas. Es la práctica de usar algoritmos para analizar datos, aprender de ellos y luego hacer una determinación o proyección sobre algo. El ML incluye técnicas estadísticas que permiten a las máquinas mejorar ciertas tareas basándose en la experiencia (aprendeIA, 2022). En la actualidad, nos vemos involucrados día a día con estos sistemas al proporcionar información en distintas plataformas digitales. El sistema de recomendación de películas de Netflix o el sistema de clasificación de correos no deseados de Gmail son algunos ejemplos de algoritmos de ML. El ML tiene como objetivo construir sistemas informáticos que mejoren automáticamente con la experiencia y se basa en el uso de algoritmos que puedan entregar información de interés en base a un conjunto de datos, como lo son, por ejemplo, las series de tiempo. De esta manera, dentro del ML podemos encontrar distintas formas de aprendizaje:

- I. Aprendizaje supervisado: Este tipo de aprendizaje supone que partimos de una serie de datos en donde se conoce el valor del atributo objetivo (en este caso, la predicción), para el conjunto de datos que disponemos. Esto le permitirá al algoritmo poder aprender y establecer una función capaz de predecir el atributo objetivo para un nuevo conjunto de datos (Healt Data Miner, 2022). Las dos grandes familias de algoritmos supervisados son:
 - i. Algoritmos de Regresión: Se aplican cuando el resultado a predecir es un atributo numérico. En la regresión, se entrena un algoritmo para identificar una relación funcional entre los parámetros de entrada y salida. El valor de salida es una función de los parámetros de entrada. La exactitud de un algoritmo de regresión se calcula en función del error entre la serie de datos de entrada y la serie de datos de salida prevista por la función

- ii. **Algoritmos de Clasificación:** Se aplican cuando el resultado a predecir es un atributo categórico. En la clasificación, se entrena a un algoritmo para clasificar los datos de entrada en variables discretas. Durante el entrenamiento, los algoritmos reciben datos de entrada de entrenamiento con una etiqueta de clasificación (Tibco, 2022).

- II. **Aprendizaje no supervisado:** Los métodos de aprendizaje no supervisado son algoritmos que basan su proceso de entrenamiento en un conjunto de datos sin etiquetas o clases previamente definidas. Es decir, a priori no se conoce ningún valor objetivo o de clase, ya sea categórico o numérico. El aprendizaje no supervisado está dedicado a las tareas de agrupamiento, también llamadas clustering o segmentación, donde su objetivo es encontrar grupos similares en el conjunto de datos (Healt Data Miner, 2022).

- III. **Reinforcement Learning: Aprendizaje reforzado.** Este tipo de aprendizaje intenta conseguir que una IA aprenda a decidir mediante su propia experiencia. Es decir, que, ante una situación determinada, sea capaz de seleccionar por sí misma la mejor acción a ejecutar en ese momento mediante un proceso interactivo de prueba y error a base de reforzar positivamente cada vez que se aproxima o logra un objetivo (ACCIONA, s.f.).

- IV. **Procesamiento de Lenguaje Natural o Natural Language Processing:** Rama de la IA que ayuda a las computadoras a entender, interpretar y manipular el lenguaje humano (SAS Institute Inc., 2022). Se enfoca en la comprensión, el manejo y la generación del lenguaje natural por parte de las máquinas. Es la interfaz entre la ciencia informática y la lingüística y, por ende, se basa en la capacidad de la máquina para interactuar directamente con los humanos (DataScientest, 2022).

- V. **Aprendizaje profundo o Deep Learning:** Es un tipo de ML que imita la forma en que los humanos obtienen ciertos tipos de conocimiento. El aprendizaje profundo incluye estadísticas y modelos predictivos. Es extremadamente beneficioso para los científicos de datos que tienen la tarea de recopilar, analizar e interpretar grandes cantidades de datos; el aprendizaje profundo hace que este proceso sea más rápido y sencillo (Tech Target, 2021). El Deep Learning lleva a cabo el proceso de ML usando una red neuronal artificial que se compone de un número de niveles jerárquicos. En el nivel inicial de la jerarquía, la red aprende algo simple y luego envía esta información al siguiente nivel. El siguiente nivel toma esta información sencilla, la combina, compone una información un poco más

compleja, y se lo pasa al tercer nivel, y así sucesivamente (Indra Sistemas S.A., 2021).

1.1.2 Machine Learning y su aplicación en el Mantenimiento Predictivo

Una buena gestión de mantenimiento supone mejorar la condición del equipo, reducir las tasas de fallas del equipo y minimizar los costos de mantenimiento, al tiempo que se maximiza la vida útil del equipo. La estrategia de aplicar un tipo de Mantenimiento Preventivo (PvM) basado en el modelo de Weibull, permite anticiparse a la falla resultando en un intercambio de repuestos con vida útil remanente media (MTBF). Por otro lado, el Mantenimiento Predictivo (PdM) utiliza herramientas predictivas para determinar cuándo son necesarias las acciones de mantenimiento. Se basa en el monitoreo continuo de la integridad de la máquina o de un proceso, lo que permite realizar estas acciones solo cuando es necesario. Además, permite la detección temprana de fallas gracias a herramientas predictivas basadas en datos históricos, como lo son las técnicas de ML. Generar modelos de ML para realizar predicciones sugiere tomar decisiones basadas en un conjunto de datos, logrando así obtener mayor beneficio en ahorros debido a la implementación de estos modelos. Por este hecho, la estrategia PdM es la que más destaca entre las demás, y llama la atención en la era de la Industria 4.0 por su capacidad de optimizar el uso y gestión de activos. Sus ventajas incluyen: maximizar el tiempo de uso y operación de los equipos, retrasar y/o reducir las actividades de mantenimiento y reducir los costos de materiales y mano de obra. El ML se ha convertido en una herramienta poderosa para desarrollar algoritmos predictivos inteligentes en muchas aplicaciones. Los modelos de ML tienen la capacidad de manejar datos multivariados de gran dimensión, y de extraer relaciones ocultas dentro de los datos en entornos complejos y dinámicos, como lo son los entornos industriales. Sin embargo, el rendimiento de estas aplicaciones depende de la elección adecuada de la técnica de ML (Carvalho, 2019).

Basados en la investigación realizada por (Carvalho, 2019) para la predicción de eventos de falla, se aplicará en el Capítulo 3 un algoritmo de ML supervisado de regresión, a una serie de tiempo correspondiente a un registro de fallas de un equipo industrial crítico, perteneciente a una planta de producción de celulosa. Este será el conjunto de datos de entrada con el cual se realizará el entrenamiento y la predicción del próximo evento de falla.

1.2 CASO DE ESTUDIO

1.2.1 Descripción del proceso

Para la obtención de productos derivados de la madera en una planta de celulosa, previamente se requiere que ésta sea tratada en diferentes procesos, hasta lograr extraer la celulosa de sus fibras, con la cual se produce posteriormente el papel y otros subproductos derivados. El proceso Kraft, también conocido como pulpeo Kraft, es usado en la producción de pulpa o pasta de celulosa. La planta en estudio tiene como producción anual, por diseño, 1.000.000 de toneladas de celulosa seca, y su proceso completo se puede representar mediante la siguiente imagen (Figura 1-2):

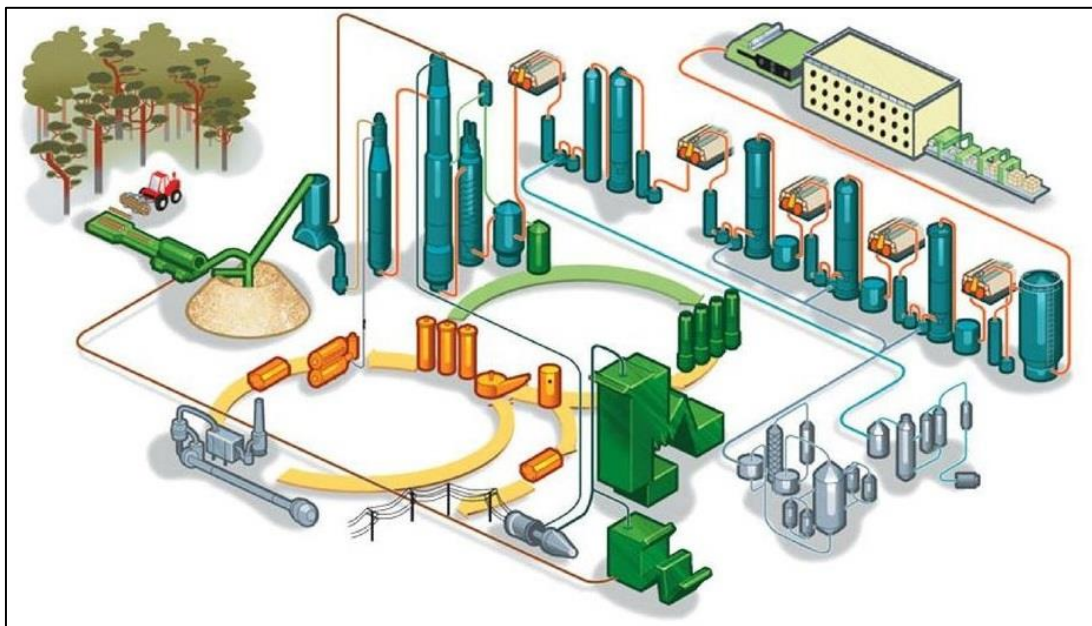


Figura 1-2. Proceso de producción de celulosa Kraft.

Fuente: Apuntes Ingeniería de Confiabilidad I (ARÁNGUIZ, 2021).

1.2.2 Descripción del equipo

El sistema Power Feed de una planta celulosa, es el primero dentro de la producción. Este cumple la función de alimentar a todo el proceso con troncos de madera cortada en bruto, mediante un transportador de tambor hacia el proceso de descortezado. Se transportan rollizos de madera de Pino Radiata a través de 9 cadenas con zapata, las

cuales se encuentran sobre una meza de concreto con cubierta de acero como se observa en las Figuras 1-3 hasta la Figura 1-6 respectivamente, mostradas a continuación:

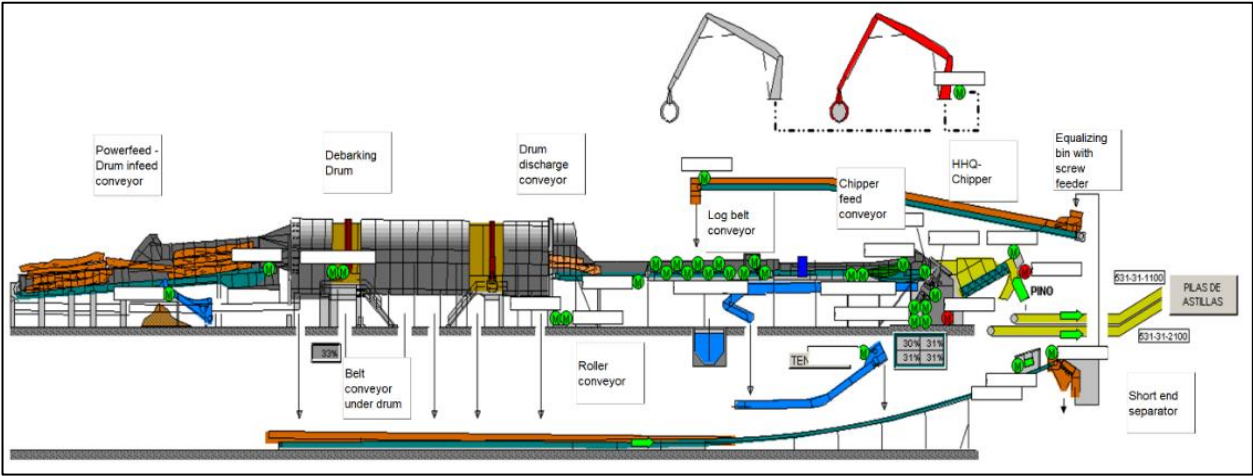


Figura 1-3. Disposición sistema de alimentación Power Feed. A la izquierda se muestra el equipo en estudio, el cual consiste en un transportador de tambor transportando rollizos de Pino radiata mediante cadenas con zapata, hacia el proceso de descortezado y astillado.

Fuente: Apuntes Ingeniería de Confiabilidad I (ARÁNGUIZ, 2021).

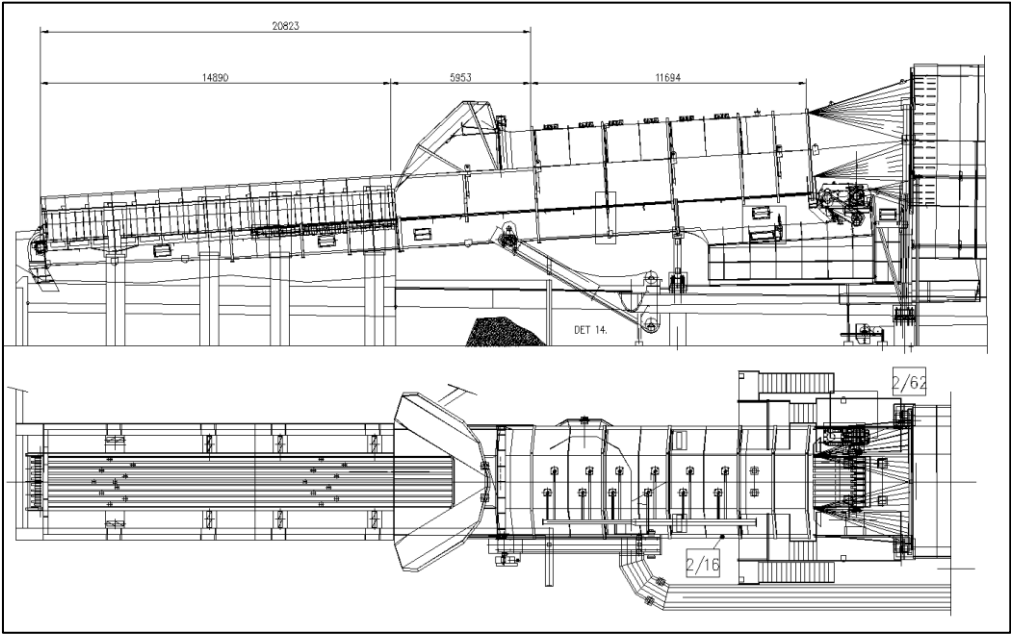


Figura 1-4. Plano detalle y dimensiones generales del sistema de alimentación Power Feed. El equipo tiene un largo y un ancho aproximados de 32 por 6 metros respectivamente.

Fuente: Apuntes Ingeniería de Confiabilidad I (ARÁNGUIZ, 2021).



Figura 1-5. Vista interior, sistema de alimentación Power Feed. Acá se pueden apreciar las 9 cadenas con zapata encargadas de transportar los rollizos en bruto de Pino radiata mediante un transportador de tambor hacia el proceso de descortezado y astillado.

Fuente: Apuntes Ingeniería de Confiabilidad I (ARÁNGUIZ, 2021).



Figura 1-6. Vista extendida de las cadenas con zapata del sistema de alimentación Power Feed.

Fuente: Apuntes Ingeniería de Confiabilidad I (ARÁNGUIZ, 2021).

1.2.3 Descripción de la serie de datos de entrada

En este estudio, se analizará el registro diario de las detenciones no programadas del sistema Power Feed, producidas por el transportador de tambor, el cual es uno de los equipos críticos dentro del proceso de producción de celulosa. En adelante denominado como “equipo 1001”. Se trabajará, en base a la información disponible sobre el registro de los tiempos de funcionamiento diarios, en un período comprendido entre el 06-01-2018 y el 10-07-2018. Se cuenta con total de 186 muestras (serie de datos de entrada representados en la Tabla 1-1), las cuales se utilizarán para implementar la metodología de predicción de eventos de falla aplicando modelos de regresión y ML, con el fin de mejorar la precisión con la que se programan las mantenciones preventivas del equipo 1001.

Tabla 1-1. Registro de tiempos de funcionamiento (en horas) de los primero cinco días de operación del equipo 1001

i	Fecha	T fun [h]
1	06-01-2018	20,42
2	07-01-2018	15,67
3	08-01-2018	23,50
4	09-01-2018	22,00
5	10-01-2018	22,82

Fuente: Base de datos del equipo 1001.

Se espera que la implementación de un modelo de predicción logre disminuir los costos de ineficiencia debido a detenciones no programadas de la planta de celulosa. Para el análisis final de la evaluación del modelo propuesto, se debe tener en cuenta que el costo de ineficiencia para ese período fue de 7.694,4 [USD/h].

1.3 MODELO CLÁSICO DE PREDICCIÓN

1.3.1 Modelo de Uso de vida de Weibull

En (Baptista, 2017) se define el modelo de uso de vida o Life Usage (LU), como un modelo probabilístico que tiene por objetivo determinar la esperanza de vida media o esperanza de falla de un equipo, utilizando una distribución de Weibull. Se ha utilizado ampliamente en el área de confiabilidad, principalmente por su capacidad para tratar con tamaños de muestra pequeños. En este modelo, los tiempos de funcionamiento del equipo se ajustan a una distribución de probabilidad para caracterizar el comportamiento de falla. El modelo se basa en una distribución continua, y permite modelar: el comportamiento de probabilidad instantánea de fallas $f(t)$, Confiabilidad $R(t)$, Mantenibilidad $F(t)$, Esperanza de funcionamiento $E(MTBF)$ y la Tasa de fallas $\lambda(t)$, mediante los parámetros Alfa (α), Beta (β) y Gamma (γ).

La función de densidad o probabilidad de falla instantánea, también llamada probabilidad de falla instantánea es la probabilidad de que un componente o sistema falle dentro de un instante predefinido a partir del instante de inicio de servicio. La función de densidad de falla propuesta por el modelo de Weibull consta de tres parámetros: Alfa (α), Beta (β) y Gamma (γ), y se define como:

$$f(t) = \begin{cases} \frac{\beta}{\alpha} \left(\frac{t - \gamma}{\alpha} \right)^{\beta-1} e^{-\left(\frac{t - \gamma}{\alpha} \right)^{\beta}} & t \geq 0, \\ 0 & t < 0 \end{cases} \quad (1)$$

Donde alfa (α) es el factor de escala, beta (β) es el factor de forma y gamma (γ), es el factor de localización o tiempo libre de fallas. La distribución de Weibull de dos parámetros se obtiene cuando " γ " se pone a cero, hecho que no afecta en el análisis de datos propuesto para este proyecto de título.

En un caso real, el estudio de la confiabilidad también se puede realizar mediante una distribución de Weibull si se tienen los parámetros alfa (α), beta (β) y gamma (γ), en donde la teoría plantea que está relacionada, además, con la Mantenibilidad o función de probabilidad acumulada de fallas $F(t)$, como se muestra en la ecuación (2):

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

En un modelo de uso de vida basado en la distribución de Weibull, el tiempo fijo entre dos acciones de mantenimiento preventivo generalmente se elige en función del parámetro de escala (α), según la criticidad y riesgo del equipo. Este parámetro, generalmente designado como la vida característica del equipo, es importante ya que corresponde a la esperanza de vida media o esperanza del tiempo medio entre fallas (MTBF) sin ajuste. Sin embargo, un MTBF ajustado según una distribución de Weibull se define como:

$$E(MTBF) = \gamma + \alpha * \Gamma \left[1 + \frac{1}{\beta} \right] \tag{3}$$

La tasa de fallas es la probabilidad de tener una falla en un sistema o componente a condición de que éste haya sobrevivido hasta el instante anterior. De acuerdo con la teoría de la confiabilidad, las fallas se pueden agrupar en tres clases distintas como se muestra en la Figura 1-7 (curva de la bañera), según el comportamiento de la función de tasa de fallas $\lambda(t)$ indicada en la ecuación (4):

$$\lambda(t) = \frac{\beta}{\alpha} \left(\frac{t}{\alpha} \right)^{\beta-1} = \frac{f(t)}{R(t)} \tag{4}$$

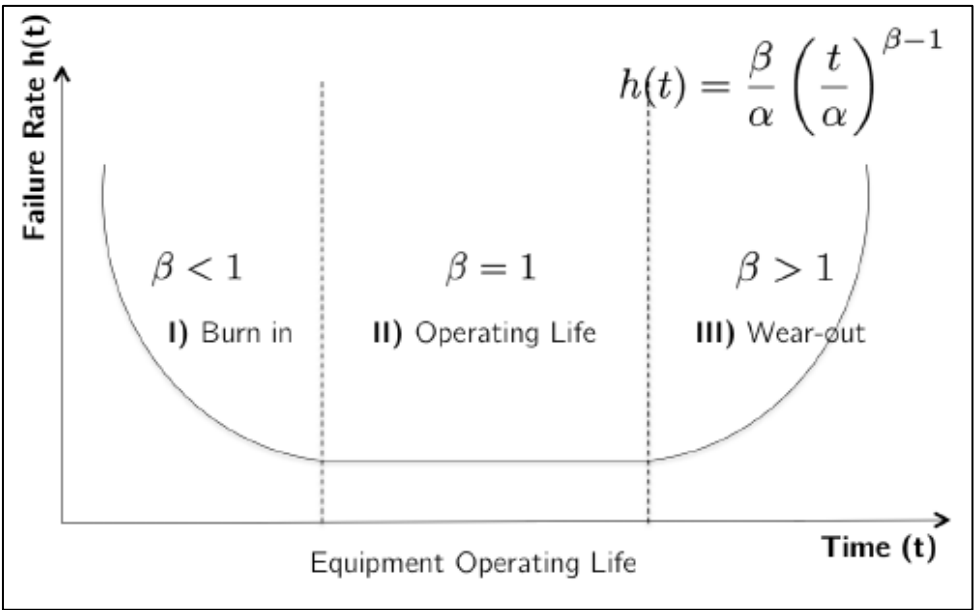


Figura 1-7. Representación de la curva de la bañera

Fuente: Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA modeling (Baptista, 2017).

La Figura 1-7 muestra la curva de la bañera, la cual representa los tres estados de falla en los que puede encontrarse un equipo o sistema y, en donde además las distribuciones de Weibull, en función del parámetro β , se pueden resumir en las condiciones:

- Si $\beta < 1,0$ indica mortalidad infantil y una etapa de falla temprana (I): esta etapa temprana describe el intervalo de tiempo en el que el comportamiento de falla no está lo suficientemente desarrollado debido a influencias desconocidas, como problemas de diseño o configuración incorrecta. La tasa de fallas es decreciente.
- Si $\beta = 1,0$ indica una etapa de fallas aleatorias (II): esta segunda etapa representa el período donde la falla tiene una tasa constante. Valores de β muy cercanos a 1,0 indican que el equipo o sistema se encuentra en su etapa de vida operativa.
- Si $\beta > 1,0$ indica una etapa de fallas por desgaste (III): esta tercera etapa ocurre en el fin del ciclo de vida del equipo. En él, el equipo tiende a tener más fallas por desgaste debido al proceso de envejecimiento, por ende, la tasa de fallas es creciente.

La gran desventaja del modelo LU de Weibull es que entrega valores de predicciones en intervalos de tiempo fijo basados en probabilidades, lo que resulta en una serie de mantenciones a tiempos constantes sin considerar otras variables que pudiesen afectar al proceso, como se muestra en la Figura 1-8. En el modelo propuesto en el Capítulo 2, se analiza el tiempo como una variable basada en función de sus valores pasados y del ruido, por ende, se calcula de manera distinta.

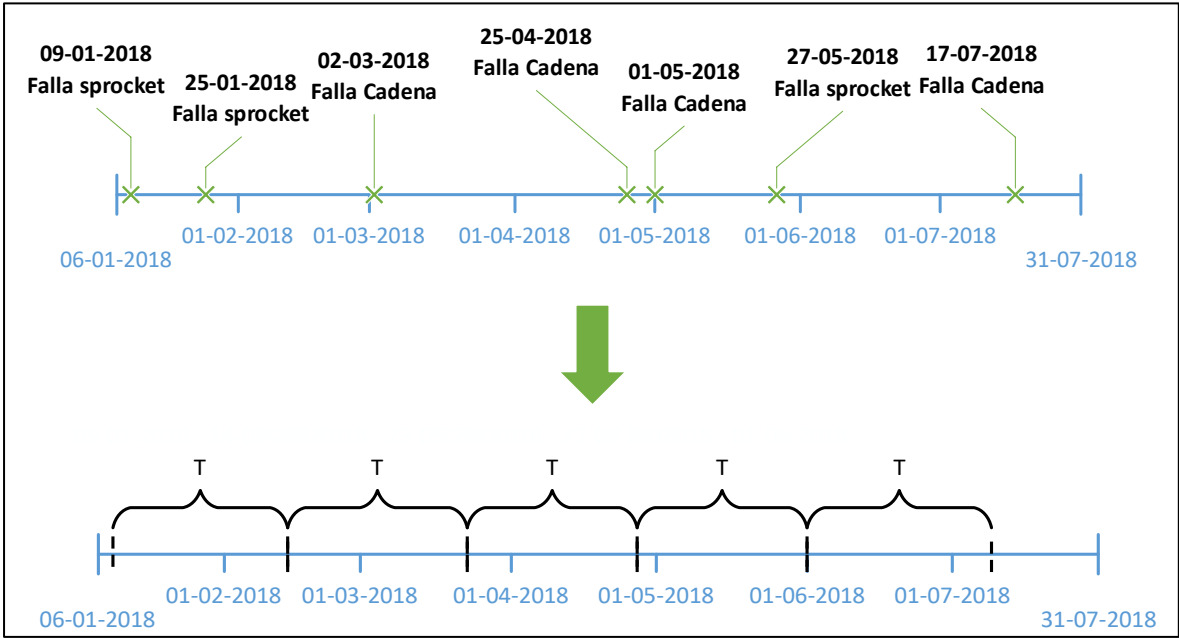


Figura 1-8. Representación gráfica del funcionamiento del modelo LU de Weibull

Fuente: Elaboración propia. Basado en Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA modeling (Baptista, 2017).

1.3.2 Aplicación del modelo de uso de vida de Weibull

Para realizar una predicción mediante el modelo de uso de vida o LU de Weibull, primero se deben calcular los parámetros alfa, beta y gamma en base al conjunto de datos ordenando los datos de la variable de interés de menor a mayor como se muestra en la Tabla 1-2:

Tabla 1-2. Orden ascendente de los primeros cinco valores de la serie de tiempos de funcionamiento “tfun [h]”, para el cálculo de parámetros del modelo de LU de Weibull

t	tfun [h]
1	6,00
2	9,33
3	10,08
4	10,17
5	10,58

Fuente: Elaboración propia. Base de datos del equipo 1001.

Luego, para cada tiempo de funcionamiento (ordenados de manera ascendente), se calcula la Probabilidad de falla acumulada $F(t)$ y su Confiabilidad $R(t)$, en función del valor del índice “t”. Además, se debe realizar una transformación logarítmica a los tiempos de funcionamiento “tfun” y a los valores de Confiabilidad $R(t)$, como se muestra en la Tabla 1-3, y luego graficar estos valores (Figura 1-9).

Tabla 1-3. Cálculo de probabilidad acumulada de falla $F(t)$, Confiabilidad $R(t)$ y transformación logarítmica aplicado a los primeros cinco tiempos de funcionamiento para el cálculo de los parámetros de Weibull

t	tfun [h]	F(t)	R(t)	Ln(tfun)	Ln(-Ln(R(t)))
1	6,00	0,0038	0,9962	1,792	-5,583
2	9,33	0,0091	0,9909	2,234	-4,693
3	10,08	0,0145	0,9855	2,311	-4,227
4	10,17	0,0198	0,9802	2,319	-3,910
5	10,58	0,0252	0,9748	2,359	-3,668

Fuente: Elaboración propia. Base de datos del equipo 1001.

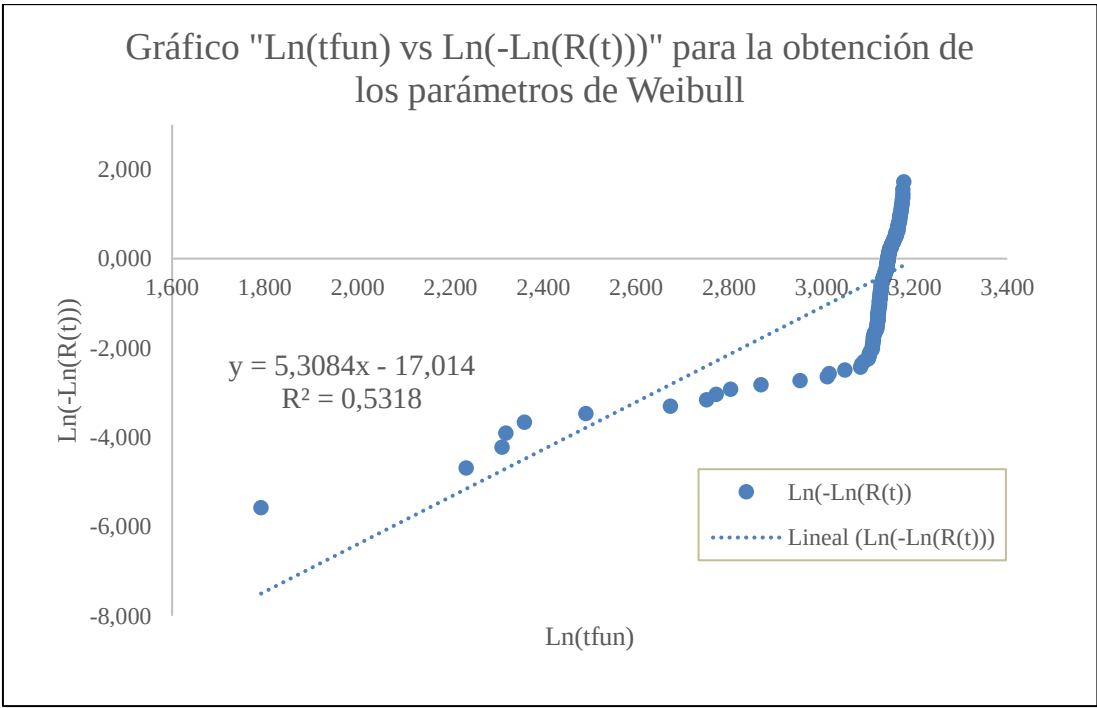


Figura 1-9. Gráfico “Ln(tfun) vs Ln(-Ln(R(t)))” para la obtención de los parámetros de Weibull

Fuente: Elaboración propia.

A. Parámetros de Weibull

Realizado esto, se procede a calcular los parámetros del modelo de uso de vida de Weibull. Por definición, β corresponde a la pendiente de la recta generada por la regresión lineal ajustada a los valores de “Ln (tfun)” (eje X) y “Ln(-Ln(R(t)))” (eje Y) mostrados en la Figura 1-9, en donde $\beta = 5,308$. Mientras que el valor de alfa α se obtiene de la intersección “a” de esta. Donde:

$$a = -\beta * Ln(\alpha) \tag{5}$$

Con la intersección $a = 17,014$ y la pendiente $\beta = 5,308$ conocidos podemos obtener el valor de alfa, donde:

$$\alpha = e^{\left(\frac{-a}{\beta}\right)} \tag{6}$$

por lo tanto, los valores de los parámetros de Weibull corresponden a los mostrados en la Tabla 1-4:

Tabla 1-4. Valores de los parámetros del modelo de predicción clásico: Uso de vida de Weibull, obtenidos desde el conjunto de datos de la Tabla 1-1

Parámetros modelo de Weibull	
α [h]	24,66
β [-]	5,308
γ [-]	0

Fuente: Elaboración propia. Base de datos del equipo 1001.

B. Esperanza del tiempo medio entre fallas

Con los parámetros de Weibull obtenidos, se determina que el tiempo medio entre fallas o, tiempo fijo entre dos acciones de mantenimiento preventivo es de 22,72 hr (Tabla 1-5), según el siguiente cálculo aplicando la ecuación (3), donde:

$$E(MTBF) = 0 + 24,66 * \Gamma \left[1 + \frac{1}{5,308} \right] = 22,72 \text{ hrs}$$

Tabla 1-5. Esperanza de vida media del equipo 1001, basado en el modelo de Weibull

E(MTBF) [h]	22,72
-------------	-------

Fuente: Elaboración propia. Base de datos del equipo 1001.

Por otro lado, el modelo de Uso de vida de Weibull nos entrega información relacionada con la Confiabilidad en base a sus parámetros ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$).

C. Probabilidad instantánea de fallas $f(t)$

Esta nos indica la probabilidad de que un equipo funcione en el instante de tiempo “t”, a condición de que el equipo se encuentre operativo en el instante “t” anterior. Como es posible apreciar en la Figura 1-10, a simple vista se observa que el equipo tiene una media de funcionamiento aproximada de 23 horas, lo cual coincide con el valor MTBF obtenido. Cercano a este tiempo de operación, el equipo 1001 tiene una probabilidad de fallar instantáneamente de $f(t) = 0,08$.

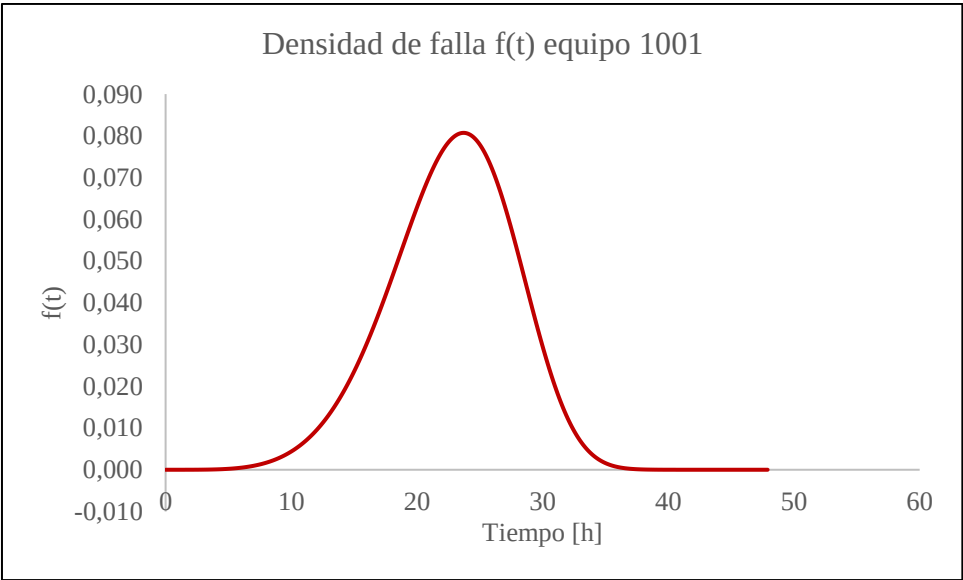


Figura 1-10. Densidad de falla o probabilidad instantánea de fallas $f(t)$ del equipo 1001, obtenida según parámetros ($\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$) del modelo LU de Weibull

Fuente: Elaboración propia. Base de datos del equipo 1001.

D. Confiabilidad y Mantenibilidad

La Confiabilidad $R(t)$ nos indica la probabilidad de que un equipo se encuentre operativo, en el instante en el que se desea ser utilizado para que cumpla la función para la cual se requiere. Por ende, basado en los parámetros calculados, se puede inferir que el equipo 1001, durante sus primeras 10 horas de funcionamiento tiene una probabilidad del 100% de encontrarse operativo, como se muestra en la Figura 1-11, a pesar, y entendiendo de que siempre pueden ocasionarse hechos imprevistos. Por otro lado, aproximadamente a las 30 horas de funcionamiento continuo, el equipo 1001 presenta una probabilidad de falla cercana al 100%. Por lo tanto, el área de mantenimiento dentro de la planta de celulosa debería fijar las actividades de mantenimiento en el rango de operación entre 10 a 30 horas. Basado en los parámetros de Weibull calculados ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$), la Confiabilidad y Mantenibilidad se pueden representar de la siguiente manera:

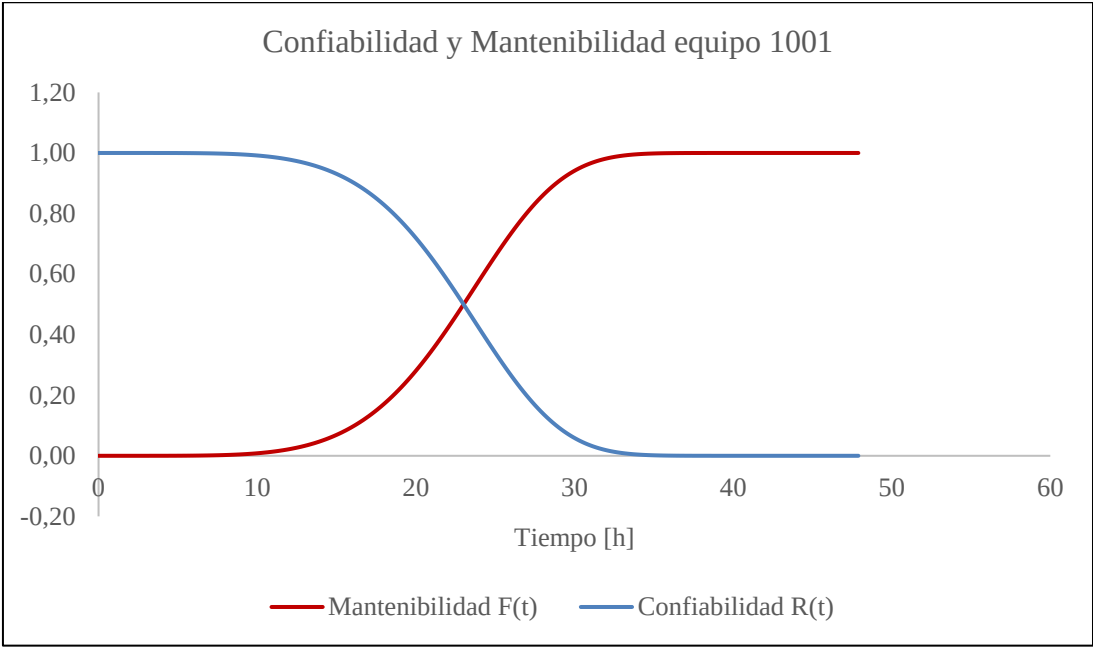


Figura 1-11. Confiabilidad y Mantenibilidad equipo 1001, obtenidas según parámetros ($\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$) del modelo LU de Weibull

Fuente: Elaboración propia. Base de datos del equipo 1001.

E. Tasa de fallas:

La tasa de fallas es un indicador que nos permite estimar la cantidad de fallas ocurridas en un intervalo de tiempo “t”. De acuerdo con el gráfico de la Figura 1-12, podemos observar que la curva tiene una tendencia positiva o tasa de fallas $\lambda(t)$ creciente, lo cual nos indica que después de 20 horas de operación continua, el equipo empieza a generar detenciones, con una probabilidad que va en aumento a medida que se superan las 20 horas. Esto se condice con el valor de $\beta > 1,0$. Por lo tanto, podemos determinar que el equipo 1001 del sistema Power Feed se encuentra en una etapa de fallas por desgaste, o sea, en el fin del ciclo de vida del equipo.

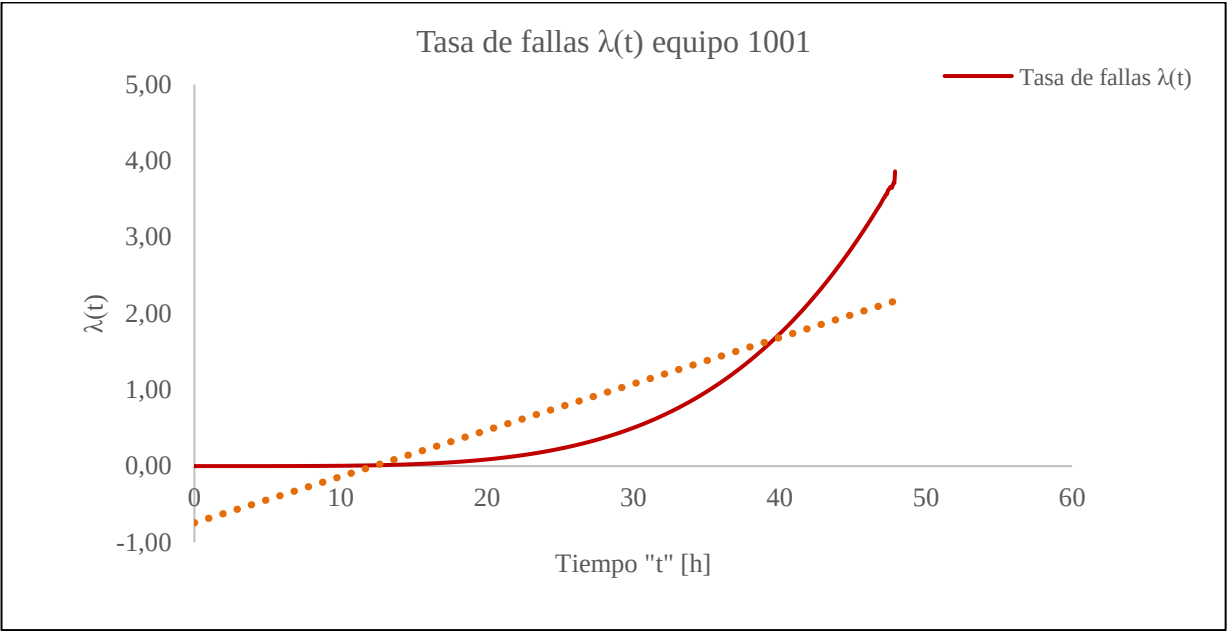


Figura 1-12. Tasa de fallas de equipo 1001, obtenida según parámetros ($\alpha = 24,66$; $\beta = 5,31$; $\gamma = 0$) del modelo LU de Weibull

Fuente: Elaboración propia. Base de datos del equipo 1001.

Todo indica que, cercano a las 20 horas de operación del equipo 1001, la planta de celulosa debería fijar sus actividades de mantenimiento, en el rango de operación entre 10 a 30 horas o, más bien dicho, a intervalos de 22,72 horas el modelo de uso de vida de Weibull recomienda realizar las mantenciones preventivas. Si bien esta información es relevante, en el capítulo siguiente se presenta un modelo capaz de realizar una predicción aún más precisa y exacta.

CAPÍTULO 2: METODOLOGÍA PROPUESTA PARA LA MEJORA EN LA PREDICCIÓN

2 METODOLOGÍA PROPUESTA PARA LA MEJORA EN LA PREDICCIÓN

2.1 MARCO DE TRABAJO PROPUESTO

Luego de enmarcado el problema se define a continuación, la metodología de predicción propuesta mediante en el siguiente marco de trabajo:

De manera general, la metodología consiste en aplicar y proponer un modelo de predicción a una serie temporal de eventos como datos de entrada para obtener resultados de la predicción, como se muestra en la Figura 2-1. Luego, se mide el error de la predicción comparando la serie de datos de entrada original con la serie de datos predichos, mediante las métricas de rendimiento. Si los resultados de las métricas son satisfactorios, o sea, tiene errores mínimos, podemos decir que el modelo de predicción propuesto es aplicable y apropiado para su uso. Sin embargo, antes de aplicar un modelo de predicción, primero es necesario explorar la serie de datos de entrada y obtener una idea general sobre su comportamiento, como se describe en la Sección 2.1.1.

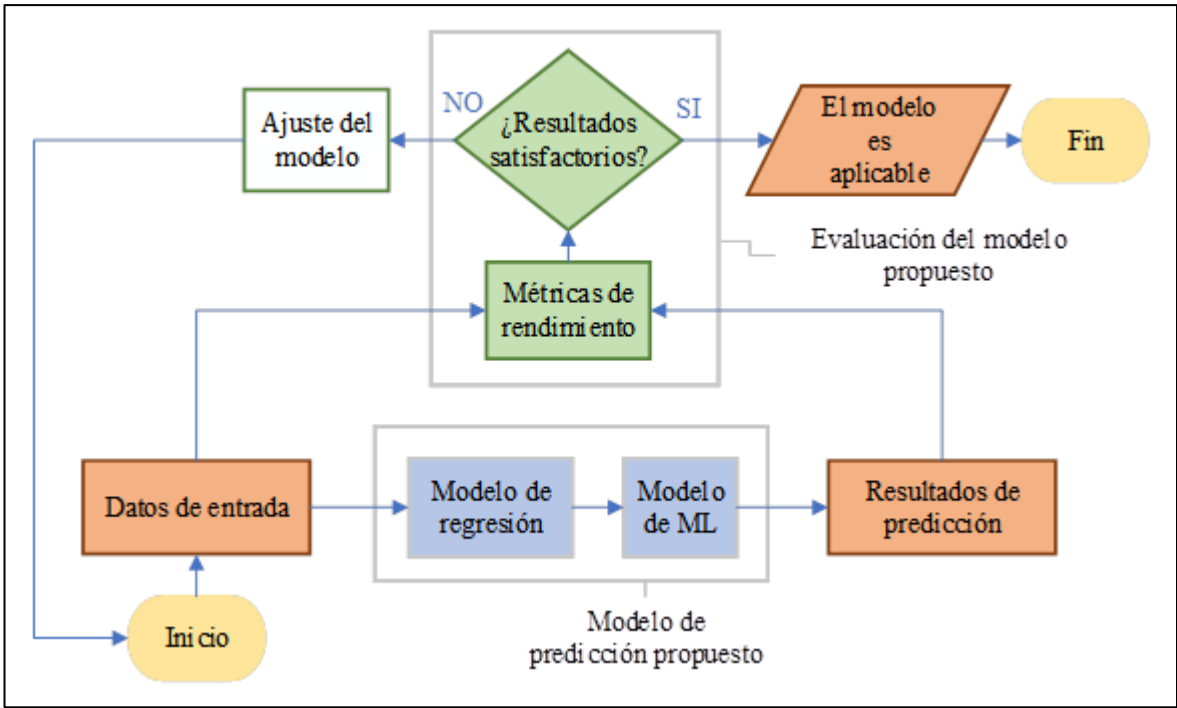


Figura 2-1. Diagrama del marco de trabajo propuesto.

Fuente: Elaboración propia. Basado en Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA modeling (Baptista, 2017)

2.1.1 Exploración y preprocesamiento de datos

Evaluar un modelo sin realizar previamente un análisis y preprocesamiento de datos, genera el riesgo de obtener resultados de predicción erróneos. En esta etapa previa, se busca que las propiedades de una sección de datos sean muy parecidas a las propiedades de las otras secciones de datos. Esto se realiza mediante la descripción, visualización e interpretación de los patrones implícitos propios de tendencia, estacionalidad y ruido de la serie de datos, como se muestra, por ejemplo, en la Figura 2-2 mostrada a continuación:

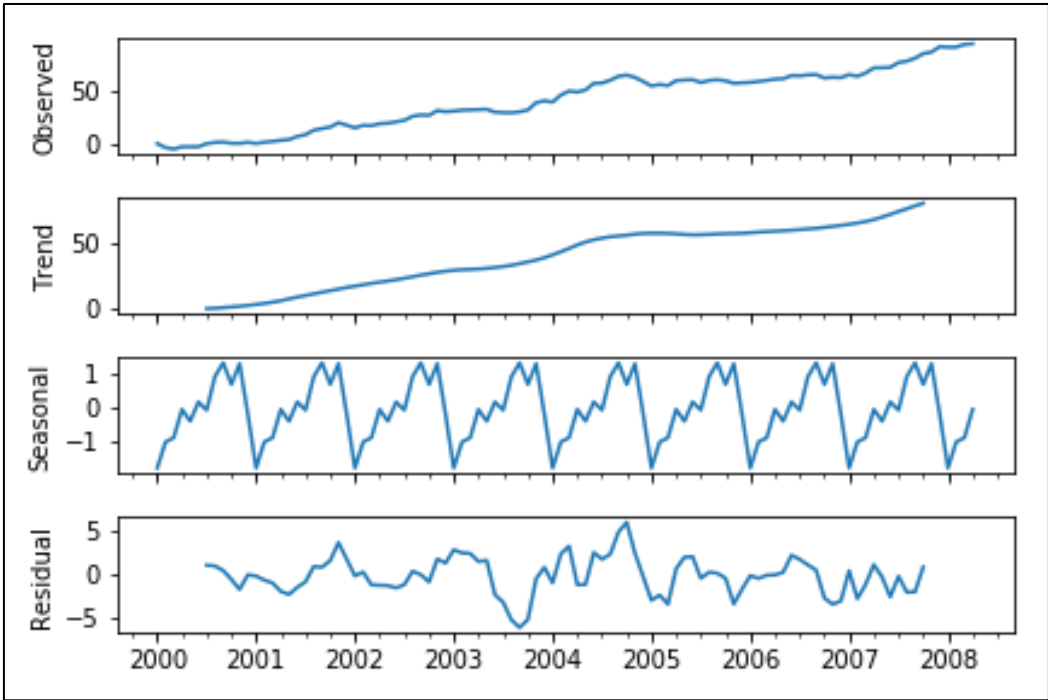


Figura 2-2. Ejemplo de descomposición de las componentes de tendencia, estacionalidad y errores residuales de una serie de datos de entrada

Fuente: Machine Learning para series temporales con ARIMA, SARIMA.
Coursera Project Network

Por lo tanto, podemos decir que, para realizar correctamente el procedimiento propuesto, en primer lugar, se debe cumplir con la condición de que la serie de datos de entrada sea estacionaria, esto quiere decir, que no existan cambios sistemáticos ni en la media (sin tendencia), ni en sus variaciones (sin varianza), ni en sus fluctuaciones periódicas (covarianza). Además, se requiere que la condición de la distribución del ruido sea normal. La alta presencia de variación de estas características en la serie de datos de entrada disminuye la capacidad de predicción del modelo. Una serie de tiempo que cuenta con estas características, se puede decir que no es estacionaria. La covarianza indica el

nivel en el que dos variables varían juntas. En la Figura 2-3, se muestra una manera de medir la fuerza de asociación lineal entre dos variables es usando el coeficiente de correlación. Cuando hablamos de correlación, nos referimos al concepto de covarianza en unidades estándar.

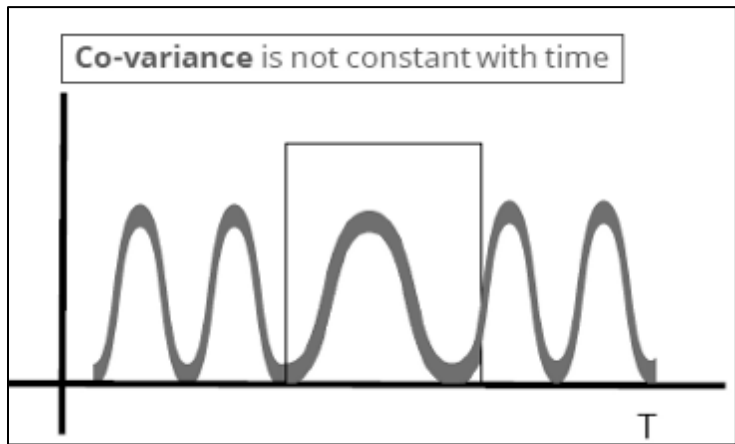


Figura 2-3. Descripción gráfica de una covarianza irregular o no constante

Fuente: Machine Learning para series temporales con ARIMA, SARIMA.
Coursera Project Network

La correlación indica la cantidad de relación de dependencia entre dos variables. Se mide en un rango de $[-1, 1]$, en donde los valores cercanos a cero indican un bajo nivel de dependencia, mientras que los valores cercanos a -1 y 1 indican una alta dependencia (negativa o positiva, respectivamente). Se espera que las variables sean independientes o que tengan un nivel de correlación cero o cercano a cero, ya que los altos niveles de dependencia también generan errores de pronósticos. Sin embargo, es posible realizar transformaciones a la serie de datos de entrada para que pueda cumplir con las condiciones de estacionariedad. Para el caso del ruido, como se menciona anteriormente, simplemente se espera que, idealmente, tenga una distribución normal, ya que, es un tipo de error irreducible (propio de la serie), por lo tanto, no es posible realizar una transformación. El análisis de todas estas variables (tendencia, varianza y covarianza), indica si la serie de tiempo es o no estacionaria y, en caso de no serlo, se deben realizar las transformaciones necesarias a la serie para que ésta sea, al menos, lo más estacionaria posible antes de pasar a la siguiente etapa. Por otro lado, hoy en día, es posible aplicar tests de estacionariedad (o más bien, normalidad), a una serie de datos mediante el uso de lenguaje de programación Python como, por ejemplo, el test de Dickey Fuller, Ljung-Box, Chi-cuadrado, D’Agostino y Pearson, entre otros.

2.1.2 Entrenamiento y prueba de un modelo

La única forma de saber qué tan bien se generalizará un modelo a nuevos casos es probarlo sobre datos nuevos. El error de generalización (o error fuera de muestra) indica qué tan bien funcionará un modelo a medida que se agregan datos nuevos. Una solución común a este problema es tener un segundo conjunto llamado conjunto de validación o prueba. Por lo tanto, los pasos a seguir se pueden definir como sigue:

1. Entrenamiento: Se deben entrenar varios modelos con varios parámetros e hiperparámetros (para el modelo de regresión y el modelo basado en datos, respectivamente), usando el conjunto de entrenamiento, siguiendo el marco de trabajo propuesto en la Figura 2-1.
2. Prueba: Una vez obtenidos los parámetros e hiperparámetros (para el modelo de regresión y el modelo basado en datos, respectivamente), que mejor funcionan con los datos de entrenamiento, se ejecuta el modelo optimizado en el conjunto de prueba. Si los resultados de rendimiento son satisfactorios, se puede decir que el modelo aprendió bien y es aplicable para realizar predicciones sobre nuevos datos.

De esta manera, el marco de trabajo propuesto consta, además, de dos etapas o fases: la fase de entrenamiento y la fase de prueba. En la fase de entrenamiento, se espera que el modelo aprenda el comportamiento sobre la serie de datos, para que luego sea capaz de replicarlo en función de sus patrones implícitos con el menor error posible durante la fase de prueba. Es común utilizar el 80% de los datos para el conjunto de entrenamiento y el 20% restante para el conjunto de prueba. Al procedimiento de optimización de hiperparámetros también se le conoce como “tuning” y consiste en un procedimiento de validación cruzada para la búsqueda de los mejores hiperparámetros para un modelo basado en datos, como lo es SVR. Este procedimiento puede ser realizado de forma automática mediante lenguaje de programación Python.

2.2 MODELOS PROPUESTOS PARA LA PREDICCIÓN

2.2.1 Autorregresive Integrated Moving Average

El modelo de regresión Autorregresive Integrated Moving Average (Media Móvil Autorregresiva Integrada), o ARIMA, es una metodología matemática que captura todo el historial pasado de eventos de fallas y proporciona un pronóstico del próximo evento en forma de función predictiva. Utiliza la serie de tiempo no solo en función de sus valores pasados, sino también como un promedio móvil del ruido pasado ponderado de manera desigual. Para estimar los parámetros del modelo ARIMA, ajustamos el modelo a una serie temporal univariada de eventos. El objetivo de tener un modelo ARIMA es tener la capacidad de realizar un aprendizaje secuencial: el modelo ARIMA captura el historial completo de eventos pasados y genera predicciones que ayudan al modelo SVR en el proceso de calcular sus predicciones finales. Para aplicar correctamente este modelo, el ruido dentro de la serie debe cumplir con la condición de ruido blanco (Z_t), donde $Z_t \sim Normal(0, \sigma_Z^2)$, es decir, que tenga una distribución normal de media cero y varianza constante igual a uno. Para lograr esto, es posible escalar o estandarizar la serie de datos mediante métodos de transformación como la normalización, diferenciación, métodos de escalado, entre otros.

Un proceso autorregresivo o $AR(p)$, se define como $\phi_p(B) X_t = Z_t$. Mientras que, por otro lado, un proceso de media móvil $MA(q)$ se define como $X_t = \theta_q(B) Z_t$. De la misma forma, un proceso de media móvil autorregresiva simple $ARMA(p, q)$ puede ser escrito como:

$$\phi_p(B) X_t = \theta_q(B) Z_t \quad (7)$$

donde $\phi_p(B)$ representa la suma de los coeficientes autorregresivos y $\theta_q(B)$ representa la suma de los coeficientes media móvil, los cuales se calculan respectivamente como:

$$\begin{aligned} \phi_p(B) &= 1 - \phi_1 B - \dots - \phi_p B^p \\ \theta_q(B) &= 1 + \theta_1 B + \dots + \theta_q B^q \end{aligned} \quad (8)$$

Por lo tanto, al reemplazar (8) en (7), el proceso $ARMA(p, q)$ (sin diferenciación), también se puede representar de la siguiente manera:

$$X_t = Z_t + \sum_{i=1}^q \theta_i Z_{t-i} + \sum_{i=1}^p \phi_i X_{t-i} \quad (9)$$

Además, un proceso $ARMA(p, q)$ que considere los efectos de la tendencia de una serie de datos, se le conoce como proceso $ARIMA(p, d, q)$. Acá, el parámetro “d” indica la cantidad de diferenciaciones aplicadas a la serie original X_t para que ésta se haga estacionaria. Se puede representar como:

$$\phi_p(B) Y_t = \theta_q(B) Z_t \quad (10)$$

donde Y_t es una serie X_t diferenciada, de manera tal que:

$$Y_t = \nabla^d X_t = (1 - B)^d X_t \quad (11)$$

y reemplazando (11) en (10) obtenemos:

$$\phi_p(B) (1 - B)^d X_t = \theta_q(B) Z_t \quad (12)$$

Por lo tanto, si ahora reemplazamos (8) en (12), la serie $ARIMA(p, d, q)$ que considera el efecto de la tendencia de una serie X_t , se puede representar como:

$$(1 - B)^d X_t = Z_t + \sum_{i=1}^q \theta_i Z_{t-i} + \sum_{i=1}^p \phi_i Y_{t-i} \quad (13)$$

Otra variante, se puede dar en un proceso $ARIMA(p, d, q)$ que considere los efectos de la estacionalidad. A este se le conoce como proceso $SARIMA(p, d, q, P, D, Q, s)$. Acá, se utilizan los parámetros del proceso $ARIMA(p, d, q)$, pero incluye parámetros autorregresivos, integrales y de media móvil estacionales representados por las letras mayúsculas P, D, Q y s , donde “s” representa la cantidad u orden de períodos observados. Un proceso $SARIMA(p, d, q, P, D, Q, s)$ tiene la estructura:

$$\Phi_P(B^s) \phi_p(B) \nabla_s^D Y_t = \Theta_Q(B^s) \theta_q(B) Z_t \quad (14)$$

donde Y_t es una serie X_t diferenciada de forma estacional de manera tal que:

$$\nabla_s^D = (1 - B^s)^D \quad (15)$$

y, reemplazando (11) y (15) en (14), se obtiene:

$$\Phi_P(B^s) \phi_p(B) (1 - B^s)^D Y_t = \Theta_Q(B^s) \theta_q(B) Z_t \quad (16)$$

$$\Phi_P(B^s) \phi_p(B) (1 - B^s)^D (1 - B)^d X_t = \Theta_Q(B^s) \theta_q(B) Z_t \quad (17)$$

donde $\Phi_P(B^s)$ y $\Theta_Q(B^s)$ son los coeficientes estacionales autorregresivos y de media móvil estacionales, respectivamente:

$$\Phi_P(B^s) = 1 - \Phi_1 B^s - \dots - \Phi_P B^{sP} \quad (18)$$

$$\Theta_Q(B^s) = 1 + \Theta_1 B^s + \dots + \Theta_Q B^{sQ}$$

Por ende, el proceso $SARIMA(p, d, q, P, D, Q, s)$ se define finalmente como:

$$(1 - B^s)^D (1 - B)^d X_t = Z_t + \sum_{i=1}^q \theta_i Z_{t-i} + \sum_{i=1}^p \phi_i Y_{t-i} + \sum_{i=1}^Q \Theta_i Z_{t-i} + \sum_{i=1}^P \Phi_i Y_{t-i} \quad (19)$$

Para seleccionar entre un modelo ARMA, ARIMA o SARIMA, se utilizará más adelante, un algoritmo formulado mediante lenguaje de programación Python para optimizar el proceso de selección de parámetros, el cual indicará de forma automática cuales son los mejores valores que se ajustan a los datos de entrada.

2.2.2 Support Vector Regression

El algoritmo de vectores de soporte de regresión o Support Vector Regression (SVR) es un modelo basado en datos el cual es, en esencia, un algoritmo de clasificación llamado Support Vector Machine (SVM) adaptado para regresión, pero ambos siguen el mismo procedimiento:

1. Crean hiperplanos que separa la muestra en clases de datos, como se muestra, por ejemplo, en la Figura 2-4 mostrada a continuación:

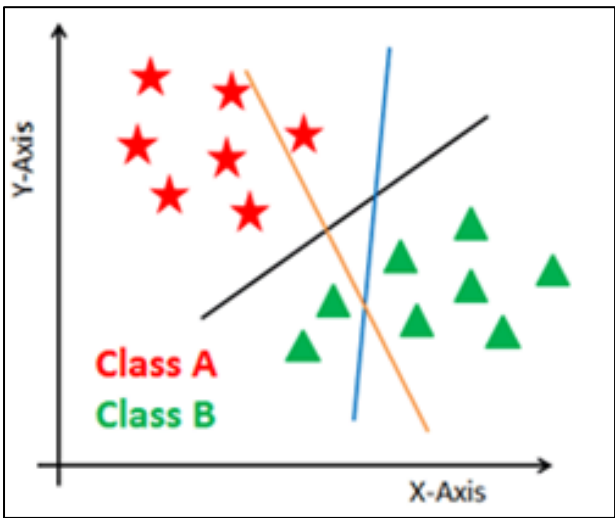


Figura 2-4. Ejemplo de los infinitos hiperplanos posibles

Fuente: <https://www.mygreatlearning.com/>

2. Comparan el margen de los hiperplanos y eligen el hiperplano con el margen más grande. El margen es la distancia más corta entre el hiperplano y los puntos de datos. Los puntos de datos en el borde y dentro del margen se denominan vectores de soporte (ver Figura 2-5).
3. Realizan una clasificación y/o predicción para los nuevos puntos de datos según el lado del hiperplano en el que se encuentren los nuevos datos.

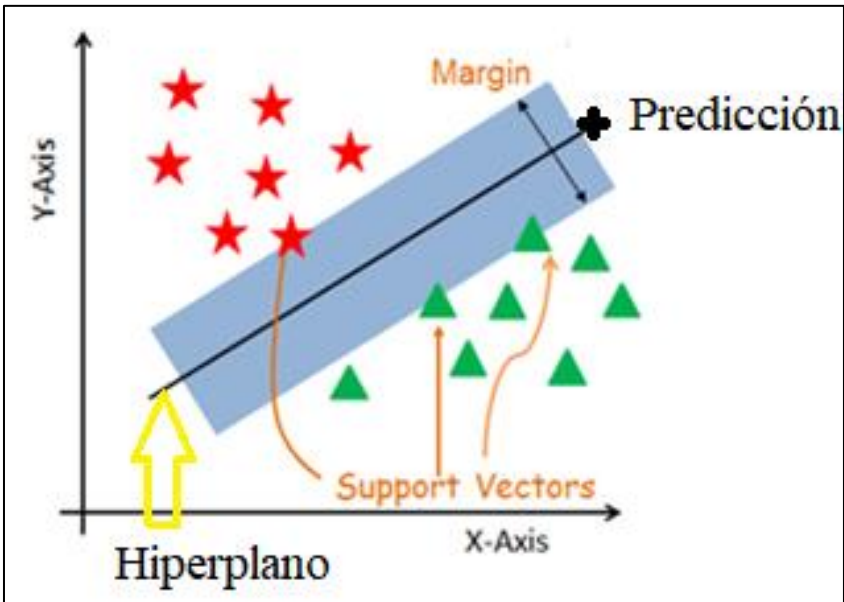


Figura 2-5. Ejemplo de hiperplano y margen suave, ajustado mediante vectores de soporte

Fuente: Elaboración propia. Basado en <https://www.mygreatlearning.com/>.

De esta manera, el algoritmo encuentra un hiperplano patrón, sobre el cual realiza una predicción del próximo evento de falla, como se puede observar también en la siguiente imagen (Figura 2-6):

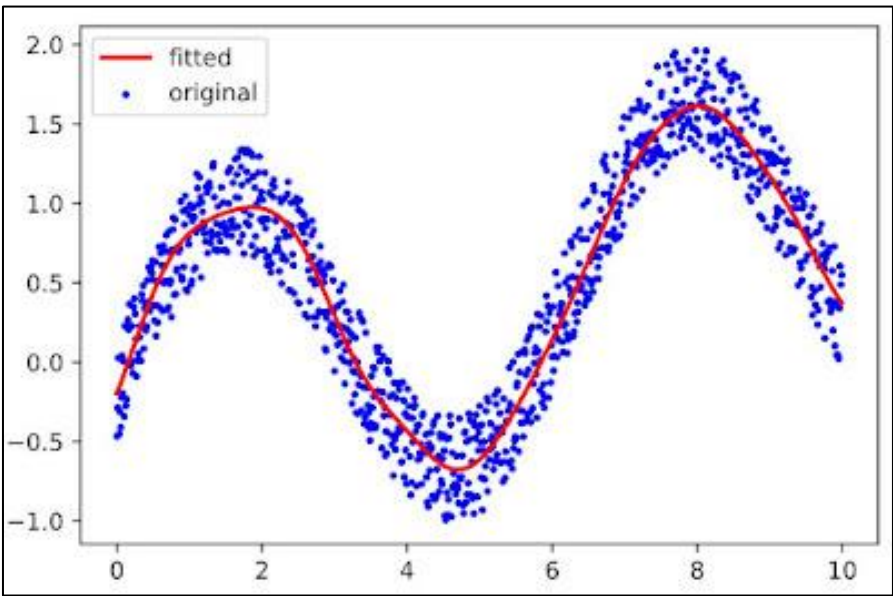


Figura 2-6. Ejemplo de aplicación de un modelo SVR con kernel RBF

Fuente: <https://itnext.io/>

Según la investigación realizada por (Qinming Liu, 2021), para realizar predicciones mediante un modelo de ML de SVR, se deben usar los datos originales de la serie de tiempos de funcionamiento, para ajustar una función continua que pueda reflejar la relación entre los valores de entrada y salida. Por otro lado, los parámetros clave de SVR incluyen una función kernel, el coeficiente de la función de pérdida insensible “epsilon” (ϵ), el parámetro de función de base radial “gamma” (σ) y el factor de penalización C :

- Kernel: La función kernel transforma el conjunto de datos de entrenamiento en dimensiones más altas para hacerlo linealmente separable. La función kernel predeterminada para la implementación de SVR en Python, es la función de base radial, más conocida como RBF o “rbf”. Sin embargo, también puede tomar valores como “linear”, “poly” o “sigmoid” y sus diferencias se pueden apreciar en los datos de ejemplo de la Figura 2-7, mostrada a continuación:

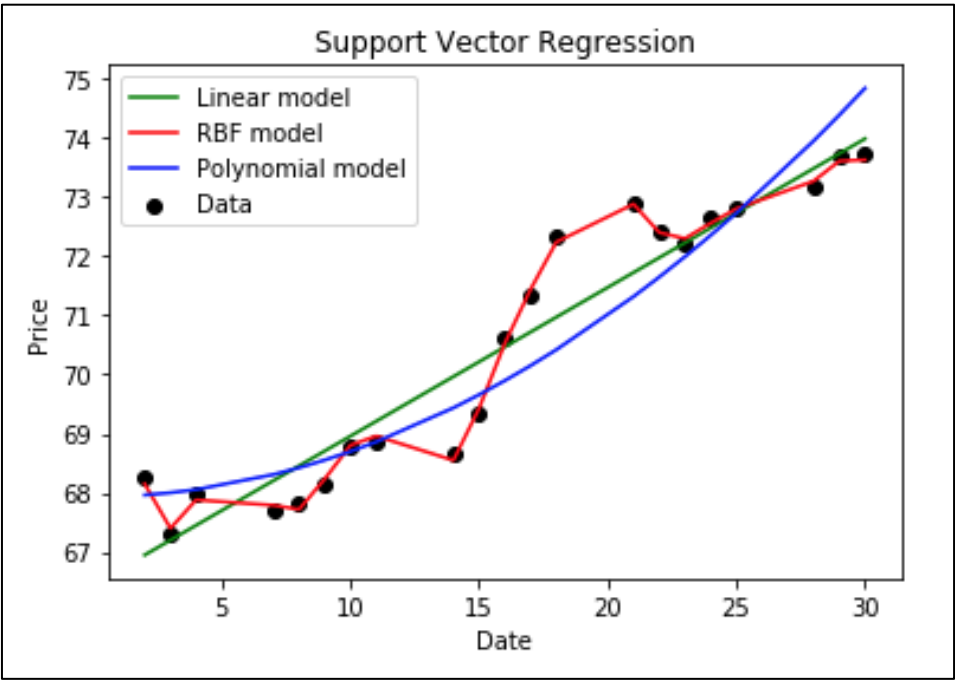


Figura 2-7. Ejemplo de distintos hiperplanos creados mediante SVR, ajustados a la serie de datos con kernels lineales y no lineales

Fuente: <https://itnext.io/>

- Epsilon (ϵ): representa el ancho de la zona insensible, también conocido como margen suave, y juega un papel decisivo en la cantidad de vectores de soporte y la capacidad de generalización del modelo. El valor ϵ puede mejorar la capacidad de

generalización; con el aumento de ϵ , el número de vectores de soporte disminuirá y la complejidad del cálculo algorítmico también se reducirá. Para un modelo SVR con kernel lineal, se puede observar su hiperplano y margen suave (o zona insensible) como se muestra en la siguiente imagen (Figura 2-8):

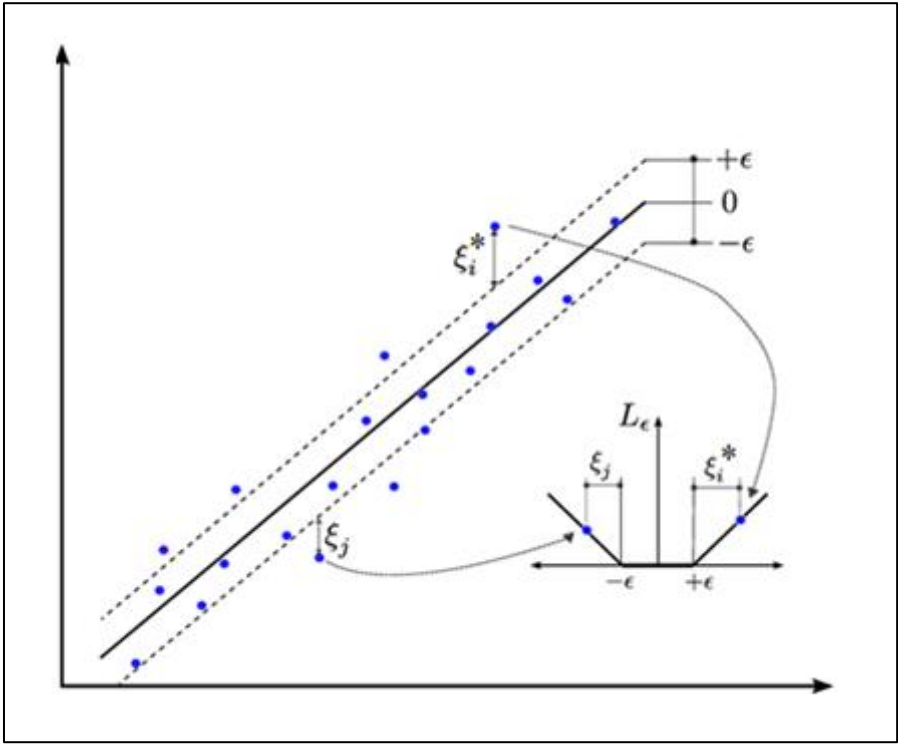


Figura 2-8. Representación del parámetro epsilon y su relación con la función de pérdida L_ϵ del modelo SVR

Fuente: <https://www.researchgate.net/>

Además, es posible observar la relación entre las variables de holgura, ξ_i (negativa) y ξ_i^* (positiva). Estas actúan como un pequeño margen de tolerancia permitido para generar el hiperplano. Están asociadas a datos de la muestra que quedan fuera de la zona insensible según el factor de penalización C y la función de pérdida L_ϵ .

- Gamma (σ): es el coeficiente para kernels rbf, poly y sigmoid. Puede verse como el inverso del radio de influencia del vector de soporte. Cuando gamma es pequeño, el radio de influencia del vector de soporte es alto. Si el valor gamma es demasiado pequeño, el radio de los vectores de soporte cubrirá todo el conjunto de datos de entrenamiento y no capturará el patrón de los datos. Cuando gamma es grande, el radio de influencia del vector de soporte es bajo. Si el valor gamma es

demasiado grande, el radio del vector de soporte será demasiado pequeño para evitar el sobreajuste.

- Constante (C): representa el grado de penalización para muestras con un error mayor que ε . Una C más grande indica que la penalización por las muestras es grande. La C más pequeña muestra que la penalización por muestras es muy pequeña y causará un gran error de entrenamiento. Cuando C es pequeño, la penalización por clasificación incorrecta es pequeña y la fuerza de la regularización es grande. Por lo tanto, seleccionará un límite de decisión con un gran margen. Cuando C es grande, la penalización por clasificación incorrecta es grande y la fuerza de la regularización es pequeña.

Para la selección de la función kernel del modelo SVR, en este proyecto de título se utilizará la función de base radial RBF, y su parámetro $\sigma > 0$ será el factor de ancho de la función kernel. Tiene una influencia importante en el efecto de predicción de regresión de SVR. Éste tiene la capacidad de aproximar universalmente cualquier distribución en el espacio de características. Con un parámetro σ apropiado, la función kernel RBF proporcionará mejores rendimientos de predicción. Se define como:

$$K(x_i, x_j)_{RBF} = \exp \left(\frac{\|x_i - x_j\|^2}{2 \sigma^2} \right) \quad (21)$$

donde x_i y x_j son espacios vectoriales de entrada y σ^2 representa el ancho de banda de la función kernel (Alrefaee, 2022). Estos tres hiperparámetros (C , σ^2 y ε) con sus respectivos kernels determinan el rendimiento del modelo SVR; por lo tanto, es necesario optimizarlos para mejorar el efecto de predicción de SVR.

Formalmente, un modelo de ML basado en datos se puede describir como una función matemática $\hat{f}$ que intenta proporcionar estimaciones lo más cercanas posible a la verdadera función real de interés (kernel):

$$\begin{aligned} \hat{f}: X &\rightarrow Y \\ \hat{f}(x) &= \hat{f}(\{x_1, \dots, x_f\}) = \text{Predicción} \end{aligned} \quad (22)$$

En este caso, el modelo tiene como objetivo aproximar la función de tiempos de funcionamiento del equipo 1001, en función de un conjunto o serie de datos de entrada univariada, que explican las variables de características (x) de la función del modelo

predictivo $\hat{f}(x)$. Un modelo de ML también puede permitir la entrada de series multivariadas de datos no correlacionados para realizar predicciones con más información y, por lo tanto, más precisas. Para el análisis matemático del modelo SVR, suponga un conjunto de datos dado $\{x_i, y_i\}$ con $i = 1, \dots, f$, donde $x_i \in R^n$ es la muestra de entrada y $y_i \in R$ es el valor esperado de salida, suponga que SVR asigna muestras a un espacio de alta dimensión mediante una transformación no lineal $\phi(x)$ para establecer la función de regresión dada por:

$$\text{Predicción} = f(x) = w * \phi(x) + b \quad (23)$$

donde w representa un vector de peso o predictor, el cual está relacionado con el tamaño del margen y b representa un vector de error, ambos son coeficientes de la función de regresión. Mientras que, $\phi(x)$ representa una transformación no lineal aplicada a la matriz (univariada o multivariada) de datos de entrada. La función de pérdida insensible L_ε , que permite generar el margen suave, se introduce y se define como:

$$L_\varepsilon(f(x), y) = \begin{cases} |y - f(x)| - \varepsilon & \text{si } |y - f(x)| \geq \varepsilon \\ 0, & \text{en otro caso} \end{cases} \quad (24)$$

por lo tanto, la función objetivo se puede definir como:

$$\min \frac{1}{2} \|w\|^2 \quad (25)$$

sujeto a las restricciones:

$$\begin{cases} y_i - w * x_i - b \leq \varepsilon \\ w * x_i + b - y_i \leq \varepsilon \end{cases} \quad (26)$$

$$i = 1, \dots, N$$

Los factores de tolerancia ξ_i y ξ_i^* se introducen bajo la condición de permitir el error de ajuste del hiperplano; entonces la función objetivo quedaría como:

$$\min \left(\frac{1}{2} \|w\|^2 + C \sum_{i=1}^N (\xi_i + \xi_i^*) \right) \quad (27)$$

sujeto a:

$$\begin{cases} y_i - w * x_i - b \leq \varepsilon + \xi_i \\ w * x_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \quad (28)$$

donde $C > 0$ es el factor de penalización, y se utiliza para controlar el castigo por errores superiores a ε . Al introducir el multiplicador de Lagrange α_i^* y α_i , el problema anterior se transforma en su problema dual. Al resolver aquella ecuación, la función de ajuste de regresión se puede resolver de la siguiente manera:

$$f(x) = \sum_{i=1}^N (\alpha_i - \alpha_i^*) K(x_i, x_j) + b \quad (29)$$

Para el problema de predicción del siguiente tiempo de funcionamiento del equipo 1001, perteneciente al sistema Power Feed de una planta de celulosa, se aplica esta función predictiva de SVR mediante lenguaje de programación Python, definiendo la función kernel.

CAPÍTULO 3: EVALUACIÓN DEL MODELO DE PREDICCIÓN PROPUESTO

3 EVALUACIÓN DEL MODELO DE PREDICCIÓN PROPUESTO

En este último capítulo, se definen las métricas de rendimiento utilizadas y se muestran los resultados de los modelos evaluados en las fases de entrenamiento y prueba. El resumen de los resultados de las métricas de rendimiento obtenidas de la predicción se presenta en la Tabla 3-10, de la Sección 3.2.5. A su vez, el marco de trabajo que considera el modelo de predicción propuesto basado en un modelo de regresión SARIMA y un modelo de ML (o basado en datos), como el SVR, se puede expresar como se muestra a continuación (ver Figura 3-1):

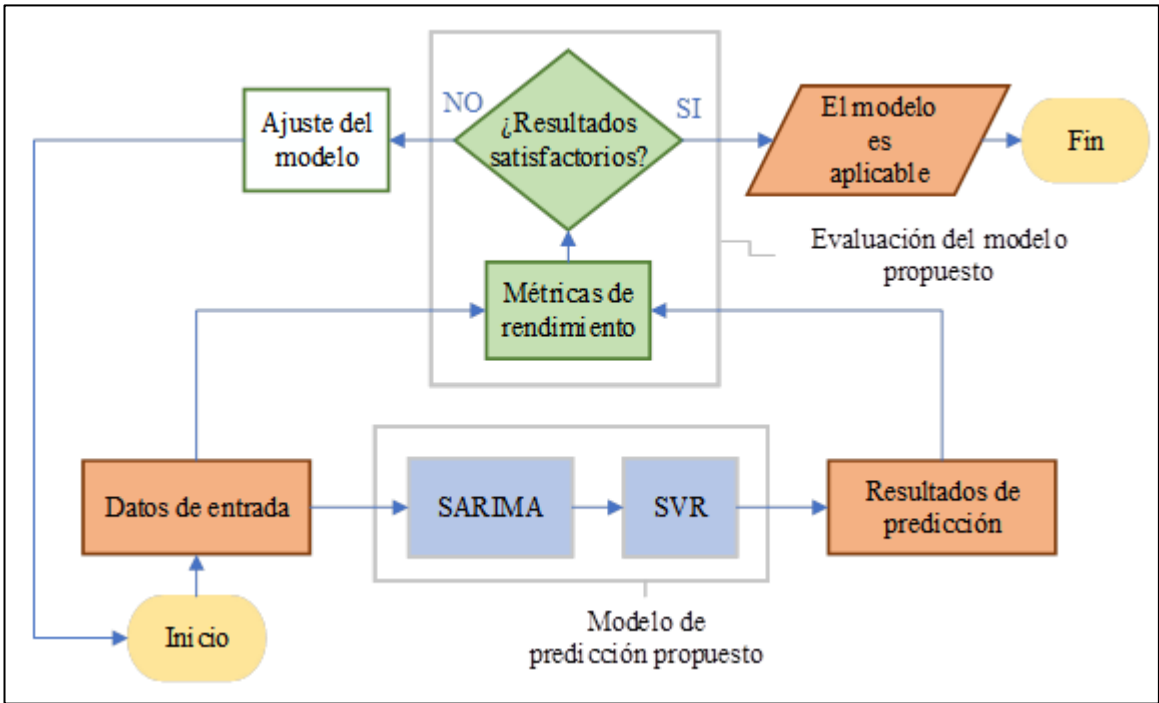


Figura 3-1. Marco de trabajo propuesto

Fuente: Elaboración propia. Basado en Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA modeling (Baptista, 2017)

3.1 MÉTRICAS DE RENDIMIENTO

Cuando se desarrolla un modelo de predicción, se busca que éste que sea lo más preciso posible ajustando iterativamente los parámetros que lo definen. Sin embargo, no es posible construir un modelo 100% preciso ya que no es posible estar libre de errores, por lo tanto, comprender las diferentes fuentes de error, ayudará a obtener modelos más precisos. Para determinar que un algoritmo funcione correctamente, se hace necesario medir el rendimiento o desempeño del modelo mediante métricas de rendimiento de regresión. En estadística, se utiliza el valor de la diferencia entre dos variables continuas (series de datos calculados y observados), para así, cuantificar la exactitud y precisión de una técnica de predicción comparando los valores predichos frente a los observados. Las métricas de exactitud y precisión utilizadas en este proyecto de título incluyen, por ejemplo, la raíz del error cuadrático medio (RMSE) y la desviación absoluta mediana (MAD). También se incluye el tiempo medio entre fallas (MTBF), una métrica del dominio de confiabilidad. Esta métrica se recomienda en (Saxena A., 2008), como una métrica de costo/beneficio destinado a medir de manera abstracta el beneficio proporcionado por los pronósticos. El MTBF mide la relación entre la duración de un componente y el tiempo que se usa antes de reemplazarlo. El uso de pronósticos debería reducir esta relación al permitir que los componentes se usen por más tiempo hasta que estén más cerca de fallar, lo que ahorraría dinero. La ganancia, en este caso, es la cantidad de dinero ahorrada como resultado del uso de pronósticos, es decir, la reducción en el costo del ciclo de vida, mientras que la inversión es el costo de desarrollar, instalar y mantener el sistema de pronósticos. Se espera que una inversión en pronósticos ahorre dinero en mantenimiento y posiblemente prevenga el tiempo de inactividad durante la vida útil del sistema. Agregar capacidad de pronóstico a un sistema existente provoca un aumento en el costo de adquisición, debido al costo de sensores adicionales, hardware informático adicional y el costo de desarrollar el sistema de pronóstico. Las métricas de costo/beneficio, están destinadas a medir el beneficio proporcionado por los pronósticos. Todos están influenciados por la precisión con la que se predice la vida útil restante o RUL. Por ejemplo, en la relación MTBF/MTBR, el tiempo medio entre reemplazos de unidades puede aumentar si RUL se predice con mayor precisión. Por lo tanto, la implementación de metodologías de predicción es rentable a medida que se obtienen métricas precisas. En el Anexo B, se propone un método de cálculo para analizar el costo/beneficio de la implementación de modelos predictivos. Mientras que, en la Tabla 3-1 mostrada a continuación, se presentan las métricas utilizadas para la evaluación del rendimiento de los modelos de predicción en estudio.

Tabla 3-1. Métricas para la evaluación del rendimiento de los modelos de predicción en estudio, correspondiente a diferentes tipos de errores entre los pronósticos $\hat{y}_E(t)$, y el valor real $y_E(t)$ observado, en un tiempo t determinado.

Error medio	$ME = \frac{1}{n} \sum_{t=1}^n (\hat{y}_E(t) - y_E(t)) \tag{30}$
Error mediano	$MdE = mediana \left(\{\hat{y}_E(t) - y_E(t)\}_{t=1}^n \right) \tag{31}$
Raíz del Error cuadrático medio	$RMSE = \sqrt{\frac{1}{n} \sum_{t=1}^n (\hat{y}_E(t) - y_E(t))^2} \tag{32}$
Error de porcentaje absoluto medio	$MAPE = \frac{1}{n} \sum_{t=1}^n \left \frac{\hat{y}_E(t) - y_E(t)}{y_E(t)} \right \tag{33}$
Error absoluto medio	$MAE = \frac{1}{n} \sum_{t=1}^n \hat{y}_E(t) - y_E(t) \tag{34}$
Error absoluto mediano	$MdAE = mediana \left(\{ \hat{y}_E(t) - y_E(t) \}_{t=1}^n \right) \tag{35}$
Desviación absoluta media	$MAD = mediana \left(\{ y_E(t) - mediana(\{y_E(t)\}_{t=1}^n) \}_{t=1}^n \right) \tag{36}$
Desviación estándar de muestra	$SSD = \sqrt{\frac{\sum_{t=1}^n \left((\hat{y}_E(t) - y_E(t)) - ME \right)^2}{n - 1}} \tag{37}$
Tiempo medio entre fallos	$MTBF = \frac{1}{n} \sum_{t=1}^n \hat{y}_E(t) = \overline{\hat{y}_E}(t) \tag{38}$

Fuente: Elaboración propia. Basado en Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA modeling (Baptista, 2017)

El Error medio (ME) representa un promedio de los errores. Puede ser positivo o negativo. El Error mediano (MdE) corresponde a la mediana de los errores y también puede ser positivo o negativo. La Raíz del Error cuadrático medio (RMSE), junto con el coeficiente de determinación, se considera como una de las métricas más importantes por su aplicación para medir el error en modelos de ML de regresión, y corresponde a la raíz cuadrada de la media de los errores elevados al cuadrado. Entre mayor RMSE, mayor es el error y menos preciso el modelo. El Error de porcentaje absoluto medio (MAPE), es otra medida de error que permite establecer el desempeño de un pronóstico. Mide el

tamaño del error absoluto en términos porcentuales. Es una medida ampliamente utilizada para establecer la magnitud del error de predicción. El Error absoluto medio (MAE), proporciona el promedio de la diferencia absoluta entre la predicción del modelo y el valor real; mientras que el Error absoluto mediano (MdAE), proporciona la mediana de la diferencia absoluta entre la predicción del modelo y el valor real. La Desviación absoluta media (MAD), se utiliza como una medida de dispersión estadística, la cual es menos sensible a los valores atípicos de una serie de datos en comparación con la desviación estándar. La Desviación estándar de muestra (SSD), mide el grado de dispersión o variabilidad de una muestra. Mientras mayor es la desviación estándar, mayor es la dispersión de la muestra. El Tiempo medio entre fallos (MTBF), corresponde a la media del registro de los tiempos de funcionamiento del equipo en estudio. Por otro lado, y de manera general, los errores de predicción para cualquier algoritmo de ML se pueden dividir en 3 tipos:

- i. Error irreducible: Es aquel que no se puede reducir, independientemente de qué algoritmo se use. También se le conoce como ruido, y por lo general proviene de factores como variables desconocidas que influyen en la distribución desde las variables de entrada a la variable de salida.
- ii. Error de bias o sesgo: Es la diferencia entre la predicción esperada de nuestro modelo y los valores verdaderos. Las métricas relacionadas son el error medio (ME) y el error mediano (MdE). Aunque el objetivo es siempre construir modelos que puedan predecir datos muy cercanos a los valores reales, algunos algoritmos son demasiado rígidos para aprender señales complejas del conjunto de datos, como lo son los modelos de regresión lineal, por lo tanto, éstos mostrarán un alto error de bias o sesgo.
- iii. Error de varianza: Se refiere a la cantidad en que la estimación de la función objetivo cambiará, si se utilizan diferentes datos de entrenamiento. Está relacionado con las métricas de exactitud y precisión. La función objetivo se estima a partir de los datos de entrenamiento mediante un algoritmo de Machine Learning, por lo que deberíamos esperar que el algoritmo tenga alguna variación. Idealmente no debería cambiar demasiado de un conjunto de datos de entrenamiento a otro, lo que significa que el algoritmo es bueno para elegir el mapeo subyacente oculto entre las variables de entrada y de salida. Los algoritmos de Machine Learning que tiene una gran varianza están fuertemente influenciados por los detalles de los datos de entrenamiento.

3.2 **RESULTADOS**

3.2.1 Exploración de la serie de datos

En esta última sección, se muestra el desarrollo y evaluación de la metodología presentada en el Capítulo 2, para la mejora en la predicción de eventos, mediante el uso de los modelos propuestos presentados, en la Sección 2.2. Para que la serie sea estacionaria, no deben existir cambios sistemáticos ni en la media (sin tendencia), ni en sus variaciones (sin varianza), ni en sus fluctuaciones periódicas (covarianza). Además, se requiere que la condición de la distribución del ruido sea normal. Mediante un gráfico de descomposición de tendencia, estacionalidad y ruido, se realizó un primer acercamiento hacia los patrones implícitos que son propios del conjunto de datos de entrada (ver Figura 3-2).

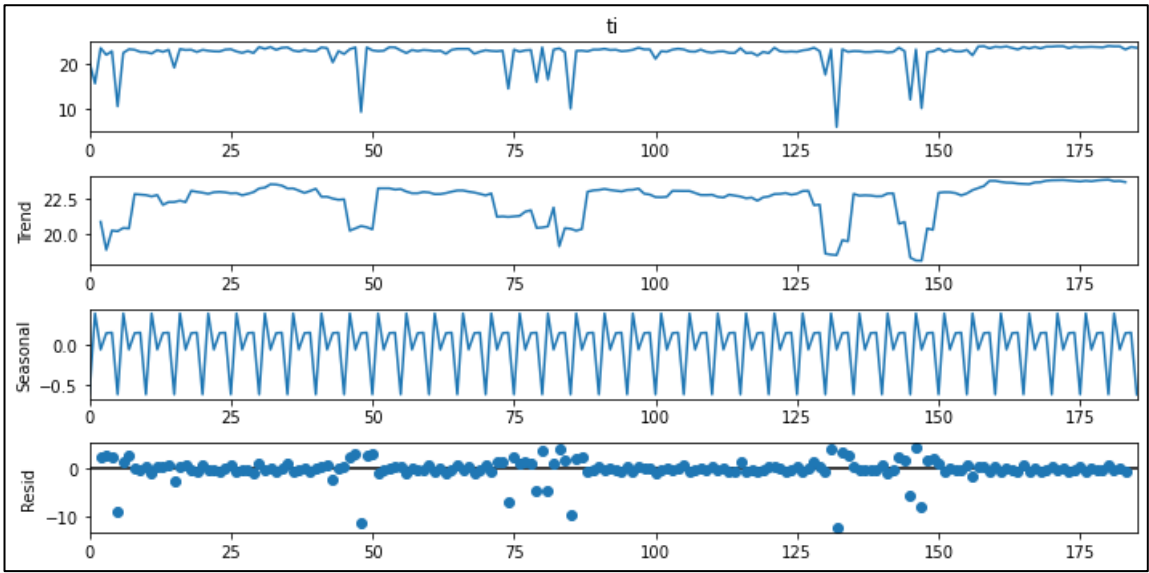


Figura 3-2. Arriba, se muestran los tiempos de funcionamiento del equipo 1001. En segundo, tercer y cuarto lugar respectivamente, se muestran los gráficos de descomposición de tendencia, estacionalidad y ruido residual del registro de los tiempos de funcionamiento del equipo 1001.

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

En este caso, es posible observar una tendencia variable con muchas fluctuaciones y eventuales peaks muy bajos en ciertos periodos de tiempo. Podemos observar, además, que existe un patrón de estacionalidad que se repite cada dos períodos con una variación de tiempos entre -0,5 y 0,5 horas. Mientras que el ruido residual muestra una distribución normal muy cercana a su media, en su gran parte.

Del histograma (ver Figura 3-3), se puede inferir que la mayor cantidad de fallas del equipo 1001 ocurren entre las 20 y las 24 horas de funcionamiento, con una media aproximada de 23 horas y una densidad de distribución de fallas con una asimetría altamente negativa.

Aplicar una diferenciación, permitiría centrar los datos para intentar tener una distribución normal.



Figura 3-3. Histograma y densidad de la probabilidad de fallas del equipo 1001

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Un gráfico de normalidad QQ (cuartil-cuartil), es un gráfico que muestra la distribución de un dato dado frente a la distribución normal, es decir, los cuantiles existentes frente a los cuantiles teóricos normales. Del gráfico de normalidad QQ de la serie de tiempos de funcionamiento del equipo 1001 (ver Figura 3-4), se observa que la distribución de los datos no sigue el comportamiento normal deseado (línea roja). Sin embargo, tiene un patrón que puede ser favorable para el modelo de predicción propuesto. Aplicar una transformación a los datos podría mejorar esta condición.

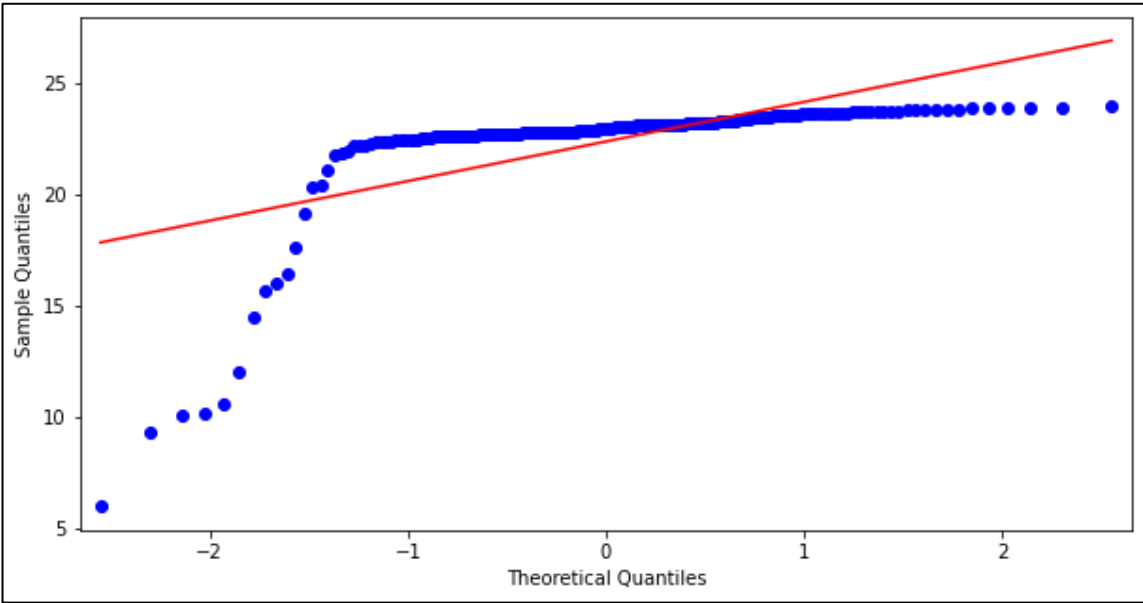


Figura 3-4. Gráfico de normalidad QQ de la serie de tiempos de funcionamiento del equipo 1001

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001

Por otro lado, es posible observar mediante un gráfico de correlación simple o autocorrelación (Figura 3-5), que existe cierto nivel de dependencia entre las variables el cual es necesario analizar, ya que esto no es favorable para el modelo de predicción. Esta condición también podría mejorar mediante alguna transformación de los datos de entrada o, quizás, mediante algún algoritmo de aprendizaje no supervisado de clasificación, como lo es el Análisis de Componentes Principales o Principal Component Analysis (PCA) el cual no se aplica en este estudio. Un análisis más acabado sobre la correlación se puede realizar mediante la autocorrelación parcial (ver Figura 3-6). De éstos, se puede comprobar que realmente existe dependencia entre las variables, pero su nivel es muy bajo. Es posible atacar esta condición incluyendo en el modelo de regresión, un parámetro autorregresivo estacional $P = 2$.

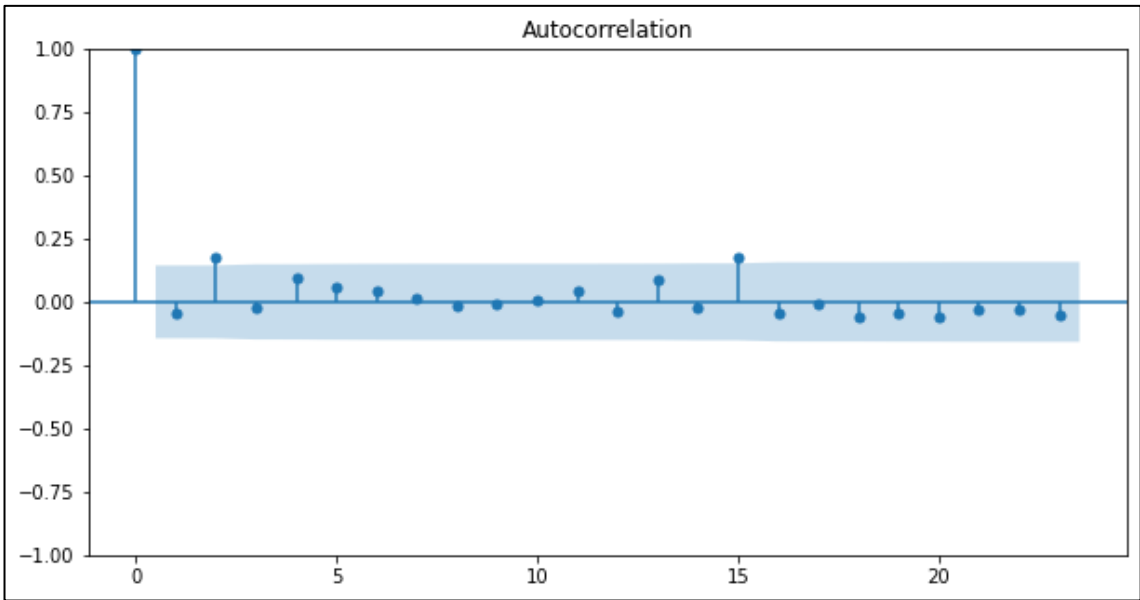


Figura 3-5. Gráfico de la función de autocorrelación. Permite evaluar el nivel de dependencia entre las variables de análisis

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001

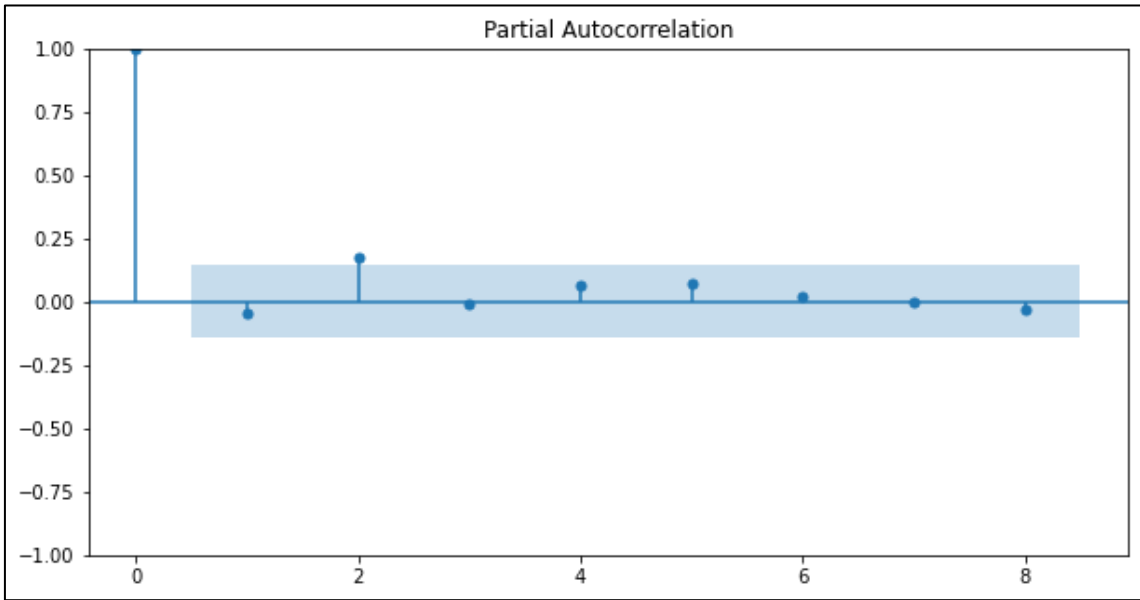


Figura 3-6. Gráfico de la función de autocorrelación parcial. Permite evaluar el nivel de dependencia entre un rango menor de variables de análisis

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001

Se entregan a continuación, los resultados de estacionariedad obtenidos en el test de Dickey Füller (Figura 3-7 y Tabla 3-2). Esta prueba nos indica directamente, según su criterio, que la serie es estacionaria, ya que el valor p-value obtenido es bastante menor que 0,05 y muy cercano a cero ($p\text{ value} = 2,73 \times 10^{-13}$). Por lo tanto, es posible aplicar

directamente nuestra serie de datos en el modelo considerando sus patrones de tendencia y estacionalidad. Sin embargo, se debe considerar también que existe una alta probabilidad de mejorar el nivel de predicción de acuerdo con los análisis anteriores, aplicando alguna transformación.

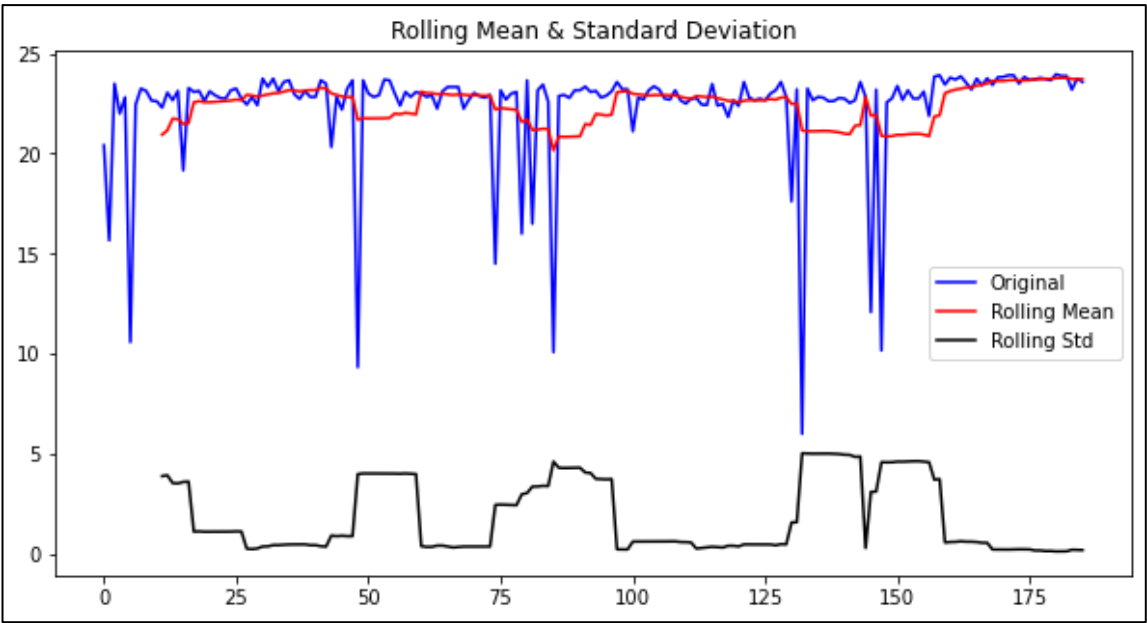


Figura 3-7. Gráfico de media móvil y desviación estándar para el análisis de estacionariedad de la prueba de Dickey Füller

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001

Tabla 3-2. Resultados de la prueba de estacionariedad de Dickey Füller

Test Statistic	-8,367
p-value	2,73E-13
#Lags Used	1
Number of Observations	184
Critical value (1%)	-3,466
Critical value (5%)	-2,877
Critical value (10%)	-2,575

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Para finalizar la etapa de exploración, se realiza una última prueba, para evaluar la estacionariedad de manera parcial. Para esto, utilizamos la prueba de Ljung-Box (Figura 3-8), evaluando iterativamente distintas cantidades de rezagos, logrando determinar que con dos rezagos la serie de tiempo se hace estacionaria, obteniendo un *p value* < 0,05. Esto coincide con la información de los gráficos de autocorrelación obtenidos. Por lo tanto, podemos afirmar y definir de antemano que el mejor valor autorregresivo estacional "P" del modelo de regresión SARIMA, es 2.

lb_stat	lb_pvalue
2	6.410651 0.040546

Figura 3-8. Resultados de la prueba de estacionariedad mediante rezagos de Dickey Füller, obtenidos mediante lenguaje de programación Python. La prueba nos indica que cada 2 rezagos, la serie se vuelve estacionaria.

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Todas las pruebas de estacionariedad aplicadas utilizan el valor “p-value” como métrica de decisión, sin embargo, no todas utilizan los mismos criterios de evaluación. De esta manera, el test de Dickey Füller (Figura 3-7), la prueba de Ljung-Box (Figura 3-8), y el test Chi-cuadrado, mostrado a continuación en la Figura 3-9, son los tests capaces de detectar e indicar que la muestra tiene una distribución probablemente gaussiana.

```
#Chi-Cuadrado Test:
statistic, pvalue = chisquare(df.ti)
print('statistics=%.3f, p=%.3f\n' % (statistic, pvalue))
if pvalue > 0.05:
    print('Probably Gaussian')
else:
    print('Probably not Gaussian')

statistics=59.326, p=1.000
Probably Gaussian
```

Figura 3-9. Resultados de la prueba de estacionariedad de Chi-Cuadrado, mediante lenguaje de programación Python

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

3.2.2 Datos de entrenamiento y prueba

A continuación, se muestra en la Figura 3-10, los datos que serán utilizados en la etapa de entrenamiento (en azul), y los que serán utilizados en la posterior etapa de predicción (en amarillo):

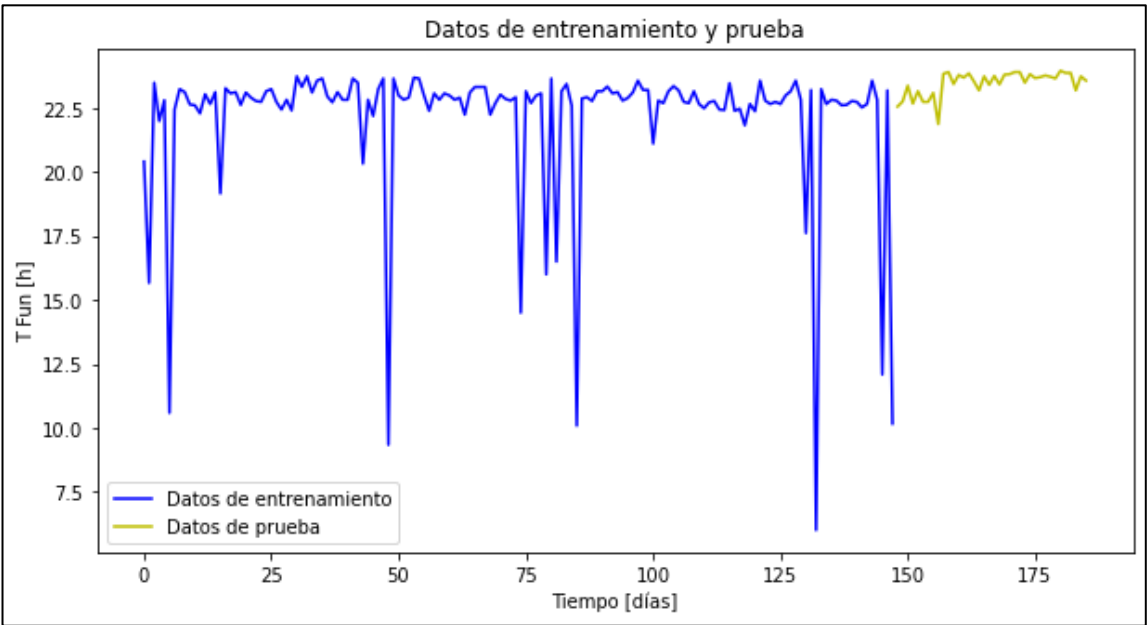


Figura 3-10. En azul se muestra el 80% de los valores de la serie de datos de tiempos de funcionamiento (en horas), del equipo 1001, que serán utilizados para la etapa de entrenamiento, en donde se busca definir los mejores valores para los parámetros del modelo de predicción. En amarillo se muestra el 20% restante de los datos, que serán utilizados para realizar predicciones con los parámetros determinados en la etapa de entrenamiento.

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001

Por ende, como se muestra en la Figura 3-10, para la etapa de entrenamiento se utilizarán los primeros 148 valores de la serie de datos, mientras que para la etapa de predicción se utilizará el 20% restante, correspondiente a 38 valores. El desafío consiste en definir los valores de los parámetros de un modelo que mejor se ajuste a los datos de entrenamiento, para luego realizar una predicción sobre datos nuevos con la mayor precisión y exactitud posible.

3.2.3 Resultados de entrenamiento

Para evitar errores de sesgo (o bias), y lograr un mejor aprendizaje del modelo de predicción, es vital otorgar igual importancia a las variables de estudio mediante una ponderación homogénea de los datos. Por ende, previo al entrenamiento, y para obtener mejores resultados de predicción, es necesario realizar un escalado a las variables en estudio (Tiempos de funcionamiento “T fun” en horas y “Tiempo” en días). Esto corresponde a igualar o estandarizar la información en una misma escala. Una variable estandarizada o normalizada, se utiliza en estadística para comparar datos procedentes de diferentes muestras o poblaciones. Como resultado de este proceso, gráficamente la data se mantiene con la misma distribución anterior, sin embargo, la expresión de sus escalas es distinta, ya que, luego de aplicar una estandarización, éstas son similares, como se muestra en la Figura 3-11 a continuación.

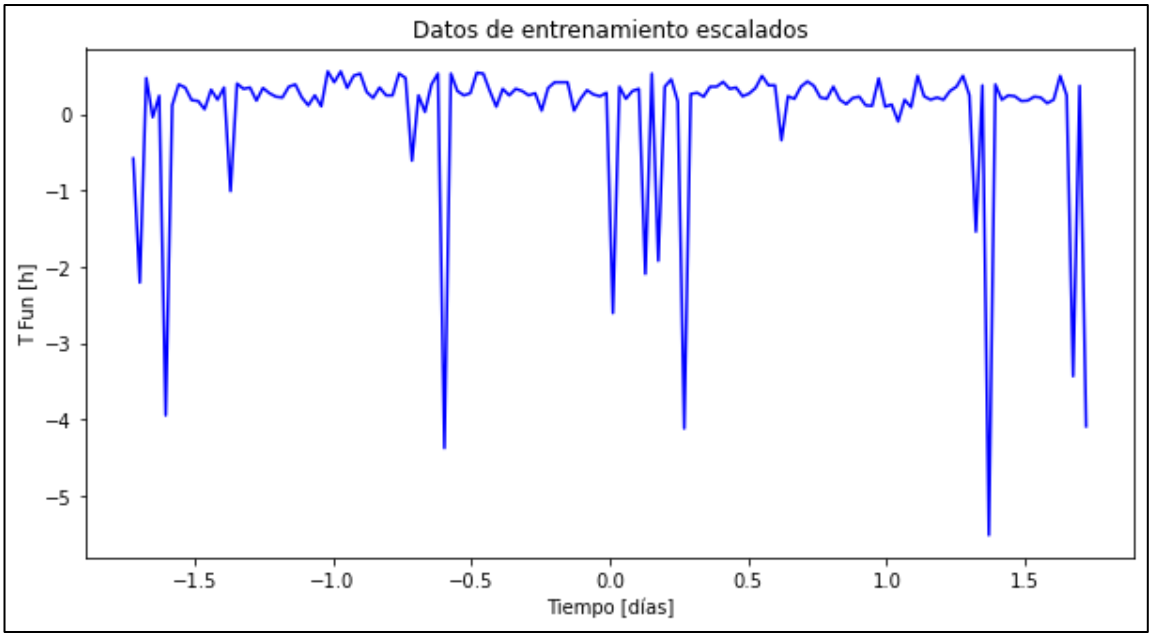


Figura 3-11. Aplicación del método de escalado a la serie de datos de entrenamiento.

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Con los datos separados y escalados, y mediante lenguaje de programación Python, se procede a optimizar los parámetros del modelo SARIMA: (p, d, q) (P, D, Q, s), con el conjunto de entrenamiento de la Figura 3-11. Acá se entrenan varios modelos con distintos parámetros usando el conjunto de entrenamiento, siguiendo el marco de trabajo propuesto en la Figura 2-1. Los criterios de información de Akaike o AIC son una buena medida para probar la bondad de cómo se ajusta matemáticamente el modelo. Las puntuaciones de los criterios de información AIC mostrados en la Figura 3-12, miden la cantidad de información perdida al entrenar y generalizar el modelo SARIMA. El AIC debe ser lo más bajo posible. El resumen de los resultados de la optimización del modelo SARIMA entrenado se muestra en la Figura 3-13, en donde podemos verificar la mejor puntuación AIC y otras medidas de rendimiento estadístico. En la Tabla 3-4, se muestran los valores de los parámetros del modelo de regresión SARIMA, obtenidos desde el conjunto de datos de entrenamiento.

```
Performing stepwise search to minimize aic
ARIMA(0,True,0)(0,0,1)[5] intercept : AIC=530.319, Time=2.48 sec
ARIMA(0,True,0)(0,0,0)[5] intercept : AIC=528.348, Time=0.03 sec
ARIMA(1,True,0)(1,0,0)[5] intercept : AIC=464.299, Time=0.28 sec
ARIMA(0,True,1)(0,0,1)[5] intercept : AIC=inf, Time=0.33 sec
ARIMA(0,True,0)(0,0,0)[5] intercept : AIC=526.388, Time=0.03 sec
ARIMA(1,True,0)(0,0,0)[5] intercept : AIC=462.751, Time=0.05 sec
ARIMA(1,True,0)(0,0,1)[5] intercept : AIC=464.327, Time=0.07 sec
ARIMA(1,True,0)(1,0,1)[5] intercept : AIC=465.645, Time=0.17 sec
ARIMA(2,True,0)(0,0,0)[5] intercept : AIC=454.001, Time=0.13 sec
ARIMA(2,True,0)(1,0,0)[5] intercept : AIC=455.537, Time=0.06 sec
ARIMA(2,True,0)(0,0,1)[5] intercept : AIC=455.532, Time=0.06 sec
ARIMA(2,True,0)(1,0,1)[5] intercept : AIC=456.619, Time=0.23 sec
ARIMA(3,True,0)(0,0,0)[5] intercept : AIC=444.507, Time=0.09 sec
ARIMA(3,True,0)(1,0,0)[5] intercept : AIC=445.232, Time=0.19 sec
ARIMA(3,True,0)(0,0,1)[5] intercept : AIC=445.180, Time=0.10 sec
ARIMA(3,True,0)(1,0,1)[5] intercept : AIC=445.789, Time=0.31 sec
ARIMA(4,True,0)(0,0,0)[5] intercept : AIC=439.629, Time=0.06 sec
ARIMA(4,True,0)(1,0,0)[5] intercept : AIC=438.035, Time=0.45 sec
ARIMA(4,True,0)(2,0,0)[5] intercept : AIC=430.906, Time=0.25 sec
ARIMA(4,True,0)(2,0,1)[5] intercept : AIC=432.757, Time=0.49 sec
ARIMA(4,True,0)(1,0,1)[5] intercept : AIC=436.351, Time=0.30 sec
ARIMA(3,True,0)(2,0,0)[5] intercept : AIC=447.227, Time=0.15 sec
ARIMA(4,True,1)(2,0,0)[5] intercept : AIC=431.592, Time=0.44 sec
ARIMA(3,True,1)(2,0,0)[5] intercept : AIC=inf, Time=0.59 sec
ARIMA(4,True,0)(2,0,0)[5] intercept : AIC=428.922, Time=0.15 sec
ARIMA(4,True,0)(1,0,0)[5] intercept : AIC=436.073, Time=0.08 sec
ARIMA(4,True,0)(2,0,1)[5] intercept : AIC=430.770, Time=0.24 sec
ARIMA(4,True,0)(1,0,1)[5] intercept : AIC=434.395, Time=0.15 sec
ARIMA(3,True,0)(2,0,0)[5] intercept : AIC=445.303, Time=0.08 sec
ARIMA(4,True,1)(2,0,0)[5] intercept : AIC=429.606, Time=0.26 sec
ARIMA(3,True,1)(2,0,0)[5] intercept : AIC=inf, Time=0.38 sec

Best model: ARIMA(4,True,0)(2,0,0)[5]
Total fit time: 8.734 seconds
```

Figura 3-12. Resultados de entrenamiento modelo de regresión mediante lenguaje de programación Python

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

SARIMAX Results						
=====						
Dep. Variable:	y	No. Observations:	148			
Model:	SARIMAX(4, 1, 0)x(2, 0, 0, 5)	Log Likelihood	-207.461			
Date:	Mon, 07 Nov 2022	AIC	428.922			
Time:	19:34:54	BIC	449.855			
Sample:	0	HQIC	437.427			
	- 148					
Covariance Type:	opg					
=====						
	coef	std err	z	P> z	[0.025	0.975]

ar.L1	-0.9763	0.082	-11.964	0.000	-1.136	-0.816
ar.L2	-0.8225	0.094	-8.755	0.000	-1.007	-0.638
ar.L3	-0.7838	0.102	-7.663	0.000	-0.984	-0.583
ar.L4	-0.6958	0.108	-6.461	0.000	-0.907	-0.485
ar.S.L5	-0.5784	0.143	-4.048	0.000	-0.858	-0.298
ar.S.L10	-0.3791	0.085	-4.438	0.000	-0.546	-0.212
sigma2	0.9675	0.050	19.480	0.000	0.870	1.065
=====						
Ljung-Box (L1) (Q):	0.71	Jarque-Bera (JB):	1088.63			
Prob(Q):	0.40	Prob(JB):	0.00			
Heteroskedasticity (H):	1.11	Skew:	-3.13			
Prob(H) (two-sided):	0.71	Kurtosis:	14.77			
=====						
Warnings:						
[1] Covariance matrix calculated using the outer product of gradients (complex-step).						

Figura 3-13. Resumen de los resultados de entrenamiento del modelo de regresión mediante lenguaje de programación Python. Acá, el algoritmo optimizador da cuenta de que el modelo de regresión ARIMA que mejor se ajusta a los datos de entrenamiento corresponde a la variante SARIMA, la cual incluye parámetros estacionales. Nota: El nombre “SARIMAX” hace referencia a la variante del modelo ARIMA que utiliza variables estacionales y exógenas, las cuales no se aplican en este proyecto de título.

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Tabla 3-3. Valores de los parámetros del modelo de regresión SARIMA, obtenidos desde el conjunto de datos de entrenamiento

Parámetros modelo de regresión SARIMA	
p	4
d	1
q	0
P	2
D	0
Q	0
s	5

Fuente: Elaboración propia. Base de datos del equipo 1001.

De acuerdo con los resultados, el mejor modelo de regresión que se ajusta al conjunto de entrenamiento es el SARIMA (4, 1, 0) (2, 0, 0, 5), con un $AIC = 428,922$. De esta manera, podemos representar el modelo SARIMA (4, 1, 0) (2, 0, 0, 5), como una serie de tiempo X_t estacional diferenciada. Matemáticamente, se puede expresar como:

$$(1 - B^5) (1 - B) X_t = Z_t + \sum_{i=1}^{p=4} \phi_i Y_{t-i} + \sum_{i=1}^{P=2} \Phi_i Y_{t-i} \tag{40}$$

Su aplicación al conjunto de datos se representa como se muestra en la Figura 3-14, junto con sus resultados de rendimiento en la Tabla 3-4. Esta es una imagen referencial sobre cómo se ajustan los parámetros del modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), a la serie de tiempos de funcionamiento (T fun), del equipo 1001. Sin embargo, la metodología de predicción propuesta en este trabajo define que éstas predicciones son los datos de entrada para ajustar los hiperparámetros del modelo SVR.

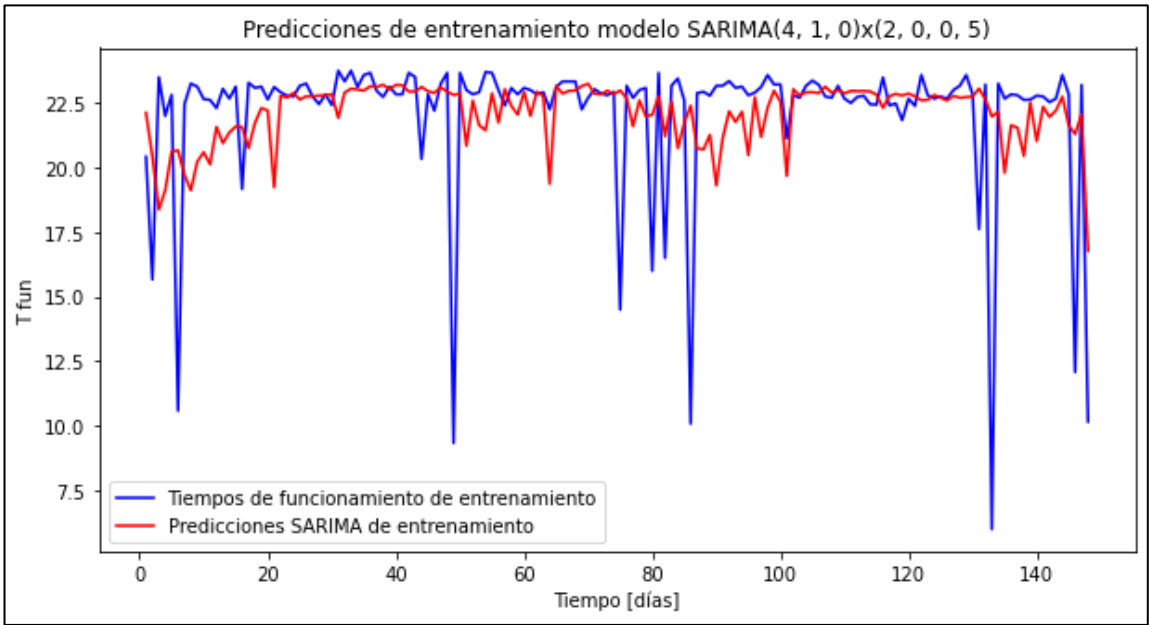


Figura 3-14. Predicciones de entrenamiento del modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), sobre el conjunto de datos de entrenamiento, mediante lenguaje de programación Python

Fuente: Elaboración propia mediante lenguaje de programación Python y la base de datos del equipo 1001.

Tabla 3-4. Resultados de rendimiento del modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), sobre el conjunto de datos de entrenamiento

ME	MdE	RMSE	MAPE	MAE	MdAE	MAD	SSD	MTBF
0,004	0,372	2,918	11,081	1,554	0,695	0,3	2,922	22,12

Fuente: Elaboración propia.

Siguiendo el marco de trabajo propuesto, se deben ajustar los mejores hiperparámetros para el modelo SVR mediante las predicciones del modelo de regresión. En la Figura 3-15, se muestra el algoritmo optimizador utilizado para encontrar los hiperparámetros que mejor se ajustan a un modelo SVR de *kernel = rbf*. Se buscan los valores de “C”, “epsilon” y “gamma” de la lista “param_grid”. Como salida, se muestran los mejores hiperparámetros encontrados. Todo esto, utilizando el método de validación cruzada y “GridSearch”, mediante lenguaje de programación Python. En la Tabla 3-5, se muestran los valores de hiperparámetros, obtenidos desde las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5), del conjunto de datos de entrenamiento.

```
gsc = GridSearchCV(
    estimator=SVR(kernel='rbf'),
    param_grid={
        'C': [0.1, 1, 10, 100, 1000, 10000, 100000, 1000000],
        'epsilon': [0.0015, 0.001, 0.01, 0.015, 0.05, 0.001, 0.005],
        'gamma': [0.001, 0.03, 0.1, 0.3, 0.5, 0.9, 1, 1.3]
    },
    cv=5, scoring='r2', verbose=False, n_jobs=-1)

grid_result = gsc.fit(X_train, y_arima1.predict())
best_params = grid_result.best_params_
best_svr = SVR(kernel="rbf", C=best_params["C"], epsilon=best_params["epsilon"],
               gamma=best_params["gamma"], tol=0.001, verbose=False, max_iter=-1)

print('Mejor puntaje: ', gsc.best_score_)
print('Mejores parametros: ', gsc.best_params_)

Mejor puntaje: 0.00388
Mejores parametros: {'C': 10000, 'epsilon': 0.05, 'gamma': 0.1}
```

Figura 3-15. Resultados de entrenamiento del modelo basado en datos SVR, mediante lenguaje de programación Python

Fuente: Elaboración propia mediante lenguaje de programación Python.

Tabla 3-5. Valores de los hiperparámetros del modelo de ML: SVR, obtenidos desde las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5) del conjunto de datos de entrenamiento

Hiperparámetros del modelo de ML: Support Vector Regression	
kernel	rbf
C	10000
epsilon	0,05
gamma	0,1

Fuente: Elaboración propia. Base de datos del equipo 1001.

En la Figura 3-16, se muestran en rojo las predicciones del modelo de regresión, junto con las del modelo de ML: $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, en verde. Posteriormente, se entregan los resultados de rendimiento en la Tabla 3-6:

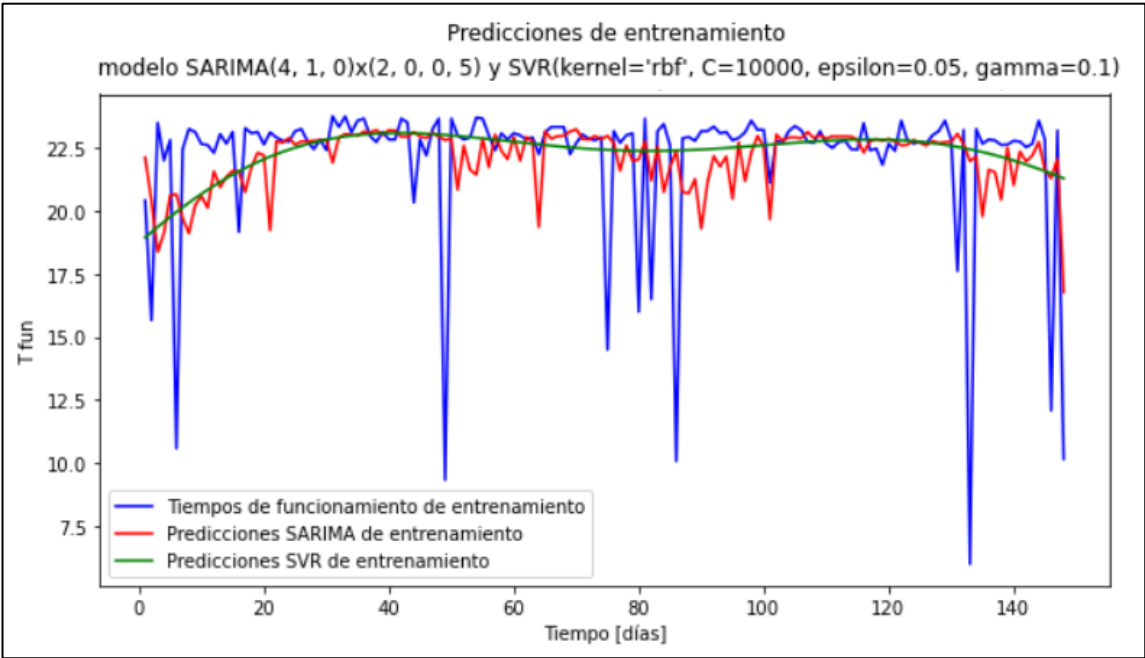


Figura 3-16. En rojo se muestran las predicciones SARIMA aplicadas al conjunto de datos de entrenamiento de los tiempos de funcionamiento del equipo 1001 (en azul). En verde se muestran las predicciones obtenidas desde el modelo basado en datos $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5).

Fuente: Elaboración propia mediante lenguaje de programación Python.

Tabla 3-6. Resultados de rendimiento del modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre los datos del conjunto de entrenamiento

ME	MdE	RMSE	MAPE	MAE	MdAE	MAD	SSD	MTBF
-0,211	0,331	2,873	10,241	1,333	0,583	0,3	2,922	22,33

Fuente: Elaboración propia.

Como se menciona en el marco de trabajo, si los resultados de rendimiento del conjunto de entrenamiento son satisfactorios, se puede decir que el modelo es aplicable para realizar predicciones sobre nuevos datos. Por lo tanto, una vez optimizados los parámetros y, además, haber obtenido errores mínimos en las métricas de rendimiento, se debe realizar el mismo procedimiento (con los parámetros SARIMA e hiperparámetros del modelo SVR, ya definidos), sobre el conjunto de datos de prueba correspondiente al 20% restante de los datos, es decir, sobre las 38 nuevas muestras (indicadas en amarillo de la Figura 3-11).

3.2.4 Resultados de prueba

Con el modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), se procede a obtener las predicciones sobre el conjunto de datos de prueba, donde se obtuvieron los siguientes valores (mostrados en rojo), en el gráfico de la Figura 3-17, junto con sus respectivos resultados de rendimiento de las métricas de evaluación en el error de predicción, en la Tabla 3-7:

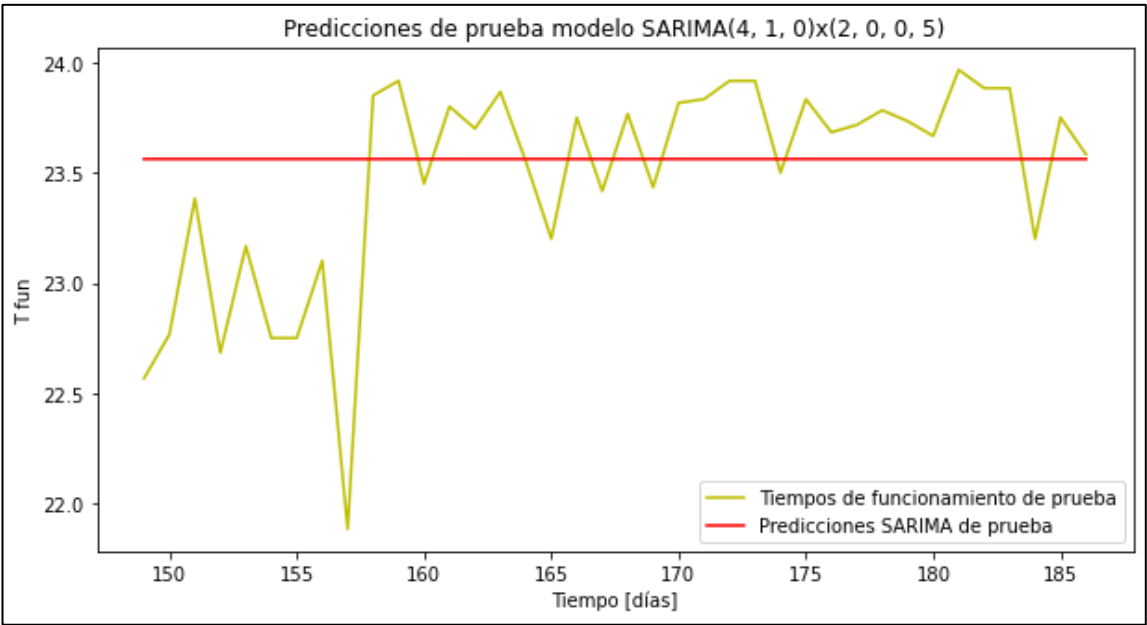


Figura 3-17. En rojo se muestran las predicciones SARIMA aplicadas al conjunto de datos de prueba de los tiempos de funcionamiento del equipo 1001 (en amarillo)

Fuente: Elaboración propia mediante lenguaje de programación Python.

Tabla 3-7. Resultados de rendimiento del modelo SARIMA (4, 1, 0) (2, 0, 0, 5), sobre los datos del conjunto de prueba

ME	MdE	RMSE	MAPE	MAE	MdAE	MAD	SSD	MTBF
-0,077	0,13	0,479	1,531	0,354	0,271	0,192	0,472	23,56

Fuente: Elaboración propia.

En la Figura 3-17, es posible apreciar que el modelo de regresión aplicado en el conjunto de datos de prueba define una línea recta. Sin embargo, este ajuste entrega los mejores resultados en cuanto a errores de predicción frente a los valores observados. Además, como se pudo observar durante la etapa de entrenamiento, el modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, mostrado en verde en la Figura 3-16, realiza un recorrido o “mapeo” sobre las predicciones del modelo SARIMA (4, 1, 0) (2, 0, 0, 5), por lo tanto, es de esperar que ocurra un efecto similar en la etapa de prueba. De esta forma, como se muestra en el gráfico de la Figura 3-18, es posible apreciar que las predicciones obtenidas por el modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, coinciden con las predicciones del modelo SARIMA (4, 1, 0) (2, 0, 0, 5), sobre el conjunto de datos de prueba. Los resultados de rendimiento finales del modelo de predicción propuesto se muestran en la Tabla 3-8.

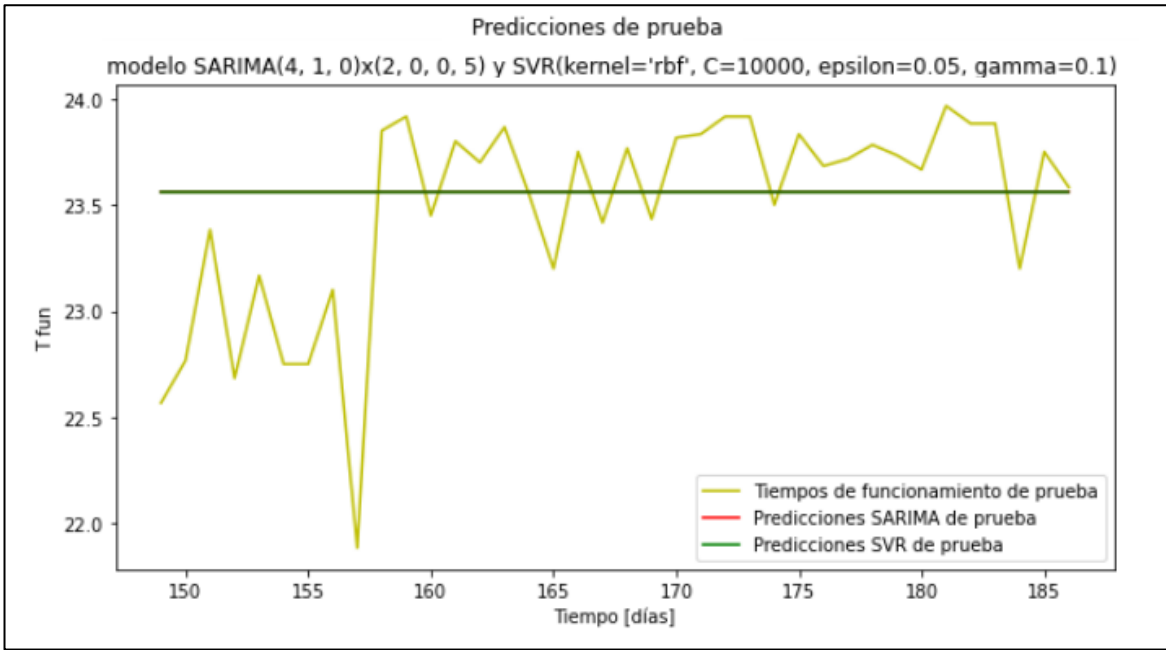


Figura 3-18. En este gráfico, se muestran las predicciones finales de la etapa de prueba, mostradas en verde, obtenidas desde el modelo basado en datos $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre las predicciones SARIMA (4, 1, 0) (2, 0, 0, 5), las cuales se deberían mostrar en rojo.

Fuente: Elaboración propia mediante lenguaje de programación Python.

Debido a la coincidencia entre las predicciones del modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, con las predicciones del modelo SARIMA (4, 1, 0) (2, 0, 0, 5), en esta etapa de prueba, los valores de las métricas de evaluación mostrados en la Tabla 3-8 coinciden con los resultados mostrados en la Tabla 3-7.

Tabla 3-8. Resultados de rendimiento del modelo $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, sobre los datos del conjunto de prueba

ME	MdE	RMSE	MAPE	MAE	MdAE	MAD	SSD	MTBF
-0,077	0,13	0,479	1,531	0,354	0,271	0,192	0,472	23,56

Fuente: Elaboración propia.

3.2.5 Resumen de resultados

Para finalizar, se muestran a continuación las predicciones del próximo evento de falla obtenidas mediante el modelo de Uso de vida de Weibull en la Figura 3-19, mientras que en la Tabla 3-9, se muestra el resumen de los resultados de rendimiento de este modelo, incluyendo los del modelo propuesto aplicado en la fase de entrenamiento y en la fase de prueba.

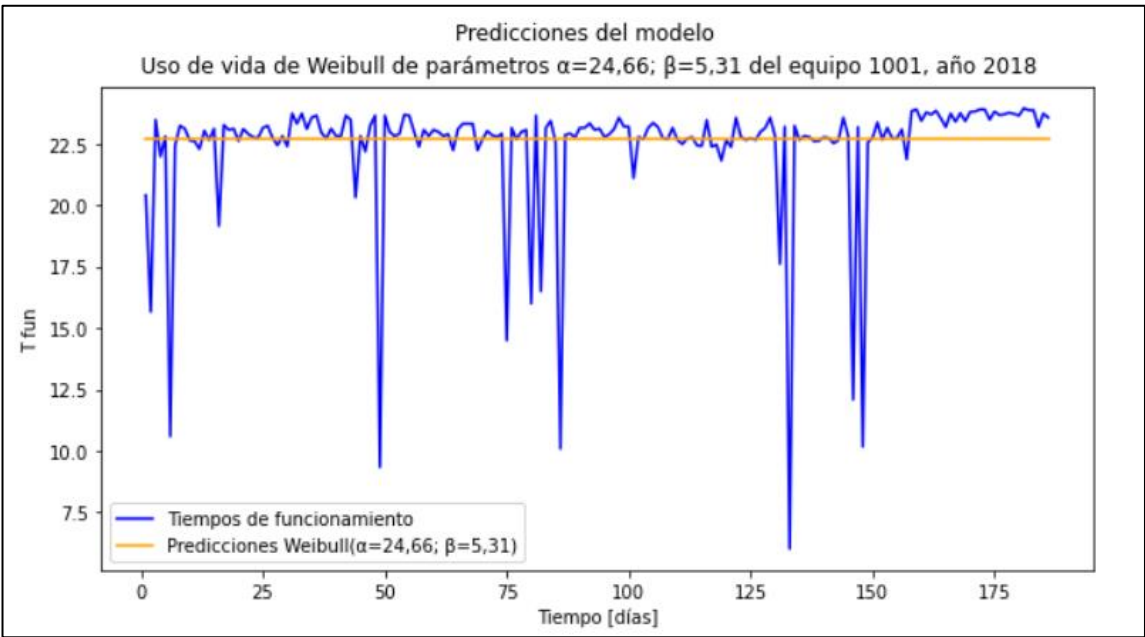


Figura 3-19. Predicciones del modelo de Uso de vida de Weibull de parámetros ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$)

Fuente: Elaboración propia mediante lenguaje de programación Python.

Tabla 3-9. Resumen de los resultados finales de rendimiento del modelo clásico de predicción de Uso de vida de Weibull junto con los resultados de las fases de entrenamiento y prueba, con un modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5) y un modelo de ML: $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$

	Bias		Exactitud				Precisión		Conf.
Modelo	ME	MdE	RMSE	MAPE	MAE	MdAE	MAD	SSD	MTBF
Weibull	-0,322	0,28	2,692	7,899	1,074	0,447	0,333	2,673	22,72
Entrenamiento SARIMA+SVR	-0,211	0,331	2,873	10,241	1,333	0,583	0,3	2,922	22,33
Prueba SARIMA+SVR	-0,077	0,13	0,479	1,531	0,354	0,271	0,192	0,472	23,56

Fuente: Elaboración propia.

Este análisis realizado, da cuenta de que el modelo de regresión SARIMA (4, 1, 0) (2, 0, 0, 5), junto con un modelo de ML: $SVR(kernel = rbf, C = 10000, \epsilon = 0.05, \gamma = 0.1)$, cuentan con los mejores resultados en cuanto a bias, exactitud y precisión, para la predicción de eventos de falla en el equipo 1001 del sistema Power Feed en estudio. Por este motivo, se puede decir que éste es recomendable para su uso, ya que supera en rendimiento al modelo de Uso de vida Weibull de parámetros ($\alpha = 24,66; \beta = 5,308; \gamma = 0$), representado en la Figura 3-19.

Con respecto al MTBF, como métrica de Confiabilidad, el tiempo promedio entre fallas obtenido mediante el modelo de Uso de vida de Weibull de parámetros ($\alpha = 24,66; \beta = 5,308; \gamma = 0$), es de 22,72 [h], con una desviación estándar $SSD = 2,673$ horas. Mientras que, con el modelo SARIMA+SVR propuesto, se obtuvo en la fase de prueba un MTBF de 23,56 [h] y una desviación estándar $SSD = 0,472$ horas. En resumen, el modelo de predicción propuesto cuenta con la capacidad de disminuir los errores de bias, exactitud y precisión, además de aumentar en un 3,7% la cantidad de horas de funcionamiento medio (MTBF), con una mínima desviación, logrando superar así al modelo de Uso de vida Weibull.

CONCLUSIONES

Se define en el Capítulo 1, el modelo de predicción clásico para contextualizar el estado del arte del Machine Learning y su relación con los procesos de gestión del mantenimiento industrial actual. En el Capítulo 2, se propone una metodología de predicción basada en un modelo de regresión (SARIMA) junto con un modelo de Machine Learning (SVR), para ser aplicada mediante un marco de trabajo y obtener los valores de rendimiento generados en cada ciclo de entrenamiento. El modelo SVR, utiliza como criterio una función kernel para definir de forma local el hiperplano que separará o clasificará los datos, junto con un margen que permite incluir mayor o menor cantidad de vectores de soporte, permitiendo así obtener más y mejor información de los errores dentro del proceso, lo cual promueve, a la vez, la mejora del aprendizaje para un modelo de predicción. Sin embargo, no importa cuán bien se genere un modelo, los datos siempre tendrán cierta cantidad de ruido o error irreducible que no es posible eliminar del todo. En el Capítulo 3, se evalúa la propuesta de metodología de predicción, mediante la comparación de las métricas de rendimiento obtenidas en los distintos modelos entrenados: Uso de vida de Weibull vs. SARIMA+SVR. De este análisis se concluye que existen patrones reconocibles en los datos de entrenamiento, y que, por lo tanto, se incluyen en el modelo de regresión SARIMA, parámetros no estacionales $(p, d, q) = (4, 1, 0)$ y estacionales $(P, D, Q, s) = (2, 0, 0, 5)$. Mientras que, para el modelo de ML: SVR, se utilizaron los hiperparámetros ($kernel = rbf$, $C = 10000$, $epsilon = 0.05$, $gamma = 0.1$), y para el modelo de Uso de vida de Weibull, se aplican los parámetros calculados en la Sección 1.3.2: ($\alpha = 24,66$; $\beta = 5,308$; $\gamma = 0$). De acuerdo con los resultados de la Tabla 3-9, es posible apreciar que el modelo propuesto SARIMA+SVR, presenta un mejor rendimiento en cuanto a errores de predicción, en comparación con el modelo clásico. En cuanto a métricas de Confiabilidad, el MTBF obtenido mediante el modelo de Uso de vida de Weibull es de 22,72 [h], mientras que con el modelo SARIMA+SVR propuesto, se obtuvo en la fase de prueba un MTBF de 23,56 [h]. Por lo tanto, debido a su capacidad de disminuir los errores de predicción, aumentar en un 3,7% la cantidad de horas de funcionamiento medio (MTBF), con una mínima desviación, se selecciona y propone este modelo predictivo como el mejor candidato para presentar en este proyecto de título, ya que, con él, se cumple la hipótesis inicial, de que un modelo de predicción, conformado por un modelo de regresión SARIMA y, otro de ML: SVR, supera en rendimiento al modelo de predicción clásico utilizado en la industria.

BIBLIOGRAFÍA

ACCIONA. (s.f.). *I'M INNOVATION*. Obtenido de <https://www.imnovation-hub.com/es/transformacion-digital/aprendizaje-reforzado-cuando-las-maquinas-aprenden-solas/>

Alrefae, S. A. (2022). *Hyperparameters optimization of support vector regression using black hole algorithm*. International Journal of Nonlinear Analysis and Applications. doi:10.22075/ijnaa.2022.6107

aprendeIA. (2022). Obtenido de <https://aprendeia.com/definicion-de-machine-learning/>

Baptista, M. S. (2017). Forecasting Fault Events for Predictive Maintenance using Data-driven Techniques and ARMA Modeling. *Computers & Industrial Engineering*.

Carvalho, T. S. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*.

DataScientest. (2022). Obtenido de <https://datascientest.com/es/nlp-natural-language-processing-introduccion>

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. Sebastopol, CA: O'Reilly Media, Inc.

Healt Data Miner. (2022). Obtenido de <https://healthdataminer.com/data-mining/aprendizaje-supervisado-y-no-supervisado/#:~:text=El%20aprendizaje%20supervisado%20supone%20que,de%20datos%20no%20etiquetados%20previamente>.

Indra Sistemas S.A. (2021). Obtenido de <https://www.indracompany.com/es/blogneo/deep-learning-sirve#:~:text=El%20Deep%20Learning%20llewa%20a,esta%20informaci%C3%B3n%20al%20siguiente%20nivel>.

Mathew J., L. M. (2017, September). Regression kernel for prognostic with support vector machines. *22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 1-5.

Qinming Liu, W. L. (2021). A New Support Vector Regression Model for Equipment Health Diagnosis with Small Sample Data Missing and Its Application. *Hindawi, Shock and Vibration*, 3-4.

Riial. (Mayo de 2022). Obtenido de <http://www.riial.org/que-es-la-inteligencia-artificial/>

SAS Institute Inc. (2022). SAS. Obtenido de https://www.sas.com/es_ar/insights/analytics/what-is-natural-language-processing-nlp.html

Saxena A., C. J. (2008). Metrics for evaluating performance of prognostic techniques. *Prognostics and Health Management*, 1–17. doi:10.1109/PHM.2008.4711436

Tech Target. (2021). *ComputerWeekly*. Obtenido de <https://www.computerweekly.com/es/definicion/Aprendizaje-profundo-deep-learning#:~:text=%C2%BFQu%C3%A9%20es%20el%20aprendizaje%20profundo,obtienen%20ciertos%20tipos%20de%20conocimiento>.

Tibco. (2022). Obtenido de <https://www.tibco.com/es/reference-center/what-is-supervised-learning#:~:text=El%20aprendizaje%20supervisado%20es%20una,de%20manera%20expl%C3%ADcita%20d%C3%B3nde%20buscar>.

ANEXOS

ANEXO A: ALGORITMO PARA ENTRENAMIENTO Y PRUEBA DE UN MODELO DE REGRESIÓN SARIMA

```
# Importar data
df = pd.read_excel('C:\\Base de datos_Python.xlsx').sort_values(by='i')
X=df.i
Y=df.ti
warnings.filterwarnings("ignore")
round(df,2)
```

```
#Graficar la data
plt.plot(X,Y, color = 'blue')
plt.title("Tiempos de funcionamiento equipo 1001, año 2018")
plt.xlabel("Tiempo [días]")
plt.ylabel("T fun")
plt.show()
plt.clf()
```

```
#Explorar los datos
round(df.describe(),2)
```

```
#Descomposicion de tendencia, estacionalidad y ruido
from pylab import rcParams
rcParams['figure.figsize'] = 10, 5
decomposition = sm.tsa.seasonal_decompose(df.ti, model='add', period=2)
fig = decomposition.plot()
plt.show()
```

```
#Escala de variables
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
sc_Y = StandardScaler()
X = sc_X.fit_transform(X.values.reshape(-1,1) )
Y = sc_Y.fit_transform(Y.values.reshape(-1,1))

# Dividir en datos de entrenamiento y prueba
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
print('La forma de X_train es: ', X_train.shape)
print('La forma de y_train es: ', y_train.shape)
print('La forma de X_test es: ', X_test.shape)
print('La forma de y_test es: ', y_test.shape)
```

```

#Entrenamiento
model_auto = pm.auto_arima(y_train, start_p=0, start_q=0,
                           test='adf', -----# use adftest to find optimal 'd'
                           max_p=5, max_q=5, -----# maximum p and q
                           m=5, -----# frequency of series
                           d=True, -----# Let model determine d'
                           seasonal=True, -----# No Seasonality
                           start_P=0,
                           D=0,
                           trace=True,
                           random=True,
                           error_action='ignore',
                           suppress_warnings=True,
                           stepwise=True)
print(model_auto.summary())

```

```

#Prueba
model_2 = SARIMAX(y_test, order=(4, 1, 0), seasonal_order=(2, 0, 0, 5))
model_fit = model_2.fit(dispatch=1)
print(model_fit.summary())

```

**ANEXO B: ANÁLISIS COSTO/BENEFICIO DE IMPLEMENTAR EL MODELO
PROPUESTO**

Antes de implementar un sistema de predicción, es de vital importancia comparar el Costo Global (*CG*) del ciclo de vida del equipo crítico en estudio, con y sin uso de un modelo de predicción de fallas. Este considera el costo de construirlo o adquirirlo y el costo de mantenerlo operativo, durante un cierto período de tiempo continuo de funcionamiento (*H*). El costo de las operaciones disminuirá si la vida útil restante (RUL) se predice con precisión, lo que dará como resultado que se reemplacen menos componentes antes de que sea necesario reemplazarlos y, potencialmente, menos fallas costosas. Por supuesto, el costo de operaciones también puede aumentar debido al costo de mantener y operar el sistema de pronóstico. Entonces, el cambio en el Costo Global del ciclo de vida causado por la adición de la capacidad de pronóstico sirve como una medida de los ahorros netos obtenidos al agregar pronósticos al sistema. Para efectos de evaluación de este cambio en el Costo Global, se utiliza una métrica económica conocida como Retorno de la Inversión o ROI, por sus siglas en inglés. El ROI es una métrica económica que indica ganancia o pérdida, y será positivo (ganancia), si agregar pronósticos reduce el costo del ciclo de vida, en caso contrario será negativo. El ROI nos dice la tasa de retorno de la inversión en pronósticos, lo que permite comparar la inversión en pronósticos con otras inversiones competidoras. El retorno de la inversión ROI, va más allá del costo del ciclo de vida al considerar también el valor del dinero en el tiempo. Sólo se debe realizar una inversión en pronósticos, si el ROI es al menos tan alto como el de otras inversiones potenciales con un riesgo similar (Saxena A., 2008).

En general:

$$ROI = \text{Ganancia} - \text{Inversión} \tag{41}$$

Asumiendo el costo global de un equipo como la inversión total a realizar dentro de un período de tiempo, podemos decir:

$$ROI = \text{Ganancia} - \text{Costo global} \tag{42}$$

Sin embargo, debemos comparar un escenario que aplica sistemas de predicción versus un escenario donde no se aplican sistemas de predicción ($ROI_{sin\ pronósticos}$ versus $ROI_{con\ pronósticos}$), o sea:

$$ROI_{\sin \text{pronósticos}} = \text{Ganancia} - CG_{\sin \text{pronósticos}} \quad (43)$$

, versus:

$$ROI_{\text{con pronósticos}} = \text{Ganancia} - CG_{\text{con pronósticos}} \quad (44)$$

Estas expresiones también son válidas comparar un escenario que aplica dos o más modelos de predicciones distintos. Sin embargo, como se indica anteriormente, solo se debe realizar una inversión en pronósticos, si el ROI con pronósticos es al menos tan alto como el ROI sin pronósticos o, en su efecto, tan alto como el de otras inversiones potenciales con un riesgo similar. De esta manera, se desarrolla a continuación, un análisis para proyectar y comparar el nivel de beneficio que se puede lograr obtener, al implementar el modelo de predicción propuesto, en contraste con el nivel de beneficio que se obtiene al aplicar el modelo de predicción clásico (Uso de vida de Weibull). Teniendo en cuenta que el Costo Global del ciclo de vida de un equipo se puede calcular como:

$$\text{Costo global} = \text{Costo de inversión} + \text{Costo de operación} \quad (45)$$

o también como:

$$CG = C_{\text{Inversión}} + [C_{\text{ejercicio}} + (1 - A_s) * H * C_{\text{inef}}] * Fa \quad (46)$$

donde $C_{\text{Inversión}}$ es el costo de inversión o adquisición, $C_{\text{ejercicio}}$ es el costo que resulta de hacer operar un equipo, $(1 - A_s)$ es la indisponibilidad del sistema, H es el tiempo en cada intervalo de operación, C_{inef} son los costos de ineficiencia o pérdidas debido a detenciones inesperadas y, Fa un factor de actualización, que corresponde al valor actual o valor presente de una inversión.

Para el caso del equipo en estudio, se tiene que el MTBF obtenido mediante el modelo de Uso de vida de Weibull es de 22,72 [h], mientras que con el modelo SARIMA+SVR propuesto, se tiene un MTBF de 23,56 [h]. Por lo tanto, para cada modelo, los valores de las métricas de Confiabilidad: Tiempo medio de reparación (MTTR), Tasa de fallas (λ) y Disponibilidad esperada (A_s), varían, como se muestra en la Tabla A mostrada a continuación:

Tabla A. Comparación entre las métricas de Confiabilidad de los modelos de predicción clásico y el modelo de predicción propuesto, utilizadas para programar las detenciones preventivas del equipo 1001 en estudio

Métricas de Confiabilidad		Modelo	
		Weibull	SARIMA+SVR
Tiempo medio de funcionamiento [h]	MTBF	22,72	23,56
Tiempo medio de reparación [h]	MTTR	1,28	0,44
Tasa de fallas [1/h]	λ	0,044	0,042
Disponibilidad esperada del sistema [-]	A_s	0,947	0,982

Fuente: Elaboración propia.

El coeficiente entre $(1 - A_s) * H * C_{inef}$ de la Ec. 46, corresponde al costo de Destrucción de negocio, y es importante controlar para la toma de decisiones operativas y gestión del mantenimiento. Por lo tanto, con los datos de Indisponibilidad esperada $(1 - A_s)$ de cada modelo, los $C_{inef} = 7694,4 [\frac{USD}{h}]$ del equipo en estudio, y un horizonte de tiempo correspondiente a $H = 4464$ horas de operación (186 días de 24 horas cada uno), se procede a estimar el costo de destrucción de negocio, como se muestra en la Tabla B, a continuación:

Tabla B. Estimación de los costos de destrucción de negocio que se obtienen aplicando distintos modelos de predicción al equipo en estudio

Cálculo de Destrucción de negocio		Modelo	
		Weibull	SARIMA+SVR
Indisponibilidad esperada [-]	$1 - A_s$	0,053	0,018
Horizonte de tiempo [h]	H	4464	4464
Costo de Ineficiencia [USD/h]	C_{inef}	\$ 7.694,4	\$ 7.694,4
Destrucción de negocio [USD]	$(1 - A_s) * H * C_{inef}$	\$ 1.831.883	\$ 629.710

Fuente: Elaboración propia.

Suponiendo que se tiene un costo de inversión $C_{Inversión} = 7000 \text{ [USD]}$, un costo de ejercicio $C_{ejercicio} = 3200 \text{ [\frac{USD}{año}]}$, una tasa de interés de mercado del 17%, y asumiendo, además, que se desea proyectar los costos globales en un período de 5 años, considerando el costo de destrucción de negocio de cada modelo de predicción (como se muestra en la Tabla B), podemos dar cuenta de que, si se aplica el modelo de predicción propuesto SARIMA+SVR, es posible obtener un aumento del 34,74% de ahorros en el Costo Global (CG) del equipo en estudio, lo que se traduce en un ahorro monetario igual a 1.202.173 [USD], como se muestra en la Tabla C a continuación:

Tabla C. Cálculo del porcentaje de variación de los Costos Globales (CG) que se obtiene al aplicar distintos modelos de predicción al equipo en estudio

Cálculo de la variación de Costos Globales		Modelo	
		Weibull	SARIMA+SVR
Tasa de interés [%]	r	0,17	0,17
Numero periodos [años]	n	5	5
Factor de actualización	Fa	3,199	3,199
Costo de Ejercicio [USD/año]	C_{ejercicio}	\$ 3.200	\$ 3.200
Costo de Operación [USD]	C_{operación}	\$ 1.835.083	\$ 632.910
Costo de Inversión [USD]	C_{Inversión}	\$ 7.000	\$ 7.000
Costo Global [USD]	CG	\$ 1.842.083	\$ 639.910
Diferencia de ahorro en costos		\$ 1.202.173	
Porcentaje de variación		34,74%	

Fuente: Elaboración propia.

Como se puede apreciar, la reducción de indisponibilidad (o aumento de disponibilidad) del equipo en estudio, disminuye el costo de destrucción de negocio, lo cual implica, además, una disminución en el costo de operación y, por lo tanto, del costo global. Si aumentar el rendimiento de predicción implica aumentar la vida útil restante de un componente (por ende, su MTBF), entonces también se puede decir y aseverar que: la implementación de sistemas predictivos en equipos críticos genera un impacto positivo en los indicadores de Confiabilidad, logrando aumentar la disponibilidad esperada $A = MTBF / (MTBF + MTTR)$, a la vez que disminuye la tasa o frecuencia media de falla $\lambda = 1 / MTBF$.

Como se menciona anteriormente, sólo se debe realizar una inversión en pronósticos, si el ROI es al menos tan alto como el de otras inversiones potenciales con un riesgo similar, por lo tanto, suponiendo que se tienen ganancias de 5.000.000 [USD] y aplicando la Ec. 42, como se muestra en la Tabla D, se tendrá un mayor Retorno de la Inversión si se aplica el modelo de predicción propuesto SARIMA+SVR, en una proporción de un 38,07% con respecto al modelo de predicción clásico.

Tabla D. Estimación del porcentaje de variación del Retorno de la Inversión al aplicar distintos modelos de predicción al equipo en estudio

Inversión en pronósticos		Modelo	
		Weibull	SARIMA+SVR
Ganancia [USD]	<i>G</i>	\$ 5.000.000	\$ 5.000.000
Costo Global [USD]	<i>CG</i>	\$ 1.842.083	\$ 639.910
Retorno de la inversión [USD]	<i>ROI</i>	\$ 3.157.917	\$ 4.360.090
Porcentaje de variación		38,07%	

Fuente: Elaboración propia.

Mediante este análisis se comprueba que, mediante la implementación del modelo de predicción propuesto SARIMA+SVR, en primer lugar, se podría aumentar el MTBF con el que se programan las mantenciones preventivas del equipo 1001 en estudio, lo cual implicaría, además, una mejora en las métricas de Confiabilidad, logrando un aumento de la disponibilidad esperada y una disminución del costo de operación (junto con el costo de destrucción de negocio), por lo tanto, se reduciría también el Costo Global del ciclo de vida, en una proporción de un 34,74%. Se puede proyectar, además, que, si se aplica el modelo de predicción propuesto al caso de estudio presentado, podría mejorar también, el Retorno de la Inversión en pronósticos en un 38,07%.

Por otro lado, de esta implementación se puede esperar también, un incremento en la efectividad total del equipo, métrica de producción más conocida como: OEE, la cual se calcula como $OEE = A * Q' * \eta$, es decir, la multiplicación entre la disponibilidad (*A*), la tasa de calidad (*Q'*) y el rendimiento del proceso de estudio (η). Asumiendo, para este caso, que: $Q' = 0,82$ y $\eta = 0,75$, y considerando los valores de disponibilidad de cada modelo (indicados en la Tabla A), se puede esperar obtener un 3,7% de aumento en la producción, como se observa en la Tabla E, mostrada a continuación:

Tabla E. Estimación del porcentaje de variación esperado de la producción que se obtiene al aplicar distintos modelos de predicción al equipo en estudio

Métricas de producción		Modelo	
		Weibull	SARIMA+SVR
Disponibilidad del sistema [-]	A	0,947	0,982
Tasa de calidad del proceso [-]	Q'	0,82	0,82
Tasa de eficiencia del proceso [-]	η	0,75	0,75
Efectividad total del equipo [%]	OEE	58,22%	60,37%
Porcentaje de variación esperado de producción		103,70%	
Aumento esperado de la producción		3,70%	

Fuente: Elaboración propia.

ANEXO C: VENTAJAS Y DESVENTAJAS DEL MODELO DE PREDICCIÓN
PROPUESTO

Ventajas:

- El uso de modelos predictivos permite disminuir el costo de las operaciones al reducir la cantidad de reemplazos de repuestos, mediante un cálculo más exacto sobre la vida útil remanente de un componente, ya que podemos hacer uso de un activo durante un mayor tiempo antes de que se genere la falla o avería, como se muestra en la Figura A, presentada a continuación:

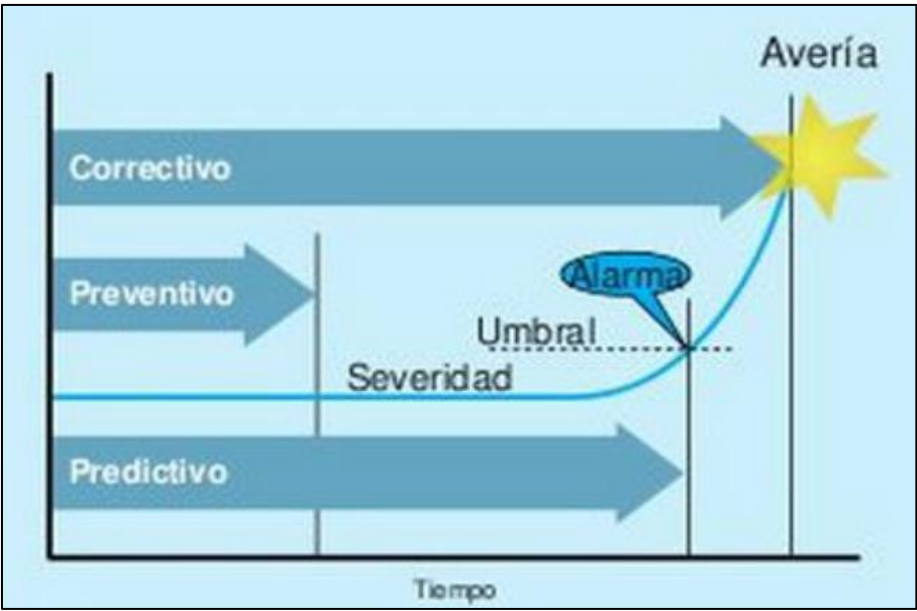


Figura A. Comparación del impacto que genera el uso de distintas estrategias de mantenimiento

Fuente: Basado en asignatura Gestión de Activos (BALDI, 2020).

- Debido a su naturaleza compleja y su alta capacidad de mapeo, las SVR aplican por sí mismas para series de tiempo estacionarias y no estacionarias, por lo tanto, es posible disminuir el tiempo de exploración y preprocesamiento en la mayoría de los casos.
- La ventaja de utilizar previamente un modelo ARIMA o SARIMA, es para generar un aprendizaje secuencial que permita mantenerlo o mejorarlo a medida que se van agregando datos con el paso del tiempo.

- Los tiempos de ejecución de estos análisis son bastante cortos gracias a la utilización del lenguaje de programación Python.
- Es posible continuar investigando sobre estos métodos de predicción, mediante la aplicación de otros modelos de ML.

Desventajas:

- Debido a su naturaleza, el modelo SVR es complejo de entender y de aplicar.
- La velocidad del procesamiento de datos para predicción se ve afectada cuando no se cuenta con algoritmos configurados apropiadamente.
- A pesar de los resultados, el modelo de Weibull sigue siendo importante para el análisis de otras métricas como lo es la Confiabilidad, Mantenibilidad, Tasa de fallas, Probabilidad de falla instantánea, etc. Por ende, no es posible descartar su uso del todo y es recomendable aplicar ambos modelos y analizar con más información las variables en juego, antes de tomar alguna decisión.
- El costo de implementar modelos predictivos puede no ser rentable en todos los casos. Es necesario realizar un estudio técnico sobre la implementación de estas tecnologías y, así, obtener información para cuantificar las inversiones y los costos de operación.
- Es de vital importancia que los datos de entrada a analizar reflejen de forma real y fidedigna el comportamiento de la variable en estudio. Esto quiere decir, que los datos sean de calidad y, así, evitar que se analicen valores manipulados, inventados o erróneos. Esto podría causar efectos altamente negativos al proceso de predicción.