EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Nuevo León

INSTITUTO TECNOLÓGICO DE NUEVO LEÓN

División de Estudios Profesionales

Trabajo de Titulación

Opción TI Informe Técnico de Residencia Profesional por Tesis

Proyecto: “Evaluación de algoritmo de Aprendizaje Automático mediante Bosques Aleatorios para el seguimiento de variables críticas en un proceso de manufactura”

ALUMNO(S):	Adrián Fernando Agundis Martínez
No. CONTROL:	18481451
CARRERA:	Ingeniería en Sistemas Computacionales
ASESOR DE RESIDENCIA:	Dr. José Isidro Hernández Vega

Guadalupe, N.L. Enero, 2023

Aceptación de documento de Tesis

Cd. Guadalupe, Nuevo León, 24/Enero/2023

ING. MAGALY BENÍTEZ TAMEZ
JEFA DE DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN
PRESENTE:

La Comisión de Revisión de Tesis nos es grato comunicarle que, conforme a los lineamientos de los planes de estudio del 2010 del Tecnológico Nacional de México para la obtención del grado de Ingeniería en Sistemas Computacionales de este Instituto, y después de haber sometido a revisión académica el proyecto de Tesis titulado: **"Evaluación de algoritmo de aprendizaje automático mediante Bosques Aleatorios para el seguimiento de variables críticas en un proceso de manufactura",** realizado por **Adrián Fernando Agundis Martínez,** Número de Control: **18481451,** dirigida por el **Dr. José Isidro Hernández Vega,** y habiendo realizado las correcciones que le fueron indicadas, acordamos **ACEPTAR** el documento final de proyecto de Tesis. Así mismo le solicitamos tenga a bien extender la documentación correspondiente para continuar el proceso de titulación integral por tesis del sustentante.

Sin otro particular, agradecemos la atención.

ATENTAMENTE
Excelencia en Educación Tecnológica®
"CIENCIA Y TECNOLOGÍA AL SERVICIO DEL HOMBRE"

DIRECTOR DE TESIS

DR. JOSÉ ISIDRO HERNÁNDEZ VEGA
DOCTORADO EN INGENIERÍA CON ORIENTACIÓN EN TECNOLOGIAS DE LA INFORMACIÓN
CÉDULA: 12058578

REVISOR

M. C. ELDA REYES VARELA
MAESTRÍA EN CIENCIAS EN COMERCIALIZACIÓN DE LA CIENCIA Y LA TECNOLOGIA
CÉDULA: 09093449

REVISOR

ING. LUIS ALEJANDRO REYNOSO GUAJARDO
INGENIERÍA EN SISTEMAS COMPUTACIONALES
CÉDULA: 5157710

c.c.p Departamento de Sistemas y Computación
c.c.p. Expediente
c.c.p. Interesados

RESUMEN.

Esta tesis aborda la problemática de aplicación de algoritmos de aprendizaje automático para el monitoreo de variables críticas en un proceso de manufactura. El uso de algoritmos aplicados a este tipo de problemas puede ayudar al pronóstico de detección de fallas, comportamiento en función de un historial de datos. Existe una diversidad de algoritmos de aprendizaje automático, en este caso se evalúo el de bosques aleatorios y el algoritmo Prophet. El objetivo de esta investigación fue evaluar el desempeño del algoritmo de Bosques Aleatorios (Random Forest) en procesos de manufactura para su estudio en el seguimiento de variables críticas con respecto a otros algoritmos. La metodología que se usó fue implementar el desarrollo de una aplicación del modelo de bosques aleatorios para el seguimiento de diversas variables como lo son temperatura, presión, aire, torque, velocidad de rotación e inclusive se abordó brevemente en series de tiempo, con el fin de evaluar con el algoritmo Prophet en base una serie de métricas en común.

Los resultados principales obtenidos en los experimentos se presentan en métricas de evaluación como lo son la Puntuación R2, varianza explicada, error absoluto medio porcentual, error absoluto medio y error cuadrático medio de los cuales los últimos tres se muestran mediciones en común en comparación al algoritmo Prophet. El desarrollo de los experimentos permitió con los datos utilizados evaluar las métricas de desempeño y precisión del algoritmo, comparado con los valores esperados por cada métrica, teniendo perdidas mínimas de información.

Palabras clave.

Algoritmos de Aprendizaje Automático, pronóstico de fallas, mantenimiento predictivo, Bosques Aleatorios, Sobreajuste.

ABSTRACT.

This thesis addresses the problem of applying machine learning algorithms to monitor critical variables in a manufacturing process. The use of algorithms applied to this type of problem can help to forecast failure detection, behavior based on a history of data. There is a diversity of machine learning algorithms, in this case the random forests and the Prophet algorithm were evaluated. The objective of this research was to evaluate the performance of the Random Forest algorithm in manufacturing processes for its study in the monitoring of critical variables with respect to other algorithms. The methodology that was used was to implement the development of an application of the random forest model to monitor various variables such as temperature, pressure, air, torque, rotation speed and was even briefly addressed in time series, in order to evaluate with the Prophet algorithm based on a series of common metrics.

The main results obtained in the experiments are presented in evaluation metrics such as the R2 Score, explained variance, percentage mean absolute error, mean absolute error and mean square error of which the last three are common measurements compared to the algorithm. Prophet. The development of the experiments allowed with the data used to evaluate the performance and precision metrics of the algorithm, compared with the expected values for each metric, having minimal information losses.

Keywords.

Machine Learning algorithms, failure forecasting, predictive maintenance, Random Forest, Overfitting.

AGRADECIMIENTOS.

Este trabajo no hubiera sido posible sin la ayuda del Dr. José Isidro Hernández Vega por su constante asesoramiento y apoyo durante la elaboración de este proyecto, siempre al tanto de nuestro trabajo y brindando todo tipo de asistencia para facilitar el trabajo.

Al Instituto Tecnológico de Nuevo León por todo el conocimiento impartido a lo largo de mi carrera, así como a la División de estudios de Posgrado e Investigación por las permitir desarrollar este proyecto.

A mi familia, mis padres y mi hermano porque me permitieron en todo momento concentrarme en mis estudios a lo largo de toda mi formación educativa contando con su respaldo en todo momento, así como aconsejándome siempre que lo necesité.

ÍNDICE.

INDICE DE FORMULAS.

INDICE DE TABLAS.

INDICE DE FIGURAS.

CAPITULO I. INTRODUCCIÓN.

1.1 MOTIVACIÓN DEL PROBLEMA.

Monterrey, N.L. se ha caracterizado a nivel nacional e internacional por su importancia en el sector de la industria. Considerada la capital industrial, Monterrey es la tercera ciudad más grande del país, mientras que Nuevo León aporta el 8.06% del PIB al país, según un estudio realizado Citibanamex en el estudio económico Indicadores regionales de Actividad Económica, 2021, convirtiendo a la Industria regiomontana en un factor muy importante para todo el país.

En todas las empresas de manufactura se van a enfrentar constantemente con nuevas regulaciones para cumplir con una amplia variedad de protocolos de seguridad, ambientales, de calidad, etc. Los fabricantes deben asegurarse de que tienen completo control del proceso para que puedan demostrar el cumplimiento de dichas regulaciones, estas regulaciones requieren de la capacidad de rastrear fallos específicos en todo el manejo de producción. Entre las causas más comunes de fallas en el área industrial se encuentra la mala capacitación a los métodos de trabajo y al mantenimiento de los equipos. Además, con el progreso constante de maquinaria y tecnología empleada en las instalaciones, esta cambia y se adapta continuamente, por lo que es necesaria una constante revisión de los procesos de manufactura.

Conocer las variables críticas en un proceso de manufactura nos permitirá medir cuan eficaces son estos procesos. El implementar un algoritmo que asegure la eficiencia y eficacia de estas variables nos asegura que en todo momento estarán siendo supervisadas para conocer su estado, ser analizadas y poder optimizarse buscando siempre cumplir los protocolos. Por esto es importante identificar nuestras variables criticas durante el proceso y cuál es el peso relativo de esta, de forma que podamos comparar con mayor precisión y tener un análisis más efectivo de cada proceso de manufactura.

La mala planeación es una de las causas que más daño hacen en este sector. En este punto se debe considerar previamente todos y cada uno de los aspectos del proceso de manufactura y tomar en cuenta la periodicidad del análisis de las distintas herramientas usadas en el proceso, una mala planeación puede llevar desde un gasto excesivo en mantenimiento a incluso tener un daño irreparable en la maquinaria utilizada en la producción causando costos aún mayores para la industria.

Otro problema muy común se presenta en la falta de control de los procesos, pues sin un mantenimiento preventivo a las maquinas se perderá tiempo y dinero en la reparación de máquinas que no fueron revisadas a tiempo.
El problema que se busca resolver es evitar paros no programados en el proceso, dar seguimiento al comportamiento de las variables evitar una mala planeación de mantenimiento en procesos de manufactura.

Lo que se busca realizar es la evaluación de un algoritmo de análisis de variables críticas utilizando bosques aleatorios para mantenimiento predictivo en procesos de manufactura.

1.2 PLANTEAMIENTO DEL PROBLEMA A RESOLVER.

El problema se plantea de la siguiente manera: Dada una serie de sensores de distintos tipos. Se cuenta con una base de datos con la información captada por estos sensores, procesada en un formato digital para ser analizada para toma de decisiones sobre predicción de fallas y mantenimiento predictivo.

Los gastos generados por mantenimiento son más que solo el gasto de reparación de las máquinas, pues el tiempo que una maquina descompuesta dure sin ser reparada significa menor producción y recursos desperdiciados, de este modo se espera que planteando un algoritmo que ayude a mejorar los procesos de manufactura se puedan predecir y detectar todas las fallas a tiempo y repararlas antes de que estas comiencen a ser un problema.

El utilizar un algoritmo de Machine Learning como Random Forest nos permite analizar grandes cantidades de información en un breve periodo de tiempo, algo que nos permite llevar un registro histórico de la información de las máquinas y sensores utilizados en los procesos de manufactura, que será útil para poder analizar tal cantidad de información, generando gráficos y datos más entendibles, haciendo un resumen de dicha información.

Este algoritmo presenta resultados claros y precisos cuando se analiza una gran cantidad de información pudiendo generar regresiones y clasificaciones de todos los datos analizados en el algoritmo además de presentar ventajas sobre otros algoritmos pues al utilizar múltiples arboles reduce el riesgo de sobreajuste.

Es importante conocer el comportamiento de los sensores y maquinas en un largo plazo, pues entre más amplio el volumen de información que se maneja, mejor comportamiento del algoritmo.

1.3 ANTECEDENTES.

1.3 .1 TRABAJOS RELACIONADOS CON EL TEMA DE INVESTIGACIÓN.

En el trabajo de *A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests (2017)* de Dazhong Wu, Connor Jennings, Janis Terpenny, Robert X. Gao, Soundar Kumara, hablan sobre el objetivo de integrar big data, análisis avanzado, computación de alto rendimiento, así como Internet industrial de las cosas en procesos de manufactura inteligente. Demuestran en su investigación que el modelo predictivo entrenado por el algoritmo Random Forest puede predecir el desgaste de una herramienta en procesos de manufactura con mucha precisión, así como queda comprobado que Random Forest superan en rendimiento a otros algoritmos como feed-forward back propagation, redes neuronales artificiales y regresión de vectores de soporte.

En la publicación de K Antosz, E Kozłowski, J Sęp y T Żabiński llamada *The use of random forests to support the decision-making process for sustainable manufacturing (2022)* demuestran que utilizando el algoritmo para predecir el estado futuro de una hoja de corte se puede obtener con una precisión de 0.9840 la alta calidad del modelo de predicción, lo cual ayudará en el proceso de toma de decisiones en la determinación de la vida útil de las herramientas de corte.

En la investigación *Anomalies Detection in Smart Manufacturing Using Machine Learning and Deep Learning Algorithms* de Mohamed Gamal, Ahmed Donkol, Ahmed Shaban, Francesco Costantino, Giulio Di Gravio and Riccardo Patriarca, (2021) compararon varios algoritmos, siendo Random Forest el que mejor recuperación, exactitud y precisión con respecto a sus competidores.

En la investigación de Aji Teguh Prihatno, himawan nurcahyanto, Yeong Min Jang, *Predictive Maintenance of Relative Humidity Using Random Forest Method* donde *(2021)* donde se analizaron procesos de manufactura casos de humedad relativa, al momento de medir el Porcentaje absoluto de error (MAPE, por sus siglas en inglés) utilizando el método de Random Forest, se obtuvo un valor del 82.49% de precisión del valor de las predicciones, comparadas con el resultado real lo que significa que es una precisión lo suficientemente precisa para considerar factible el implementarse en procesos de manufactura reales para dar el mejor resultado de precisión con el algoritmo.

1.4 PREGUNTA DE INVESTIGACIÓN.

El fallo de máquinas en una línea de manufactura se puede predecir haciendo uso de ML mediante el algoritmo de bosques aleatorios. Se busca encontrar la relación entre variables para detectar fallos y patrones en procesos de manufactura, encontrando una solución buena en un tiempo de computo razonable.

1.5 OBJETIVOS.

1.5.1 General.

Evaluar el desempeño del algoritmo de Bosques Aleatorios (Random Forest) en un proceso de manufactura de estudio para el seguimiento de variables críticas con respecto a otros algoritmos.

1.5.2 Específicos.

- Implementar el algoritmo mediante la programación para el análisis de variables.
- Representar gráficamente los resultados mediante el uso del algoritmo de Bosques Aleatorios.
- Probar el algoritmo en procesos reales de manufactura.
- Evaluar la precisión y calidad del algoritmo en diferentes situaciones en procesos de manufactura.
- Comparar el desempeño del algoritmo con respecto a otros similares.

1.6 JUSTIFICACIÓN DEL PROYECTO.

Con esta investigación se buscó evaluar un algoritmo que pueda ser de utilidad en la industria manufacturera optimizando el proceso de detección de fallas y predicción comparando entre dos variables usando el algoritmo de Random Forest. Basado en el fundamento teórico en el trabajo de Luca Puggini, John Doyle, Sean McLoone llamado Fault Detection using Random Forest Similarity Distance (2015) demuestran que el método es particularmente adecuado para detección de anomalías en escenarios de big data.

Se buscó construir la solución mediante el uso de herramientas de programación utilizando el lenguaje Python, apoyado de librerías como Scikit-learn para analizar los datos generados, una vez sean digitalizados. Se obtuvieron resultados a partir de los sensores que estarán almacenando todos los resultados que estos muestren en un periodo de tiempo para posteriormente ser procesados por el algoritmo generando datos gráficos que busca ayudar a facilitar el análisis para detección de fallas mediante el análisis de las variables críticas en la industria.

1.7 IMPACTO O BENEFICIO EN LA SOLUCIÓN A UN PROBLEMA RELACIONADO CON EL SECTOR PRODUCTIVO O LA GENERACIÓN DEL CONOCIMIENTO CIENTÍFICO O TECNOLÓGICO.

Con esta investigación se buscó poder dar seguimiento a variables críticas en procesos de manufactura reales con la implementación de algoritmo Random Forest, además de contribuir al desarrollo de algoritmos de Machine Learning como parte de la formación de ingeniero en sistemas computacionales en el área de inteligencia artificial.

1.8 LUGARES DONDE SE DESARROLLÓ EL PROYECTO.

La investigación se llevó a cabo dentro de las instalaciones del Instituto Tecnológico de Nuevo León, una institución educativa federal de nivel de estudios superiores. Perteneciente al sistema del Tecnológico Nacional de México, ubicado en el municipio de Guadalupe, en el estado de Nuevo León, en México.

1.9 INFRAESTRUCTURA.

En el desarrollo de la investigación se utilizó una Laptop de la marca Dell con un procesador Intel Core i7-7500U, 16 GB de RAM en donde se utilizó el editor de texto y código Sublime Text utilizando el lenguaje de programación Python apoyado de las librerías Numpy, Pandas, Matplotlib y Seaborn, además de la librería Scikit-learn utilizada especialmente para el algoritmo de Random Forest.

CAPITULO II. MARCO TEÓRICO.

2.1 Python.

Es un lenguaje de programación interpretado de alto nivel. Su principal característica es la facilidad que brinda el lenguaje de ser leído, por lo que es utilizado en empresas de todo el mundo para elaborar aplicaciones web, análisis de datos, automatización de operaciones, etc.

2.1.1 NumPy.

Es una librería de Python especializada principalmente en calculo numérico, enfocado en grandes volúmenes de datos utilizando matrices para representas conjuntos de datos en varias dimensiones.

2.1.2 Pandas.

Pandas es una librería utilizada en análisis de datos para simplificar información, diseñada originalmente como una alternativa a las hojas de cálculo.

2.1.3 Matplotlib.

Matplotlib es una librería de Python de código abierto utilizada para crear diagramas de barra, histogramas, mapas de calor y otras gráficas. Fue desarrollada como una alternativa de código abierto de MATLAB.

2.1.4 Scikit-learn.

Es una librería gratuita para el lenguaje de programación Python. Ofrece distintos tipos de algoritmos (Entre estos, Random Forest) de clasificación, regresión, reducción de dimensionalidad, etc. Esta librería presenta compatibilidad con otras librerías, tales como NumPy o Matplotlib, entre otras.

2.2 Modelo Lineal y Regresión.

El modelo lineal simple consiste en dos variables, una predictor (x) y una variable de respuesta (y) modeladas por la siguiente ecuación:

$$\hat{Y} = \beta_0 + \beta_1 \cdot X_1 \quad (1)$$

En casos donde hay más de una variable a tomar en cuenta, se utilizan subíndices:

$$\hat{Y} = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 \quad (2)$$

Ejemplo.

$$\hat{Y} = 75.4 + (-0.76) \cdot X \quad (3)$$

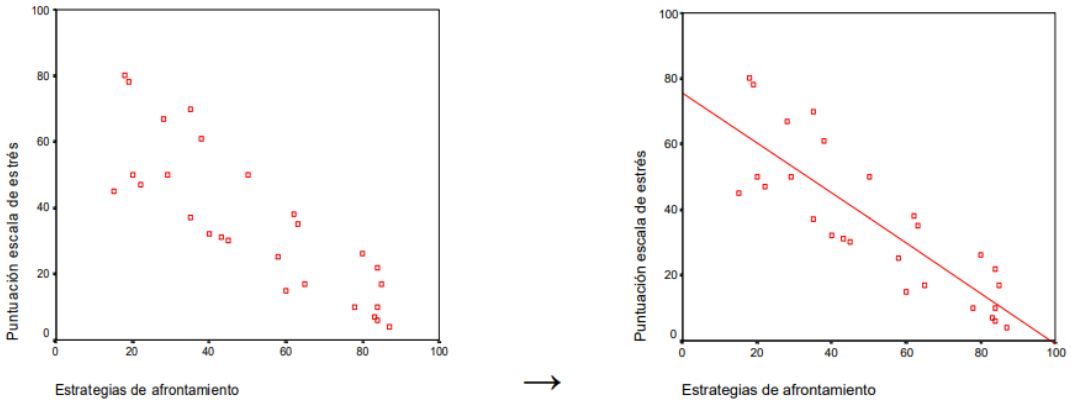

Figura 1. Ejemplo de modelo de regresión lineal.

2.3 Ingeniería de Características.

La ingeniería de características es el momento en el que se definen las características que se utilizarán durante el entrenamiento del modelo, la precisión del modelo dependerá de cuan exactas sean estos conjuntos de características. Lo primero a tener en cuenta es la extracción de datos, es un proceso largo pues deben recopilarse de distintas fuentes de almacenamiento, más aún puesto que Random Forest es un algoritmo enfocado a Big Data, dichos datos deben estar unificados en un solo formato, pues cada fuente tendrá su propio método de ordenar datos y cuanto mayor sea nuestro volumen de datos, mayor será también el esfuerzo a aplicar en asegurar la calidad de los mismos.

Una vez estos datos sean recopilados, deben ser etiquetados acorde a su etiquetas significativas o descriptivas que permitan a nuestro modelo comprender estos datos y así aprender de estos. Una vez los datos sean ordenados y etiquetados deben ser interpretados, generando gráficos, histogramas o clasificaciones con el fin de detectar patrones, fallas o verificar información de manera gráfica y simple.

2.4 Ciencia de datos.

El Data Science utiliza distintas herramientas como métodos científicos, inteligencia artificial y análisis de datos, utilizando programación, estadística y matemáticas. Como tal Data Science y Machine Learning son subtemas de la inteligencia artificial, mientras que el Data Science se enfoca buscar información y conocimiento de los datos, el Machine Learning busca de generar información a partir de los datos y aplicarlo en Inteligencia Artificial.

El Data Science consta de 6 etapas principales en su ciclo de vida:

-Planificación: Definir el proyecto.

- Crear un modelo: Se realiza mediante programación con apoyo de interfaces de programación y librerías de código abierto para gestión de datos y creación gráficos.

- Evaluación del modelo: En esta fase se busca que la precisión del modelo sea mayor, buscando porcentajes elevados que den solidez al modelo para ser implementado.

-Explicación: Consiste en demostrar que los resultados del modelo son automatizados acorde a su relevancia y valorando los factores que intervienen en el proceso de desarrollo de una predicción.

-Implementación: Aplicar un modelo a un sistema generalizado es una de las partes más difíciles, pues se debe escalar y replicar en distintas áreas.

-Supervisión: Una vez el modelo se implemente, se debe asegurar su funcionamiento, debe estar bajo supervisión para garantizar su funcionamiento, pues pueden surgir nuevos problemas que no existían al momento del desarrollo del modelo, problemas que deben ser tomados en cuenta como datos para implementarlos al modelo.

2.5 Mantenimiento Predictivo.

El mantenimiento predictivo se basa en, mediante la supervisión de un proceso de manufactura, conocer datos de las máquinas para detectar si existe algún problema antes de que este problema se produzca. A diferencia del mantenimiento preventivo, que es un conjunto de acciones tomadas para mantener las maquinas en funcionamiento evitando tiempo perdido, anticipándose a fallas inesperadas, el mantenimiento predictivo busca detectar los fallos antes de que ocurran, evitando así, procesos de mantenimiento en tiempos innecesarios.

En el área de mantenimiento predictivo se utiliza la inteligencia artificial o el Machine Learning para generar procesos de mantenimiento predictivo más rápidos y eficaces. Estos procesos recopilan los datos capturados en tiempo real, así como datos históricos para mejorar el entrenamiento del proceso, haciendo así que las maquinas aprendan, por medio del algoritmo, a detectar fallas de manera autónoma, haciendo que de esta manera la misma maquina alerte cuando es que se tiene que reemplazar una pieza, detener un proceso o directamente parar la producción temporalmente para evitar un fallo que podría ser perjudicial para una industria.

2.6 Machine Learning.

El aprendizaje automático es una rama de la inteligencia artificial que ofrece la posibilidad de que las maquinas sean entrenadas para aprender de los datos que se ingresan y no de las instrucciones programadas.

En su libro, *Machine Learning* (1997), Tom Mitchell menciona que el diseño de un entorno enfocado al aprendizaje automático implica una serie de opciones de diseño, donde se debe tomar en cuenta el tipo de entrenamiento, el objetivo que se quiere aprender, una representación para la función de este objetivo y un algoritmo que le permita aprender a partir de una serie de ejemplos de entrenamiento.

Según, Peter Falch, en el documento *Machine Learning: The Art and Science of Algorithms that Make Sense of Data* el aprendizaje automático es el estudio sistemático de algoritmos y sistemas que mejoran su conocimiento o desempeño con la experiencia.

2.6.1 Aprendizaje supervisado.

Es una rama de Machine Learning, se utiliza como método de análisis de datos en donde algoritmos aprenden de los datos para generar nueva información sin necesidad de ser programada.

Puede ser de dos Maneras:

- Clasificación: El algoritmo etiqueta por clases la información.
- Regresión: El algoritmo predice resultados basándose en un rango de valores posibles de entradas pasadas.

2.6.2. Aprendizaje no supervisado.

A diferencia del aprendizaje supervisado, en este no se dispone de datos para el entrenamiento, por lo que solo se puede describir la estructura de datos para encontrar la organización.

El aprendizaje no supervisado se suele usar en:

- Problemas de clustering, una tarea que consiste en agrupar un conjunto de objetos (no etiquetados) en subconjuntos de objetos llamados Clusters.
- Agrupamientos de co-ocurrencias
- Perfilado o profiling.

2.7 Random Forest.

El modelo Random Forest se forma de un conjunto de árboles de decisión individuales, utilizando distintas muestras de datos de entrenamiento. Es utilizado para entrenar datos en base a información histórica para generar tendencias y clasificaciones posibles. El modelo se forma a través del conjunto de predicciones creadas de cada árbol individual.

2.7.1. Definiciones:

- Judith Sandoval publicó en la Convención de Centroamérica y Panamá el articulo *Machine Learning Algorythms for Analysis and Data Prediction* (2017) donde define Random Forest como un modelo preciso, estable y más sencillo de interpretar, pues representan relaciones no lineales para resolver problemas.

- Leo Breiman define un bosque aleatorio en su publicación *Random Forest* (2001) como un clasificador que consta de una colección de árboles estructurados clasificadores:

$$\{ h(x, \Theta_k), \quad k = 1 \tag{4}$$

Donde $\{\Theta_k\}$ son vectores aleatorios distribuidos idénticamente de manera independiente y cada árbol emite un voto para la clase más popular en x.

2.4.2 Características.

Una de las principales ventajas de Random Forest es la precisión de sus resultados, pues con una base de datos presenta resultados bastante exactos, es un algoritmo que funciona muy bien para analizar big data generando tendencias y clasificaciones de los datos analizados, además de responder bien ante casos de datos perdidos sin alterar gravemente el resultado.

2.7.2 Regresión de Random Forest.

La regresión en Random Forest se aplica con árboles de predicción cuando hay una respuesta continua, generando una estructura de árbol que se recorre hasta un nodo terminal generando un promedio de las observaciones de entrenamiento que están en ese mismo nodo terminal. Como tal, una regresión usando el algoritmo Random Forest es una técnica de Ensemble Learning, donde se toman varias veces un mismo algoritmo y se genera un modelo más completo que el original.

En Random Forest se genera una predicción basada en arboles de decisión con distintos valores dentro de un rango, esto hace que, si un cambio se genera en un árbol, ese árbol es descartado únicamente y no todo el conjunto de árboles.

Para generar una regresión con Random Forest lo primero que se debe hacer es elegir un conjunto de datos para entrenar el modelo, en base a estos datos se crea el árbol de decisión, se debe establecer un número de árboles que se debe crear.

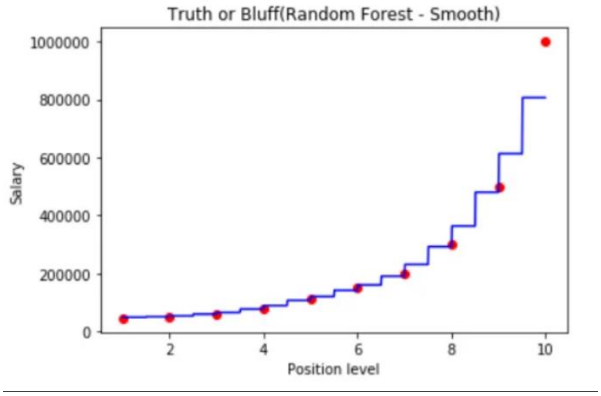

Figura 2. Ejemplo de regresión lineal del salario acorde al puesto del empleado.

2.7.3 Clasificación de Random Forest.

La clasificación se utiliza para determinar a qué conjunto es más probable que pertenezca un conjunto de datos. Según Breiman (1999) Los clasificadores de Random Forest consisten en una combinación de árboles que se encargan de clasificar cada valor. El clasificador se genera utilizando un vector aleatorio definido independientemente del valor de entrada, donde cada árbol emite un voto unitario para asignarlo a una clase, el voto más popular pasa a ser la clase definida.

El Clasificador de Random Forest utiliza el índice de Gini como medida de selección de cada atributo, se define como:

$$\sum\sum_{j\neq i}(f(C_i,T)/|T|)(f(C_j,T)/|T|) \quad (5)$$

Donde $f(C_i,T)/|T|$ es la probabilidad de el dato seleccionado pertenezca a C_i.

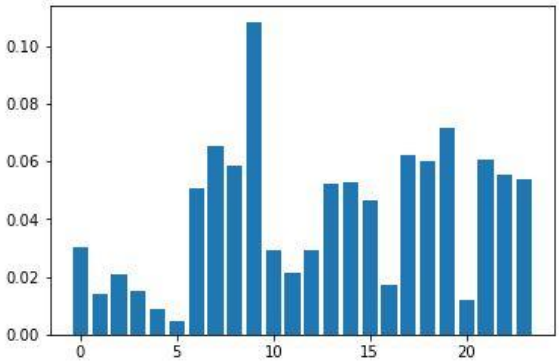

Figura 3. Ejemplo de clasificación usando Random Forest.

2.8 Prophet.

Prophet es un modelo de aprendizaje automático utilizado para pronósticos de series de tiempo basado en un modelo aditivo no lineal. Funciona mejor con series de tiempo estacionales y diferentes temporadas de datos históricos.

2.9 Métricas de evaluación.

Para saber si nuestro modelo funcionó adecuadamente es necesario tener como referencia datos cuantificables que desglosen el funcionamiento y especifique cuan preciso fue. Las métricas de regresión que se muestran a continuación, forman parte de conjunto ofrecido por la librería Scikit-learn. Estas reflejan información en específico del proceso de nuestro algoritmo, por lo que, es importante conocer el funcionamiento de cada métrica previamente para saber qué información nos es de utilidad al momento de evaluar el desempeño de nuestro algoritmo.

2.9.1. R2.

La función de puntuación R2 calcula el coeficiente de determinación. Este proporciona una medida de la bondad del ajuste de un modelo a la variable que pretende explicar. Se realiza de la siguiente manera.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n_{muestras}-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n_{muestras}-1}(y_i - \bar{y})^2} \quad (6)$$

Donde:

$\hat{y}_i$ es la variable predictor de la i-ésima muestra.

y_i es el valor verdadero correspondiente.

$\bar{y}$ es igual a $\frac{1}{n_{muestras}}\sum_{i=0}^{n_{samples}-1} y_i$

En la puntuación R cuadrado cuanto más cerca de 1 se encuentre, mejor será el ajuste del modelo a la variable q se quiere predecir.

2.9.2. Varianza explicada.

La varianza explicada calcula la diferencia entre la varianza de destino y la varianza de error de predicción. Es dada por:

$$VE(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}} \quad (7)$$

Donde:

$\hat{y}$ es la salida estimada.

y es la salida correcta.

En el caso de la puntuación de varianza explicada, el mejor resultado posible es 1.0.

2.9.3. Error absoluto medio.

Es una métrica que calcula la función de riesgo correspondiente al valor esperado de perdida de error absoluto. Se calcula:

$$MAE(Y; \hat{y}) = \frac{1}{n_{muestras}} \sum_{i=0}^{n_{muestras}-} |y_i - \hat{y}_i| \quad (8)$$

Donde:

$\hat{y}_i$ es la variable predicho de la i-ésima muestra.

y_i es el valor verdadero correspondiente.

El mejor resultado posible es 1.0.

2.9.4. Error absoluto porcentual.

El MAPE (por sus siglas en inglés) mide la precisión de un sistema. Esta se mide como un porcentaje y se puede calcular como:

$$MAPE(y, \hat{y}) = \frac{1}{n_{muestras}} \sum_{i=0}^{n_{muestras}-} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (9)$$

Donde:

$\hat{y}_i$ es la variable predicho de la i-ésima muestra.

y_i es el valor verdadero correspondiente.

Esta medida es la más comúnmente utilizada para pronosticar error, al ser medida mediante un porcentaje es la más simple de interpretar.

2.9.5. Error cuadrático medio.

El error cuadrático medio es una función de riesgo que corresponde al valor esperado de perdida por error cuadrático. Se calcula:

$$MSE(y, \hat{y}) = \frac{1}{n_{muestras}} \sum_{i=0}^{n_{muestras}} (y_i - \hat{y}_i)^2 \quad (10)$$

Donde:

$\hat{y}_i$ es la variable predicho de la i-ésima muestra.

y_i es el valor verdadero correspondiente.

El mejor resultado posible es 1.0.

CAPITULO III. METODOLOGIA DE SOLUCIÓN.

3.1 METODOLOGIA UTILIZADA.

Con el fin de optimizar el tiempo buscando mejores resultados se planeó el desarrollo de la investigacion bajo el siguiente proceso:

3.1.1 Comprension del problema.

Para desarrollar correctamente la investigacion es necesario comprender el problema a abordar, lo principal es generar preguntas que sirvan para la investigaicon, con el fin de absorber conocimiento, recopilar información, consultar fuentes y contrastar la informacion obtenida con el fin de determinar la relevancia de nuestros datos.

3.1.2 Criterios a tomar en cuenta.

Es importante establecer una serie de criterios para definir el modelo que se busca realizar, tomando en cuenta la precisión y la calidad de la información que se busca generar con el modelo, por lo que se deben desarrollar una serie de reglas para definir nuestro modelo.

3.1.3 Percepción de la actualidad.

Al ser el Machine Learning un tema de actualidad, constantemente se desarrollan soluciones desde diferentes perspectivas, debe tomarse en cuenta en todo momento el estado actual de desarrollo de nuevas soluciones para poder establecer si nuestra solucion ofrece un aporte mas simple, accesible o superior a las actuales soluciones disponibles.

3.1.4 Adecuar la información.

Es importante tomar en cuenta que se busca que los datos utilizados sirvan como muestra de datos reales utilizados en procesos de manufactura por lo que los datos deben ser cuidadosamente recopilados, considerando recopilaciones de datos completas, pues, siendo Random Forest un algoritmo que contrasta grandes cantidades de datos, cuanto mas robusta sea nuestra base de datos, mayor será la precisión con la que el algoritmo funcionará.

Es importante, sobre todo cuando se recopilan datos de distintas fuentes, el unificar los datos en un solo formato, de este modo un solo proceso puede analizar todos los datos mediante librerias enfocadas en Machine Learning tomando en cuenta la misma escala en todos los datos que se analizarán.

3.1.5 Modelo.

Se busca trabajar con el algoritmo de Machine Learning de Bosques Aleatorios, por lo que es importante construir un modelo basado en este algoritmo que permita generar resultados aprendiendo de los datos recopilados previamente. Debe tomarse posibles errores que surjan durante el desarrollo de nuestro modelo, comprender sus limitantes y asimilar las necesidades que nuestro modelo presenta para ser capaces de mejorar y corregir para generalizar el proceso de nuestro modelo..

3.1.6 Integracion.

Una vez el modelo este en un estado aceptable de precisión es importante comprobar si es factible implementarlo dentro de un proceso pues debe poder repetir el proceso cada que sea ejecutado, debe recibir informacion del preceso donde se integrará, asi como avisar de errores que se generen durante el proceso y poder ser capaz de corregirlos posteriormente.

3.2 Desarrollo de los experimentos.

Para la fase de experimentación aplicamos el modelo de bosques aleatorios usando el lenguaje de programación Python, apoyándonos de las librerías Numpy, Matplotlib, Pandas, Seaborn y Scikit-learn (Ver Código en Anexo 1).

Con la librería de Numpy mandamos llamar el csv de nuestro set de datos, la librería de Matplotlib será la librería que utilizamos para generar los gráficos y resultados de nuestros experimentos, pandas será utilizada para elaborar DataFrames para ordenar nuestra información y poderla analizar y la información, así como Seaborn para realizar el conteo de distribución de la información. Adicionalmente utilizaremos la librería de Scikit-learn que será utilizada para entrenar nuestro modelo y realizar los algoritmos de bosques aleatorios.

Después creamos un DataFrame en el que indicaremos la ruta del archivo separado por comas (CSV) en donde se encuentra nuestro set de datos (Ver Código en Anexo 2).

Para conocer mejor nuestro DataFrame utilizaremos la función describe que nos arrojará estadísticas descriptivas de nuestro set de datos. Para obtener un resumen conciso del marco de datos, resulta útil utilizar la función info. Esta función es utilizada para hacer un análisis exploratorio de los datos. La función dtypes nos permitirá conocer el tipo de dato de nuestras columnas. Por último, la función se utiliza para mostrar el tamaño del DataFrame de Pandas. Devuelve el tamaño del DataFrame que es equivalente al número total de elementos (Ver código en Anexo 3).

Adicionalmente, utilizaremos una función que nos que nos mostrará la distribución de los datos a través del tiempo (Ver código en Anexo 4). Ya que analizamos nuestra información, definimos los predictores y etiquetas de nuestra regresión. Asignaremos los valores de *x* para nuestros predictores, nuestras etiquetas serán las correspondientes a la variable *y* (Ver código en Anexo 5).

Y con los datos seleccionados comenzamos el entrenamiento del modelo. Definimos nuestros datos de prueba de los de entrenamiento, en nuestro caso, separamos el 80 de los datos para prueba dejando el 20% restante para entrenar nuestro modelo (Ver código en Anexo 6).

Dependiendo del modelo, es necesario normalizar los datos, en el caso de Random Forest esto no es necesario, pero si deseamos realizarla, la librería Scikit-learn nos permite realizarla (Ver código en Anexo 7). Con el modelo entrenado, definidos los datos de prueba y de entrenamiento, elaboramos la regresión del método de bosques Aleatorios. Esta se realiza automáticamente con ayuda de la misma librería Scikit-learn, por lo que el proceso se hace de manera automática. Únicamente es necesario definir el número de estimaciones, en este caso 0, pues generan una cantidad de datos aceptable, sin saturar el equipo al momento de ejecutar el código (Ver código en Anexo 8).

3.3 Metodología de evaluación.

3.3.1. Evaluación teórica.

Se comparó el algoritmo de Machine Learning con el algoritmo Prophet. Tomando en cuenta el funcionamiento, sus bases, áreas donde se desempeña, ventajas y carencias de ambos modelos.

El funcionamiento del modelo se centró principalmente en evaluar el contexto en el que se desarrolló el algoritmo, diseño del modelo, vigencia y utilidad de cada modelo. Las bases evalúan cuan extenso es tomando como referencia el tiempo que lleva funcionando y fundamentos teóricos.

Las áreas donde se desempeña cada algoritmo toman en cuenta en que proceso puede ser ventajoso usar dicho modelo. También se compararon las principales ventajas y carencias significativas de cada modelo, todo con el fin de evaluar justamente el desempeño de cada algoritmo.

3.3.1. Evaluación práctica.

Se compararon 2 experimentos en los que Random Forest y Prophet cuentan con suficiente variedad de datos para realizar sus respectivos procesos en donde se buscó evaluar la eficiencia, eficacia, así como el desempeño del algoritmo en diferentes condiciones, además de evaluar la precisión y la calidad de resultados que el algoritmo ofrece en diferentes situaciones en procesos de manufactura.

El desarrollo de los experimentos se realizó con dos sets de datos distintos en donde se evaluaron con métricas de regresión (pues en ambos casos los algoritmos realizaron una regresión) donde se busca evaluar la precisión, perdida de datos, desempeño, entre otras, para comparar que tan eficiente y eficaz fue el desempeño a lo largo de los experimentos.

CAPITULO IV. DESARROLLO.

4.1. Desarrollo de los experimentos.

Se elaboraron 3 experimentos utilizando el algoritmo de bosques aleatorios con diferentes sets de datos de diferentes procesos de manufactura obteniendo los siguientes resultados.

4.1.1. EXPERIMENTO 1.

Para el primer experimento nos apoyaremos de un set de datos de más de 4 mil datos de sensores de Temperatura, CO2, Presión y Aire.

Tabla 1. Resultados de estadísticas del DataFrame

	Temperatura	**CO2**	**Presión**	**Aire**
Count	4378.000000	4378.000000	4378.000000	4378.000000
Mean	42.066743	2687.652810	1862.664002	7296.235724
Std	3.012033	204.134289	113.466682	427.943217
Min	0.000000	0.000000	0.000000	0.000000
25%	41.200000	2697.000000	1851.000000	7318.000000
50%	42.300000	2706.000000	1877.000000	7321.000000
75%	43.500000	2710.000000	1887.000000	7326.000000
max	51.400000	2737.000000	2021.000000	7684.000000

La tabla obtenida con la función describe de la librería Pandas muestra información estadística de nuestro DataFrame, nos detalla el número total de cada una de nuestras columnas (Count), el promedio (Mean), la dispersión de los datos o desviación estándar (Std), el valor mínimo y máximo de nuestro DataFrame, y los valores correspondientes a los percentiles que dividen nuestro conjunto de datos ordenados en cien partes iguales, mostrando en la tabla los resultados de los percentiles de 25%, 50% y 75%.

Procedemos a realizar el análisis exploratorio de los datos.

Tabla 2. Resultados del análisis exploratorio del DataFrame.

#	**Column**	**Non-Null Count**	**Dtype**
0	Fecha/Hora	4378 non-null	Object
1	Temperatura	4378 non-null	Float64
2	CO2	4378 non-null	Int64
3	Presión	4378 non-null	Int64
4	Aire	4378 non-null	Int64

Ejecutamos la función para ver el tipo de datos de nuestro dataset.

Tabla 3. Tipo de datos del DataFrame.

Fecha/Hora	**Object**
Temperatura	Float64
CO2	Int64
Presión	Int64
Aire	Int64
dtype	object

Se debe tener en cuenta que la mayoría de las columnas en nuestros datos de encuesta son del tipo int64. Esto significa que son enteros de 64 bits. Pero la columna Temperatura es un valor de punto flotante o float, lo que significa que contiene decimales. Usamos la función para ver el tamaño de nuestro DataFrame, en este caso 6760.

Para que el set de datos no se vea alterado vamos a eliminar los datos que afectan a nuestro documento separado por comas, en este caso se eliminarán aquellos valores que sean iguales a cero, esto porque al momento de verificar el registro de los datos se agrega un valor en 0 al documento. También se eliminarán unos picos que producen problemas en la predicción. Eliminando estos datos se evitará que la gráfica se vea alterada con valores exageradamente diferentes a los reales (Ver anexo 9).

Haciendo la gráfica de distribución podemos ver que hay uniformidad en los datos puesto que no hay valores repetidos en una misma fecha, siendo que todos tienen la misma distribución de información.

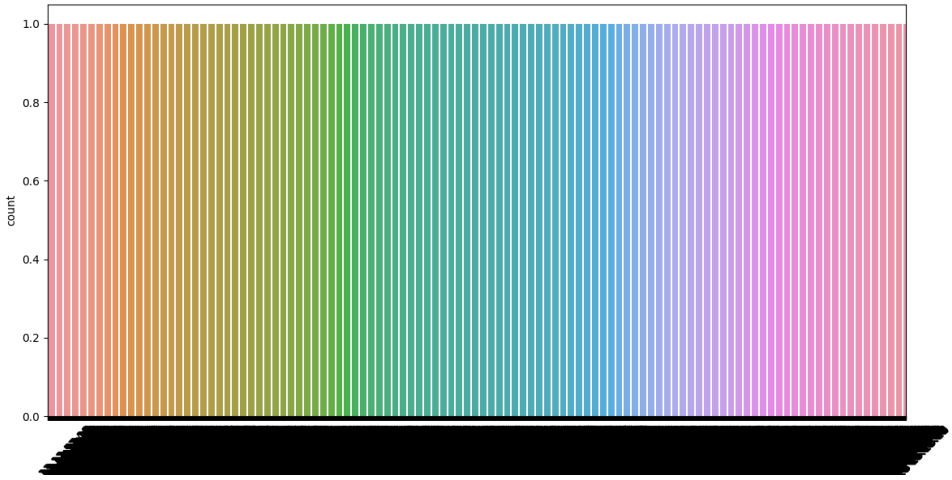

Figura 4. Distribución de datos.

En nuestro caso queremos saber cómo afecta el CO2 a la temperatura. Del mismo modo haremos un segundo grafico visualizando como afecta la presión al diferencial de Aire.

Procedemos con el entrenamiento de los datos y realizamos la regresión. El resultado de la primera regresión fue el siguiente.

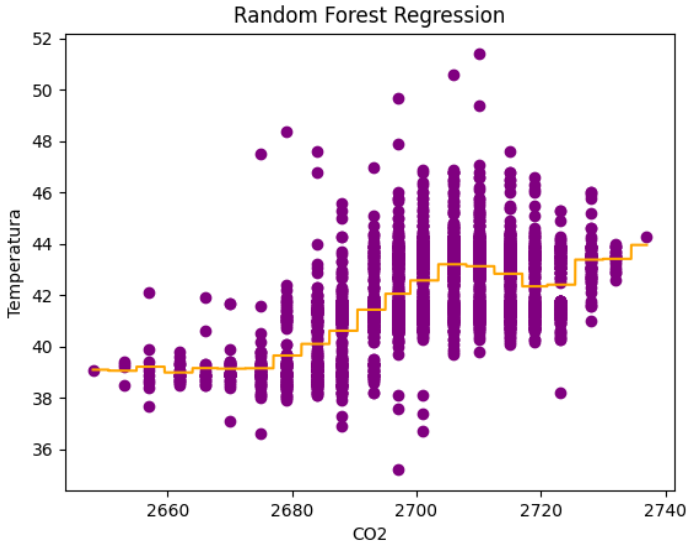

Figura 5. Regresión con el modelo de Bosques Aleatorios.

Podemos observar que la tendencia en el modelo es un incremento escalonado en la temperatura a medida que el dióxido de carbono va aumentado.

Para evaluar el desempeño, mediante la puntuación R2 (cuya mejor puntuación posible es 1.0 y puede ser negativa) que representa la proporción de varianza (de y) que ha sido explicada por las variables independientes en el modelo. Proporciona una indicación de la bondad del ajuste y, por lo tanto, una medida de qué tan bien es probable que el modelo prediga muestras invisibles, a través de la proporción de varianza explicada (su mejor puntuación es 1.0), que se usa para medir la discrepancia entre un modelo y los datos reales. El error absoluto medio (mejor puntuación es 0.0) es una métrica de riesgo que mide al valor esperado de la pérdida de error absoluto, el error absoluto porcentual medio (su mejor puntuación es 0.0 y no debe ser negativa), es una métrica de evaluación para problemas de regresión. La idea de esta métrica es ser sensible a los errores relativos y el error cuadrático medio (mejor puntuación 0.0) que mide el promedio de los errores al cuadrado.

Tabla 4. Resultados del desempeño del modelo.

Puntuación R2	0.47718100678135955
Puntuación de variación explicada	0.4777390128156791
Error absoluto medio	0.9983029222082913
Error absoluto porcentual	0.023750424240294412
Error cuadrático medio	1.6335070046859357

Considerando la cantidad de datos utilizados es un valor aceptable puesto que, en el resultado de la puntuación r2 es cercano a 1.0, mismo caso que la puntuación de varianza explicada, la diferencia entre el resultado del desempeño del algoritmo es menos de una unidad en ambos casos. Para el caso del error absoluto medio, el caso es similar, aunque el resultado es casi de 1, siendo el mejor caso 0.0, podemos ver que igualmente la diferencia es de menos de una unidad.

En el caso del error absoluto medio porcentual podemos ver quizá el escenario donde mejor se desempeña Random Forest, pues vemos que el resultado es de un 2.37% del tamaño del error (absoluto) lo cual, considerando la cantidad de datos utilizados mantiene más del 97% de desempeño sin errores. Por ultimo en el caso del error cuadrático medio tiene un desempeño de 1.6 vemos que el modelo se aleja en promedio 1.6 unidades de los valores reales, no se considera una gran diferencia entre los datos, puesto que es un valor cercano a los datos reales. Para la siguiente comparativa definimos las columnas, en este caso, será presión y aire, por los motivos antes mencionados. Repetimos el proceso de entrenamiento, con los nuevos valores, nuevamente usaremos el 15% de los datos como prueba y el 85% restante de entrenamiento. Con los datos entrenados, realizamos la regresión con el modelo de bosques aleatorios.

El resultado de la segunda regresión fue el siguiente.

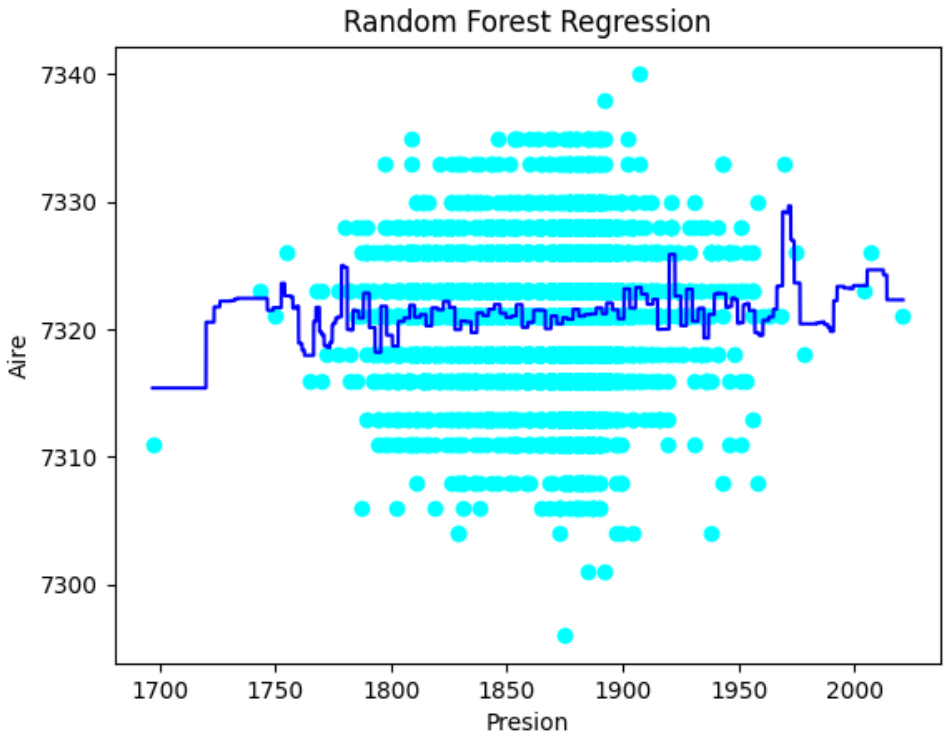

Figura 6. Regresión con el modelo de Bosques Aleatorios.

A diferencia del primer comparativa, podemos observar un desempeño bastante más irregular por lo dispersos que están los datos, esto se ve reflejado en el desempeño las métricas de error. Evaluamos el desempeño obteniendo el siguiente resultado.

Tabla 5. Resultados del desempeño del modelo.

Puntuación R2	-0.009241112626214631
Puntuación de variación explicada	-0.00902588985258057
Error absoluto medio	4.2935919856827285
Error absoluto porcentual	0.0005865234890956334
Error cuadrático medio	5.625926714788197

En este caso podemos observar que tanto la puntuación R2 como la varianza explicada mantienen aún mayor cercanía con el mejor resultado (1.0), teniendo un buen desempeño, caso contrario al error absoluto medio en donde se aleja en más de 4 unidades del mejor resultado posible, en el caso del error absoluto medio porcentual vemos un gran desempeño pues hay un 0.9% de error, lo cual se considera un porcentaje bajo, en el caso del error cuadrático medio es comprensible que su desempeño no sea el mejor por lo dispersos que son los datos, viéndose reflejado con ese resultado de 6.3 puntos.

4.1.2. EXPERIMENTO 2.

Para este segundo experimento nos apoyaremos de un set de datos de 2 mil 100 datos de sensores de Temperatura, en un horno de fundición. En este experimento se busca utilizar el modelo de bosques aleatorios en series de tiempo para evaluar su desempeño. Después crearemos un DataFrame en el que indicaremos la ruta del archivo separado por comas (CSV) en donde se encuentra nuestro set de datos. Para conocer mejor nuestro DataFrame utilizaremos la función que nos arrojará estadísticas descriptivas de nuestro set de datos.

Tabla 6. Resultados de estadísticas del DataFrame.

	Temperatura
Count	1895.000000
Mean	616.282322
Std	95.796748
Min	176.000000
25%	578.000000
50%	631.000000
75%	694.000000
max	714.000000

Esta información incluye el número de muestras, el valor medio, la desviación estándar, el valor mínimo, máximo, la mediana y los valores correspondientes a los percentiles 25% y 75%.

Para obtener un resumen conciso del marco de datos usamos la función que es utilizada para hacer un análisis exploratorio de los datos.

Tabla 7. Información del DataFrame.

#	column	Non-null count	dtype
0	Fecha/Hora	1895 non-null	Object
1	Temperatura	1895 non-null	Int64

Se debe tener en cuenta que la temperatura de nuestro set de datos es de tipo int64. Esto significa que son enteros de 64 bits. Pero la columna Fecha es un valor de punto de objeto, lo que significa que contiene decimales.

Para que el set de datos no se vea alterado vamos a eliminar los datos que afectan a nuestro documento separado por comas, en este caso se eliminarán aquellos valores que sean iguales a cero, esto porque al momento de verificar el registro de los datos se agrega un valor en 0 al documento. Eliminando estos datos se evitará que la gráfica se vea alterada con valores exageradamente diferentes a los reales (Ver código en Anexo 10).

Ejecutamos el siguiente código que nos mostrará la distribución de los datos a través del tiempo. Podemos ver que hay uniformidad en los datos dando prácticamente una distribución mayormente uniforme.

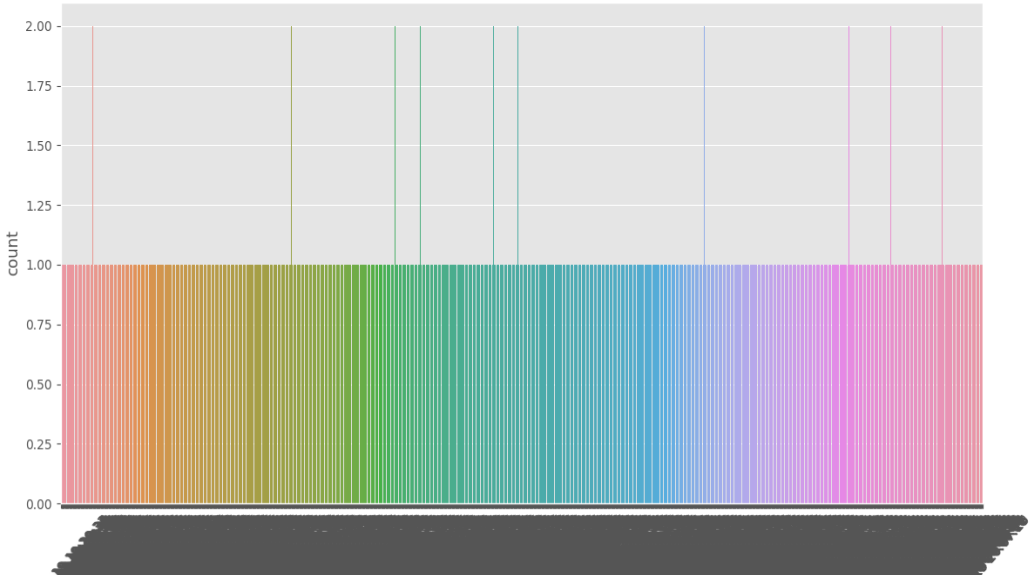

Figura 7. Distribución de datos.

Ya que analizamos nuestra información, definimos los predictores y etiquetas de nuestra regresión. En este segundo experimento solo analizaremos la temperatura, usando como predictor la fecha y la hora de nuestro dataset, Si bien Random Forest no es un algoritmo de series de tiempo, este puede realizar mediante el procesamiento de datos, resultados similares con regresión. Es importante para utilizar Random Forest en series de tiempo procesar los datos previamente para adaptarlo a una serie de tiempo. Primero extraeremos información de la columna Fecha/Hora y generaremos las columnas dteday y hora en nuestro DataFrame (Ver código en Anexo 11).

Generaremos un gráfico de la información de nuestros datos, para ver el desarrollo de la temperatura en el horno durante el proceso de fundición.

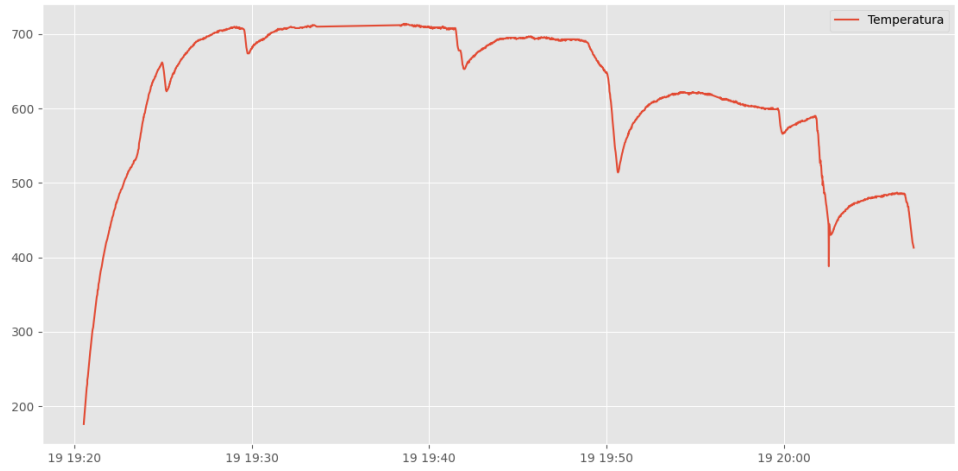

Figura 8. Temperatura del horno de fundición durante el proceso.

Podemos observar el incremento inicial al encender el horno, la estabilización de la temperatura durante la fundición y un descenso escalonado para reducir la temperatura al final del proceso.

Para poder usar bosques aleatorios en series de tiempo trataremos los datos de la siguiente manera (Ver código en Anexo 12).

Desplazamos los datos para reestructurar el dataset y de ese modo poder usar el paso de tiempo como variable para predecir el futuro. En este punto empezaremos con el proceso de Regresión usando Bosques Aleatorios. Para poder realizar el entrenamiento hay que definir datos serán predictores y cuales etiquetas, en este caso la etiqueta será la columna de temperatura, mientras que el predictor será la fecha, el proceso es el siguiente (Ver código en Anexo 13).

Debemos crear unos array con las diferentes columnas de temperaturas desplazadas y concatenarlos en la x final para poder usarla como reemplazo de la fecha, pues como tal la fecha no es utilizable para el modelo de Random Forest.

Ya con nuestros predictores y etiquetas definidas, asignamos nuestros datos de prueba de los de entrenamiento, en nuestro caso, como son pocos los datos, separamos solo el 30% de los datos para prueba dejando el 70% restante para entrenar nuestro modelo. Con el modelo entrenado, elaboramos la regresión del método de bosques Aleatorios. Con esto, podremos ver más claramente la comparativa entre los valores actuales comparados con la predicción de Random Forest.

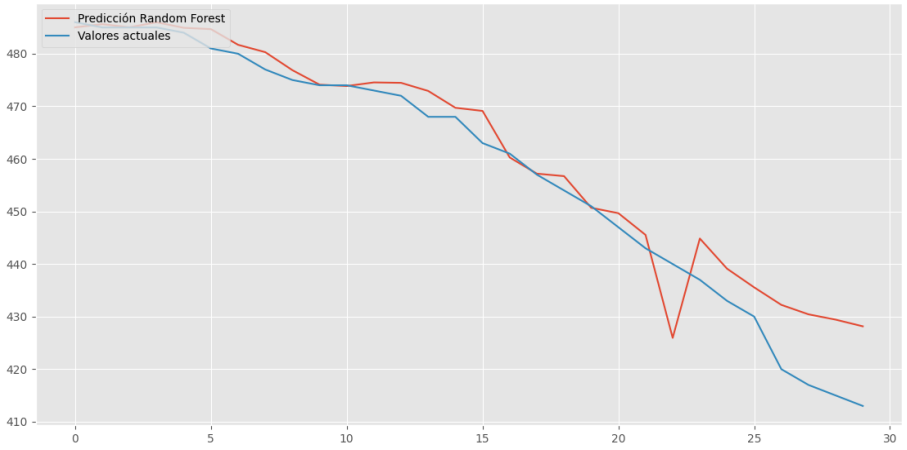

Figura 9. Regresión con el modelo de Bosques Aleatorios.

Podemos observar como la tendencia indica un descenso de datos en donde la predicción mantiene bastante cercanía a hasta un punto posterior a 20 donde disminuye drásticamente, se estabiliza y posteriormente intenta corregir. Utilizando la función para evaluar el desempeño obtenemos el siguiente resultado. Mediante la puntuación R2 (cuya mejor puntuación posible es 1.0 y puede ser negativa) que representa la proporción de varianza (de y) que ha sido explicada por las variables independientes en el modelo.

Proporciona una indicación de la bondad del ajuste y, por lo tanto, una medida de qué tan bien es probable que el modelo prediga muestras invisibles, a través de la proporción de varianza explicada (su mejor puntuación es 1.0), que se usa para medir la discrepancia entre un modelo y los datos reales.

En otras palabras, es la parte de la varianza total del modelo que se explica por factores que realmente están presentes y no se debe a la varianza del erro, el error absoluto medio (mejor puntuación es 0.0) que es una métrica de riesgo correspondiente al valor esperado de la pérdida de error absoluto, el error absoluto porcentual medio (su mejor puntuación es 0.0 y no debe ser negativa), es una métrica de evaluación para problemas de regresión.

La idea de esta métrica es ser sensible a los errores relativos y el error cuadrático medio (mejor puntuación 0.0) que mide el promedio de los errores al cuadrado, es decir, la diferencia entre el estimador y lo que se estima. El ECM es una función de riesgo, correspondiente al valor esperado de la pérdida del error al cuadrado o pérdida cuadrática.

Tabla 8. Resultados de desempeño del algoritmo.

Puntuación R2	0.9238136015837581
Puntuación de variación explicada	0.9432163561750544
Error absoluto medio	4. 4.303114682539697
Error absoluto porcentual	0.009863143513462238
Error cuadrático medio	6.38629626721255

Considerando la cantidad de datos utilizados es un valor aceptable puesto que, en el resultado de la puntuación r2 es cercano bastante a 1.0, mismo caso que la puntuación de varianza explicada, la diferencia entre el resultado del desempeño

del algoritmo es menos de una unidad en ambos casos. Para el caso del error absoluto medio, el caso es algo distinto, pues el modelo se alejó en hasta 4 unidades del mejor valor posible. En el caso del error absoluto medio porcentual podemos ver quizá el escenario donde mejor se desempeña Random Forest, pues vemos que el resultado es de un 0.98% del tamaño del error (absoluto) lo cual, considerando la cantidad de datos utilizados mantiene un porcentaje de casi 100% de desempeño sin errores. Por ultimo en el caso del error cuadrático medio tiene un desempeño de 6.3 vemos que el modelo se aleja de los valores reales, no se considera una gran diferencia entre los datos, puesto que es un valor cercano a los datos reales considerando que Random Forest no es un algoritmo utilizado para series de tiempo.

4.1.3. EXPERIMENTO 3.

Para el tercer experimento utilizaremos un set de datos de más de 10 mil datos obtenidos de un csv de máquinas donde se analizará la temperatura y revoluciones por minuto de las máquinas. Para este experimento contamos con distintas variables de las cuales las principales para este experimento serán la temperatura del aire (en grados Kelvin), la temperatura del proceso (en grados Kelvin) el Torque (esfuerzo de torsión), usamos la función que nos dará información de nuestros datos, incluye el número de muestras, el valor medio, la desviación estándar, el valor mínimo, máximo, la mediana y los valores correspondientes a los percentiles 25% y 75%.

Tabla 9. Resultados descriptivos del DataFrame.

	…	**Air temperature [K]**	**Process temperature [K]**	**Rotational speed [rpm]**	**Torque [Nm]**	**…**
Count	…	10000.00000	10000.000000	10000.000000	10000.000000	…
Mean	…	300.004938	310.005560	1538.776100	39.986910	…
Std	…	2.000259	1.483734	179.284096	9.968934	…
Min	…	295.300000	305.700000	1168.000000	3.800000	…
25%	…	298.300000	308.800000	1423.000000	33.200000	…
50%	…	300.100000	310.100000	1503.000000	40.100000	…
75%	…	301.500000	311.100000	1612.000000	46.800000	…
max	…	304.500000	313.800000	2886.000000	76.600000	…

Para obtener un resumen conciso del marco de datos, resulta útil utilizar la función de análisis exploratorio. Podemos observar con mayor detalle datos estadísticos de nuestro DataFrame.

Tabla 10. Información del DataFrame.

#	**column**	**Non-null count**	**dtype**
0	UDI	10000 non-null	Int64
1	Product ID	10000 non-null	Object
2	Type	10000 non-null	Int64
3	Air temperature [K]	10000 non-null	Float64
4	Process temperature [K]	10000 non-null	Float64
5	Rotational speed [rpm]	10000 non-null	Int64
6	Torque [Nm]	10000 non-null	Float64
7	Tool wear [min]	10000 non-null	Int64
8	Machine failure	10000 non-null	Int64
9	TWF	10000 non-null	Int64
10	HDF	10000 non-null	Int64
11	PWF	10000 non-null	Int64
12	OSF	10000 non-null	Int64
13	ENF	10000 non-null	Int64

Debemos observar que la mayoría de columnas son de tipo int64. Esto significa que son enteros de 64 bits. Pero las columnas Air Temperature [K], Process Temperature [K] y Torque [Nm] son de tipo float 64, esto significa que contiene decimales.

En este dataset los datos ya vienen previamente procesados, eliminando valores nulos o ajenos al proceso, por lo que en este caso en específico no modificaremos los datos, pues ya están correctamente procesados.

Ejecutamos el código que nos mostrará la distribución de los datos a través del tiempo. Podemos ver que hay uniformidad en los datos puesto que no hay valores repetidos en una misma fecha, siendo que todos tienen la misma distribución de información.

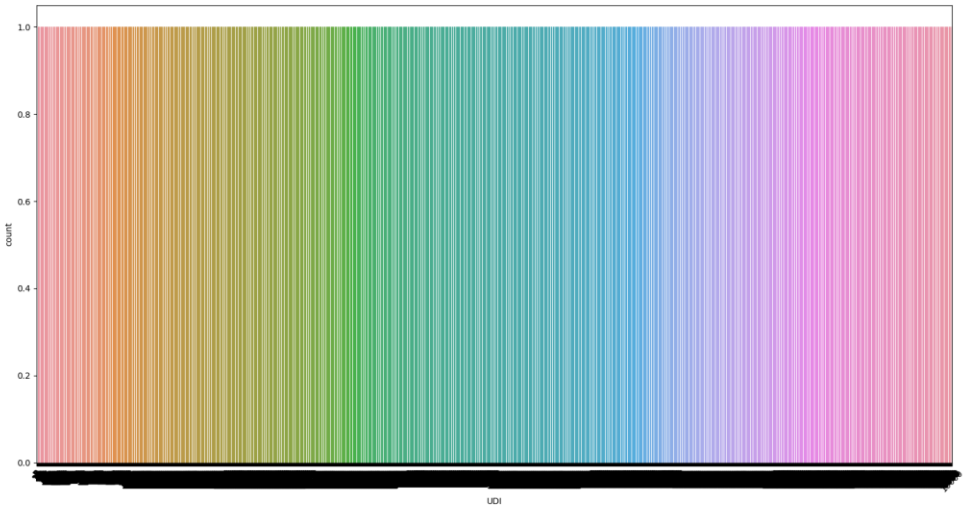

Figura 10. Grafica de distribución de los datos.

Primero que nada, se muestra el código que muestra una regresión lineal realizada donde se muestran los datos de la velocidad de rotación, el torque y la comparativa entre máquinas que fallaron y máquinas que no (Ver código en anexo 14).

Nos mostrará el siguiente gráfico.

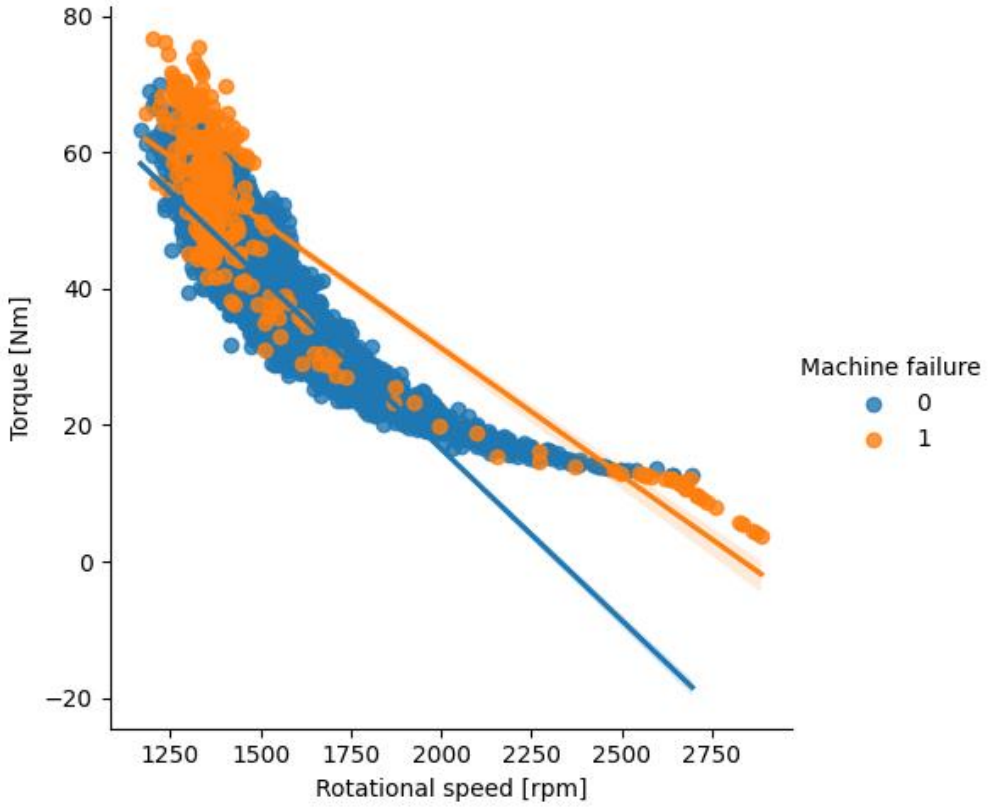

Figura 11. Regresión lineal.

Definimos nuestros predictores y nuestras etiquetas con los datos que nos son de interés, en este caso serán dos, el primero Las revoluciones por minuto, así como la fuerza de rotación y para el segundo la temperatura tanto del proceso, como la del aire.

Entrenamos el modelo y realizando la regresión nos mostrará la gráfica hecha con Random Forest.

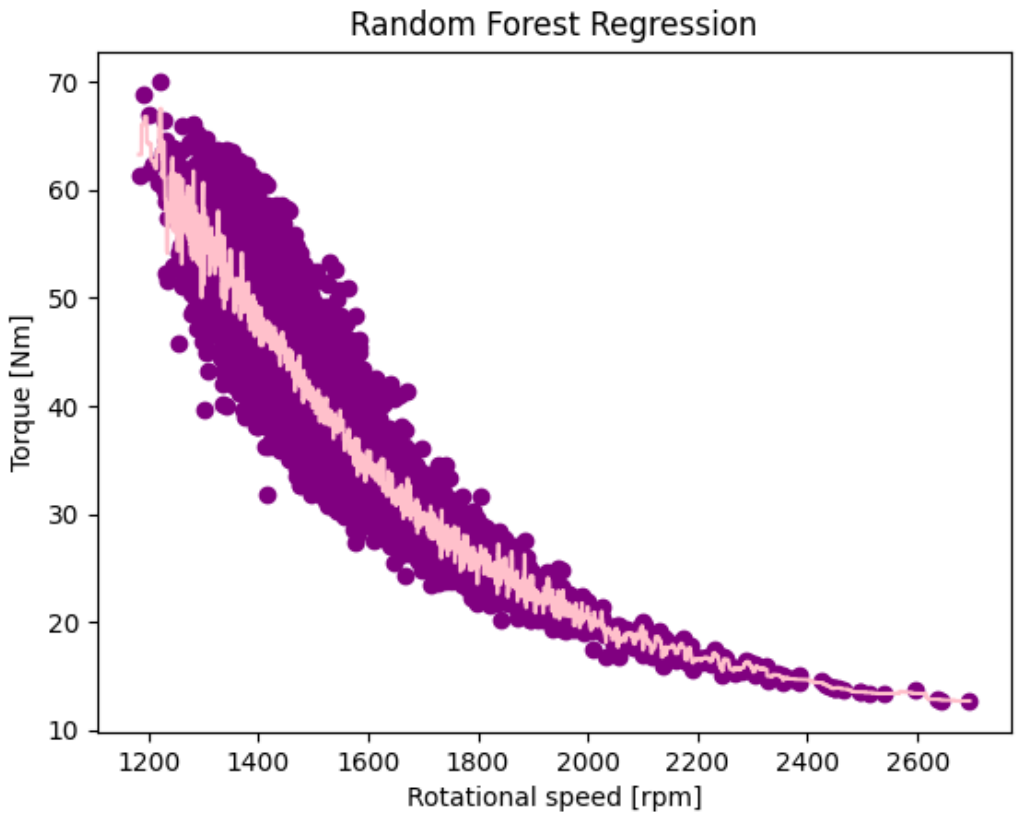

Figura 12. Regresión utilizando Random Forest.

Podemos ver un descenso progresivo conforme avanza el modelo. Para evaluar la precisión del método utilizamos distintos métodos. Utilizando esta función obtenemos el siguiente resultado. Mediante la puntuación R2 (cuya mejor puntuación posible es 1.0 y puede ser negativa) que representa la proporción de varianza (de y) que ha sido explicada por las variables independientes en el modelo. Proporciona una indicación de la bondad del ajuste y, por lo tanto, una medida de qué tan bien es probable que el modelo prediga muestras invisibles, a través de la

proporción de varianza explicada (su mejor puntuación es 1.0), que se usa para medir la discrepancia entre un modelo y los datos reales. En otras palabras, es la parte de la varianza total del modelo que se explica por factores que realmente están presentes y no se debe a la varianza del erro, el error absoluto medio (mejor puntuación es 0.0) que es una métrica de riesgo correspondiente al valor esperado de la pérdida de error absoluto, el error absoluto porcentual medio (su mejor puntuación es 0.0 y no debe ser negativa), es una métrica de evaluación para problemas de regresión. La idea de esta métrica es ser sensible a los errores relativos y el error cuadrático medio (mejor puntuación 0.0) que mide el promedio de los errores al cuadrado, es decir, la diferencia entre el estimador y lo que se estima. El ECM es una función de riesgo, correspondiente al valor esperado de la pérdida del error al cuadrado o pérdida cuadrática.

Tabla 11. Resultados de desempeño del modelo.

Puntuación R2	0.844236023426191
Puntuación de varianza explicada	0.8444224294745373
Error absoluto medio	2.8921809109326646
Error absoluto porcentual	0.07136547470658752
Error cuadrático medio.	3.7668439949212433

Podemos observar que tanto en la puntuación R2 y la varianza explicada el desempeño fue bastante bueno, pues es un dato muy cercano a 1, que es la mejor puntuación posible. Vemos como el error absoluto medio refleja que hubo un 2.8 de valor absoluto que se espera perder, pero vemos que en el porcentual hubo un 0.7%

de diferencia entre los datos estimados y los reales, por último, el error cuadrático medio nos muestra como, por la cantidad de datos analizados se llega a separar hasta 3.7 décimas de los valores esperados de perdida.

Para la siguiente comparativa definimos las columnas, en este caso, la temperatura del aire y del proceso como mencionamos anteriormente. Se muestra una regresión lineal realizada donde se muestran los datos de la temperatura del aire y del proceso y la comparativa entre máquinas que fallaron y máquinas que no. Nos muestra la siguiente gráfica.

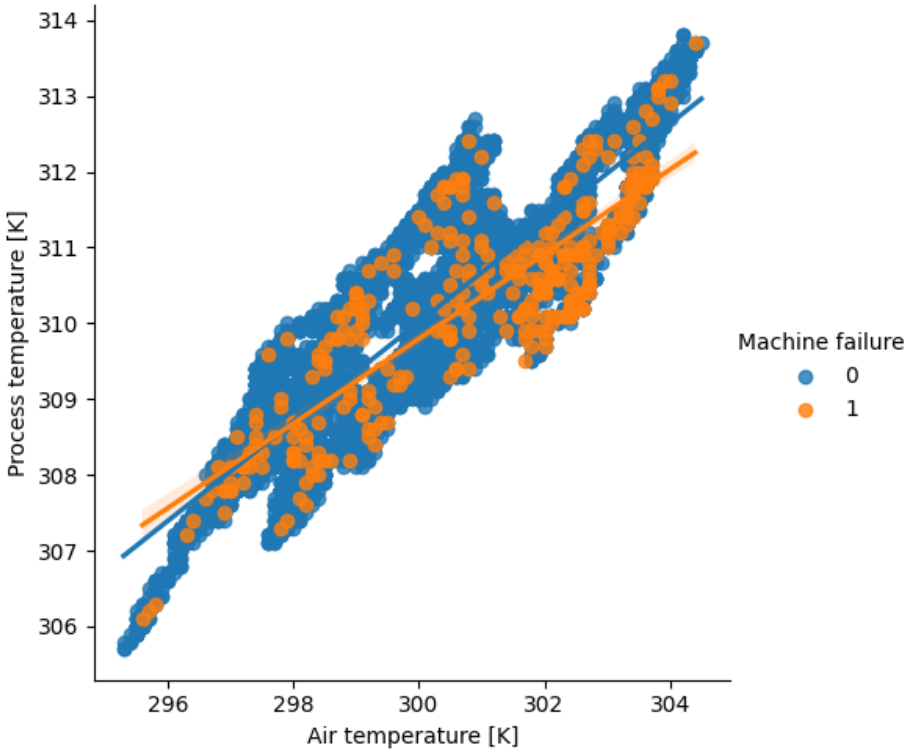

Figura 13. Regresión lineal.

Procedemos con el entrenamiento del modelo. Con el modelo entrenado, realizamos la regresión.

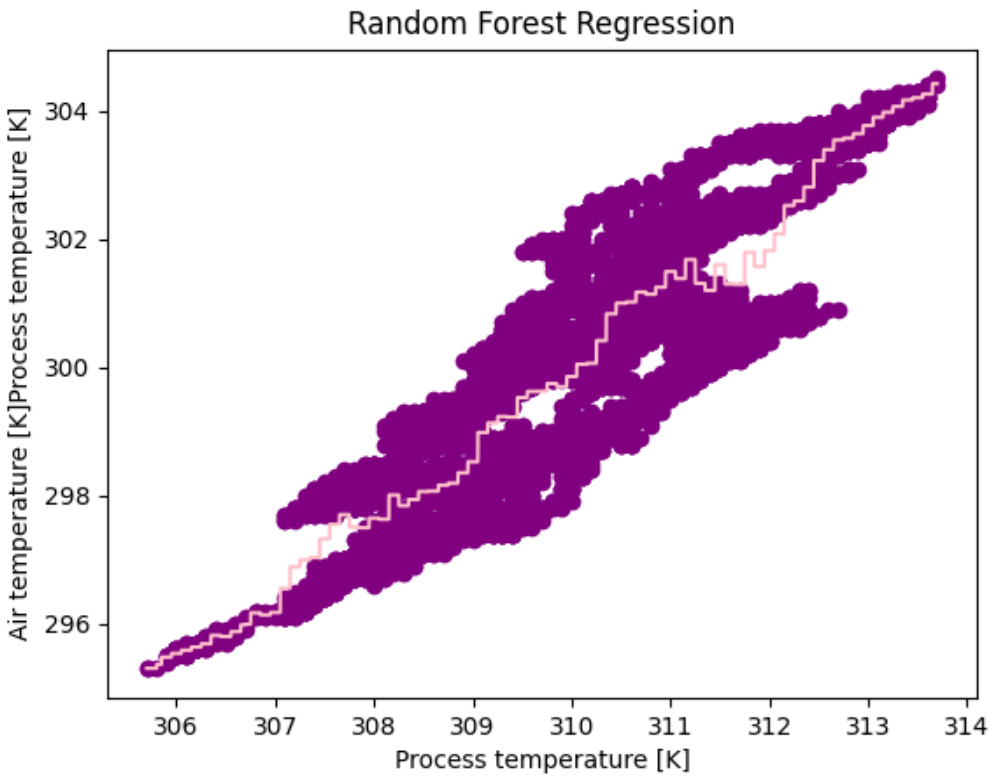

Figura 14. Regresión lineal usando el modelo de Random Forest.

Podemos observar como a lo largo del proceso la temperatura aumento en ambos valores de manera constante, esto se ve reflejado en el desempeño las métricas de error.

Tabla 12. Resultados de desempeño del modelo.

Puntuación R2	0.7824378701954381
Puntuación de varianza explicada	0.7824901398974619
Error absoluto medio	0.7786851155647859
Error absoluto porcentual	0.0025685565720078318
Error cuadrático medio.	0.9287719616119456

En este caso podemos observar que tanto la puntuación R2, como la varianza explicada se mantienen en valores cercanos a 1. El error absoluto nos refleja que hubo esperado sobre la perdida de error absoluto. El error absoluto porcentual es de 0.25 de perdida. Mientras que el error cuadrático medio tiene un valor de menos de una unidad de dispersión de los datos.

4.2. Desarrollo de evaluación.

4.2.1. Evaluación teórica.

En el caso de Random Forest, el modelo selecciona de manera aleatoria elementos para generar diferentes muestras los cuales promedia para reducir su variación, es un modelo que se beneficia de este método pues los arboles de decisión se benefician al promediar los datos, manteniendo una baja parcialidad. Para realizar una predicción cada árbol le asigna una etiqueta al nodo final de manera iterativa en todos los arboles dentro del modelo y la etiqueta que obtenga más veces resultados es la considerada como la predicción. Se utilizó principalmente en análisis de variables múltiples para realizar regresiones lineales y clasificaciones. También puede desempeñarse para realizar series de tiempo, pero no reconoce la columna con formato de tiempo, por lo que esta debe ser adaptada previamente.

Las principales ventajas que ofreció Random Forest fueron el funcionar muy bien conforme más grande fuera el set de datos, permitiendo hacer más completo cada experimento permite trabajar con múltiples tipos de variables además de ser uno de los mejores algoritmos evitando el sobreajuste. Por la necesidad de los experimentos se realizaron regresiones, pero el algoritmo funciona mejor para realizar clasificaciones

A diferencia de Random Forest, Prophet es un modelo para pronosticar datos de series temporales basado en un modelo no-lineal aditivo pues no se puede escribir de forma lineal, sino más bien a través de una función desconocida. El mayor problema con este tipo de modelos es que no contamos con una forma explícita para suavizar la regresión por lo que sebe utilizar ciertas funciones específicas según sea el caso lo que lo hace más complejo a la hora de realizar la predicción. Es un modelo que tiene muy en cuenta la estacionalidad usando series de Fourier y se emplea principalmente para el análisis de series de tiempo. Esto hace que él se tenga que contar con un orden en los datos en base al tiempo para poder visualizar la información que Prophet ofrece.

4.2.2. Evaluación práctica.

Para la evaluación práctica se compararon dos experimentos de los 3 realizados con ambos modelos de Machine Learning en el que se evaluó el desempeño de ambos algoritmos y se comparó utilizando las metricas de error cuadrático medio, error absoluto medio y error absoluto medio porcentual para evaluar que cuan acertados eran estos modelos.

El primer experimento nos muestra como a pesar de tener muchas similitudes, lo diferentes que pueden llegar a ser estos algoritmos.

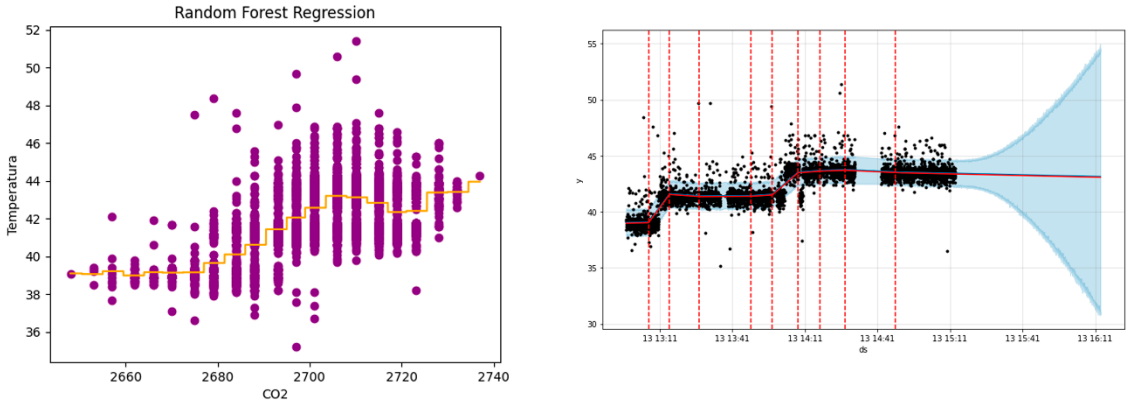

Figura 15. a) Lado izquierdo regresión con Random Forest usando las variables de CO2 (x) y Temperatura (y). b) Lado derecho regresión con Prophet usando las variables de Fecha/Hora (x) y Temperatura (y).

Podemos observar como primeramente el modelo de Random Forest permite el uso de cualquier variable como predictor (x), el resultado dependerá de la relación entre las variables y, por tanto, la variable predictor se define según el sentido del propio proceso de manufactura que se desea evaluar, por otro lado, Prophet, al estar enfocado a series de tiempo la variable predictor debe ser siempre el tiempo pues esta es la que nos interesa conocer para evaluar como varia el modelo a través del tiempo.

Si bien en ambos casos se realiza una regresión, en el modelo de Random Forest se toma en cuenta la tendencia de los datos ingresados, para saber qué ocurriría en que rango de valores, mientras que en Prophet se busca hacer una predicción a futuro de los datos, los cuales podemos observar en el espectro de color celeste en la imagen derecha, estos valores irán aumentando exponencialmente conforme el tiempo avance, pues la certeza a futuro será cada vez menor.

En este experimento las métricas de precisión obtenidas fueron las siguientes.

Tabla 13. Comparativa métricas RF y Prophet.

Métricas	**Regresión RF (CO2/TEMP)**	**Regresión Prophet (Tiempo/TEMP)**
MAPE	0.023750424240294412	0.013305
MAE	0.9983029222082913	0.583330
MSE	1.6335070046859357	0.606332

En este primer experimento las métricas obtenidas fueron dadas por los respectivos modelos en sus librerías, en el caso de Random Forest podemos observar mayor precisión al mostrar resultados que el algoritmo Prophet. Podemos observar en cuando al error absoluto medio porcentual (MAPE) ambos actuaron bastante bien pues en cuanto a porcentaje de perdida de regresión tuvieron un 2.3% y 1.3% respectivamente, teniendo ambos valores demasiado bajos de perdida, por lo que ambos se desempeñaron de manera correcta en este aspecto. Vemos en lo correspondiente a la métrica de error absoluto medio (MAE) podemos observar que el algoritmo de Random Forest tuvo un desempeño ligeramente menor con respecto a Prophet lo cual concuerda con ese 1% de perdida reflejado en el MAPE, mostrando el valor de perdida de error absoluto.

Donde se nota una diferencia un tanto mayor es en el error cuadrático medio donde hay 1 punto más en Random Forest con respecto a Prophet. En este caso como Random Forest es uno de los algoritmos que mayor parcialidad tiene a la hora de generar muestras aleatorias, es de entender que pueda producir una perdida esperada mayor pues no toma en cuenta información que podría ayudar a una estimación más precisa, siendo más aleatorio.

Podemos observar el desempeño de Random Forest fue ligeramente inferior al algoritmo Prophet, considerando algunos datos que puedan afectar el desempeño de los modelos como la calidad de los datos analizados, la cantidad de los mismos y principalmente la relación entre los datos y factores extra en el funcionamiento general de modelo es de esperar que puedan generarse datos que sean menos precisos, aun así, el desempeño del algoritmo Random Forest es suficientemente bueno como para ser considerado, siendo Prophet ligeramente superior en este tipo de datos.

En el caso del segundo experimento, buscando generar datos con mayor similitud entre ambos algoritmos y dado que Prophet requiere necesariamente del uso de la variable de tiempo para generar datos, se evaluó el algoritmo de Random Forest en una regresión utilizando la variable tiempo. Primero que todo es importante remarcar que Random Forest no reconoce como tal el formato de tiempo, por lo que primero se tuvo que procesar la información para simular el tiempo de manera aproximada la variable de tiempo.

Una vez procesada podemos ver que el resultado de los datos es bastante preciso en comparación al realizado por Prophet. La tendencia es muy similar en ambos casos, pero podemos analizar que ofrece más información el modelo hecho con Prophet.

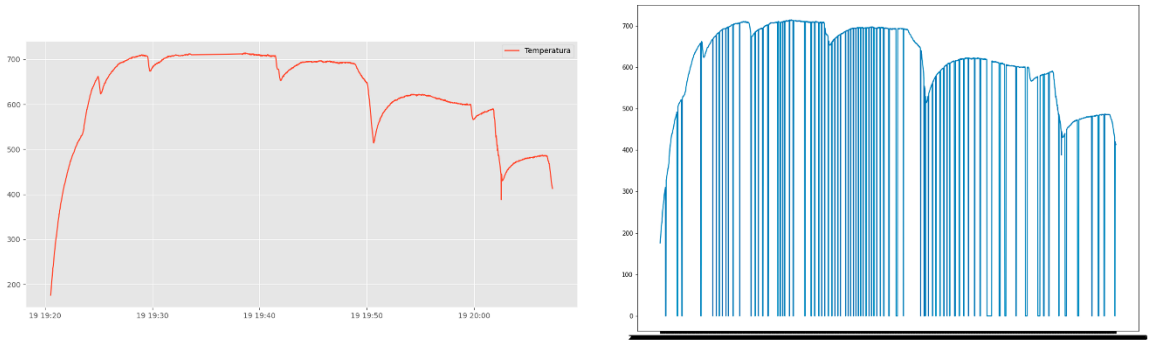

Figura 16. a) Lado izquierdo muestro de datos con Random Forest usando las variables de Fecha/Hora (x) y Temperatura (y). b) Lado derecho muestro de datos con Prophet usando las variables de Fecha/Hora (x) y Temperatura (y).

Con base a esos datos se realizó la regresión con ambos modelos, en el caso de Random Forest solo se muestra la predicción realizada de los datos, mientras que en el algoritmo Prophet podemos ver el muestro y continua con la predicción de los datos.

En el caso de Random Forest se muestran en azul los valores actuales mientras que en rojo los valores de la predicción realizada por Random Forest, también es importante aclarar que la variable predictor (X) es el tiempo procesado, recordando que Random Forest no procesa la variable tiempo, entonces se debe tratar para ser reconocida. Del dalo derecho se puede observar el orden de los valores con las diferentes estacionalidades en líneas rojas verticales y la tendencia se muestra en una gráfica lineal roja con la predicción en celeste, creciendo exponencialmente conforme avanza a lo largo del tiempo. Es evidente que el desarrollo visual es mejor en Prophet puesto que Random Forest no está planeado para realizar series de tiempo. A pesar de eso la regresión de Random Forest no está demasiado alejada de los valores actuales mostrados en azul, siendo una aproximación relativamente precisa para un área en la que no se especializa.

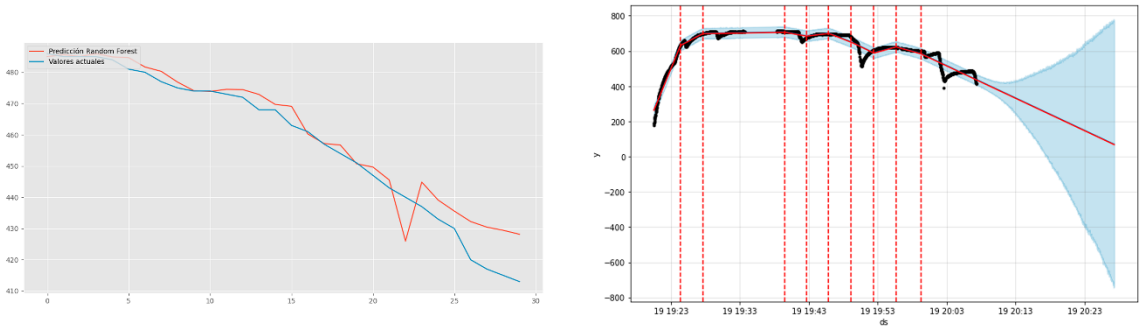

Figura 17. a) Lado izquierdo regresión con Random Forest usando las variables de Fecha/Hora (x) y Temperatura (y). Lado derecho regresión con Prophet usando las variables de Fecha/Hora (x) y Temperatura (y).

En cuanto al desempeño de este experimento podemos observar que de acuerdo a los datos el desempeño es un tanto más complejo de evaluar el desempeño puesto que Random Forest muestra resultados más precisos pues se limita únicamente a los datos de valores actuales y no hacer predicciones a futuro, puesto que el análisis a través del tiempo no es el fuerte principal de Random Forest.

Tabla 14. Comparativa métricas RF y Prophet.

Métricas	**Regresión RF (Tiempo/TEMP)**	**Regresión Prophet (Tiempo/TEMP)**
MAPE	0.009863143513462238	0.080603
MAE	4.303114682539697	49.074613
MSE	6.38629626721255	2769.541050

Los resultados del segundo experimento usando el set de datos de fundición con las variables de Fecha/hora y Temperatura fueron los siguientes. En el caso del error cuadrático medio (MSE) podemos ver que la función de riesgo en el caso de Random Forest empieza a aumentar, pero en el caso de Prophet el modelo evalúa un riesgo demasiado elevado para lo esperado por el modelo, siendo que estima la pérdida de 2769 datos por punto porcentual, en el caso de Random Forest siendo solo 6. Para el caso del error absoluto medio estima que va a terminar perdiendo 49 datos del total absoluto para Prophet y solo 4 puntos para el caso de Random Forest. Podemos observar que en el caso de Prophet si se perdió un 8% de la información comparado con el 0.9% de perdida de datos que hubo, podemos ver que, apoyados de un buen procesamiento de la información, una mayor cantidad de datos, una relación correcta de los datos y un buen entrenamiento del modelo el resultado de Random Forest mantiene un porcentaje de perdida muy bajo y un buen desempeño, a pesar de que series de tiempo no sea el fuerte de Random Forest.

CAPITULO V. CONCLUSIONES.

Durante el desarrollo de esta investigación se realizó, con el modelo de bosques aleatorios una aplicación que en cumple con el objetivo planteado inicialmente, pues se consiguió que la aplicación analice satisfactoriamente la relación entre variables con diversos procesos de manufactura reales, se consiguió implementar el algoritmo para analizar distintas variables como Temperatura, Presión o CO2, además de realizar una prueba de bosques aleatorios para análisis en series de tiempo, se logró representar gráficamente los resultados gracias a la librería de Matplotlib y Seaborn pues esto nos permitió reflejar la regresión con el modelo en graficas de líneas con el que se reflejan las tendencias que ayudan a realizar su debido análisis, además de esto se usaron distintas métricas para evaluar el desempeño de la regresión con el modelo de bosques aleatorios, así como métricas de precisión que nos permiten evaluar cuan preciso fue el modelo.

Los resultados obtenidos pueden ser considerados como satisfactorios pues en el caso de los primeros experimentos el set de datos contenía un número limitado de datos considerando que Random Forest es uno de los algoritmos que se ve menos afectado por le sobreajuste por lo que utilizar una mayor cantidad de datos hubiera sido provechoso para obtener mejores resultados. Aun con los datos utilizados, las métricas de desempeño y precisión arrojaron resultados bastante cercanos a los valores esperados por cada métrica, teniendo perdidas mínimas de información durante la regresión. En el segundo experimento se obtuvo un desempeño aceptable considerando que Random Forest está pensado para el análisis de series de tiempo, aun así, el resultado obtenido fue bueno considerando estas problemáticas. El tercer experimento podemos ver cómo incluso el resultado es aún más preciso pues la cantidad de datos era bastante alta y el procesamiento de los datos era bueno.

Las métricas de precisión mostraron que el modelo de Prophet realizó un mejor trabajo, puesto que la relación del dataset se hizo pensando en series de tiempo, afectado la relación de variables entre sí, algo que se vio reflejado en el desarrollo del algoritmo de bosques aleatorios. Si bien, en ambos algoritmos durante el primer experimento el desempeño fue similar, la métrica del promedio de los errores al cuadrado (MSE) señala un trabajo más estable para el caso de Prophet.

En el segundo experimento vemos un desarrollo de los experimentos completamente diferente. El algoritmo de Bosques Aleatorios tuvo un desempeño visiblemente mejor puesto que en el caso de las tres metricas, los resultados reflejan mejores datos en comparación con Prophet, siendo el caso de la diferencia entre dos variables continuas (MAE) y la métrica del promedio de los errores al cuadrado (MSE) donde el experimento realizado en Prophet obtuvo resultados de casi 50 puntos en la primer métrica y más de 2500 en el caso de la segunda, en comparación de los 4 y 6 puntos mostrados por el algoritmo de bosques aleatorios. Estos no son valores muy buenos puesto que estas metricas buscan en todo momento estar lo más cercanas a cero, aun asi, dado que para el desarrollo de series de tiempo es necesario el uso de cantidades de datos de grandes cantidades de tiempo y por tanto requiere de una enorme cantidad de datos, el buen desempeño en relación al sobreajuste que nos ofrece el algoritmo de bosques aleatorios minimiza considerablemente el error en las métricas, reflejando resultados menos alejados del propósito final.

TRABAJOS FUTUROS.

Para futuros trabajos se recomienda el uso de dataset de mayor volumen de datos, con una relación entre datos clara y sin factores externos que puedan alterar la relacionalidad de los dato. Se recomienda ampliar la cantidad de muestreo a un periodo mayor de tiempo, los algoritmos se recomienda aplicarlo a otros experimentos con variables diferentes a las trabajadas en esta tesis.

Otro factor a tomar en cuenta es la comparativa de Random Forest con otros algoritmos basados en arboles de decisión como podría ser XGBoost, por lo que sería de interés a futuro desarrollar una investigación del algoritmo de Bosques aleatorios con otros modelos de su área.

REFERENCIAS Y FUENTES DE INFORMACIÓN .

Alberca, A. S. (2022, 12 mayo). La librería Numpy. Aprende con Alf. Recuperado 9 de junio de 2022, de https://aprendeconalf.es/docencia/python/manual/numpy/

Antosz, K., Kozłowski, E., Sęp, J. y Żabiński, T. (2022, mayo). El uso de bosques aleatorios para apoyar el proceso de toma de decisiones para la fabricación sostenible. En Journal of Physics: Conference Series (Vol. 2198, No. 1, p. 012006). Publicación IOP.

AskPython. (2020, 21 septiembre). Random Forest Regression: A Complete Reference. https://www.askpython.com/python/examples/random-forest-regression.

Breiman L (2001). “Random Forests”. Machine Learning.

Bressert, Eli (2012). Scipy and Numpy: An Overview for Developers. O'Reilly. ISBN 978-1-4493-0546-8.

Cardellino, F. (2021, 20 marzo). La guía definitiva del paquete NumPy para computación científica en Python. freeCodeCamp.org. Recuperado 9 de junio de 2022, de https://www.freecodecamp.org/espanol/news/la-guia-definitiva-del-paquete-numpy-para-computacion-cientifica-en-python/

Citibanamex, & Rodriguez, G. (2021, julio). Indicadores Regionales de Actividad Económica 202 (N.o IRAE20210721). Direccion de estudios económicos. https://www.banamex.com/sitios/analisisfinanciero/pdf/revistas/IRAE/IRAE20210721.pdf

El uso de NumPy. (2021, 12 diciembre). frankgalandev. Recuperado 9 de junio de 2022, de https://frankgalandev.com/el-uso-de-numpy/

Estevez, M. (2017, 4 septiembre). RANDOM FOREST EN PYTHON. Inteligencia Analítica. https://inteligencia-analitica.com/random-forest-python/.

Gamal, M., Donkol, A., Shaban, A., Costantino, F., Di Gravio, G., & Patriarca, R. (2021). Anomalies Detection in Smart Manufacturing Using Machine Learning and Deep Learning Algorithms. Proceedings of the International Conference on Industrial Engineering & Operations Management, 1611–1622.

Deep Learning Algorithms. Proceedings of the International Conference on Industrial Engineering & Operations Management, 1611–1622.

Heras, J. M. (2020, 19 septiembre). Las 7 Fases del Proceso de Machine Learning. IArtificial.net. https://www.iartificial.net/fases-del-proceso-de-machinelearning/

Ho, Tin Kam (1995). Random Decision Forests. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal.

Hu, J. (2021, 18 julio). Tutorial de Numpy - NumPy Multidimensional Array-ndarray. Delft Stack. Recuperado 9 de junio de 2022, de https://www.delftstack.com/es/tutorial/python-numpy/numpy-ndarray/#:%7E:text=NumPy%20es%20una%20biblioteca%20que%20utiliza%20matrices%20multidimensionales,list%20data%20type%2C%20porque%20list%20funciona%20relativamente%20despacio.

José Antonio García, Ana Ferrero (2005). «Representación gráfica 2D: Matplotlib» (pdf). Linux Magazine (Málaga, España: Linux New Media Spain) (11): 53 a 56. Archivado desde el original el 24 de marzo de 2012.

Losilla, J. M., Navarro, B., Palmer, A., Rodrigo, M. F. y Ato, M. (2005). Del contraste de hipótesis al modelado estadístico. Documenta Universitaria.

Manotoa Moreno, A. I. (2021, 22 septiembre). Tutorial de NumPy en Español. Deepnote. Recuperado 9 de junio de 2022, de https://deepnote.com/@anthonymanotoa/Tutorial-de-NumPy-en-Espanol-180f7d51-b297-4aea-b61e-34ef867ca6fb

McKinney, W. (2015). Pandas: a Python data analysis library de http://pandas. pydata.org.

McKinney, Wes (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython (2nd edición). Sebastopol: O'Reilly. ISBN 978-1-4919-5766-0.

Mesa, A. R. (2020, 27 octubre). Por qué usar NumPy. OpenWebinars.net. Recuperado 9 de junio de 2022, de https://openwebinars.net/blog/por-que-usar-numpy/#:%7E:text=Algunas%20de%20las%20caracter%C3%ADsticas%20principales%20de%20NumPy%20que,mundo%2C%20probablemente%20usar%C3%A1%20a%20diario%20la%20librer%C3%ADa%20NumPy.

Mitchell, T. (1997). Aprendizaje automático (Vol. 1, No. 9). Nueva York: McGraw-hill.

NumPy.org. (s. f.). NumPy. NumPy. Recuperado 9 de junio de 2022, de https://numpy.org/

NumPy Community. (2022, Mayo) NumPy User Guide. Release 1.22.4 Recuperado 9 de junio de 2022, de https://numpy.org/doc/stable/numpy-user.pdf

Prihatno, A. T., Nurcahyanto, H., & Jang, Y. M. (2021, abril). Predictive Maintenance of Relative Humidity Using Random Forest Method. IEEE. https://doi.org/10.1109/ICAIIC51459.2021.9415213

Prophet. (s. f.). Prophet. https://facebook.github.io/prophet/

Puggini, L., Doyle, J. y McLoone, S. (2015). Detección de fallas utilizando distancia de similitud de bosque aleatorio. IFAC-PapersOnLine, 48 (21), 583-588.

Sandoval, LJ (2017, noviembre). Algoritmos de aprendizaje automático para análisis y predicción de datos. En 2017 IEEE 37th Convención de Centroamérica y Panamá (CONCAPAN XXXVII) (pp. 1-5). IEEE.

Wu, D., Jennings, C., Terpenny, J., Gao, RX y Kumara, S. (2017). Un estudio comparativo sobre algoritmos de aprendizaje automático para la fabricación inteligente: predicción del desgaste de herramientas utilizando bosques aleatorios. Revista de ciencia e ingeniería de fabricación, 139 (7).

ANEXOS.

Anexo 1.

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import seaborn as sns
```

Anexo 2.

```
     df=pd.read_csv(r'C:\Users\ca1\Documents\ResidenciasAdrianAgun
dis\docs\sdp(1).csv')
```

Anexo 3.

```
     df.describe()

df.info()

print(df.dtypes)

print(df.size)
```

Anexo 4.

```
      plt.figure(figsize=(30,10))
sns.countplot(x='Fecha/Hora',data=data)
plt.xticks(rotation=45)
plt.show()
```

Anexo 5.

```
x = data.iloc[:,2:3].values
print(x)
y = data.iloc[:, 1].values
print(y)
```

Anexo 6.

```
      from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.15, random_state=19)
x_train
print(x_train)
x_test
print(x_test)
```

Anexo 7.

```
"""from sklearn.preprocessing import MinMaxScaler, StandardScaler
minmax_scaler = MinMaxScaler().fit(df_X_train)
df_X_norm_train = minmax_scaler.transform(x_train)
df_X_norm_test = minmax_scaler.transform(x_test)
std_scaler = StandardScaler().fit(df_X_train)
df_X_norm_train = std_scaler.transform(x_train)
df_X_norm_test = std_scaler.transform(x_test)"""
```

Anexo 8.

```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 100, random_state
= 0)
regressor.fit(x_train, y_train)
X_grid = np.arange(min(x_train), max(x_train),.001)
X_grid = X_grid.reshape((len(X_grid), 1))
plt.scatter(x_train, y_train, color = 'purple')
plt.plot(X_grid, regressor.predict(X_grid),
color = 'orange')
plt.title('Random Forest Regression')
plt.xlabel('CO2')
```

```
plt.ylabel('Temperatura')

plt.show()
```

Anexo 9.

```
dt = df[df["Temperatura"]==0].index

dtf = df.drop(dt)

dc = dtf[dtf["CO2"]==0].index

dcf = dtf.drop(dc)

dp = dcf[dcf["Presion"]==0].index

dpf = dcf.drop(dp)

da = dpf[dpf["Aire"]==0].index

daf = dpf.drop(da)

dat = daf[daf["Aire"]> 7400 ].index

data = daf.drop(dat)
```

Anexo 10.

```
dt = ds[ds["Temperatura"]==0].index

data = ds.drop(dt)
```

Anexo 11.

```
data["dteday"] = pd.to_datetime(data['Fecha/Hora'])

data["Hora"]= data["dteday"].dt.hour

df = data[["dteday", "Temperatura"]]

df = df.set_index("dteday")

df.head()
```

Anexo 12

```
data["Temperatura1"] = data['Temperatura'].shift(+1)

data["Temperatura2"] = data['Temperatura'].shift(+2)

data["Temperatura3"] = data['Temperatura'].shift(+3)

data["Temperatura4"] = data['Temperatura'].shift(+4)

data["Temperatura5"] = data['Temperatura'].shift(+5)

data= data.dropna()

print(data)
```

Anexo 13

```
t1,t2,t3,t4,t5,y =
data['Temperatura1'],data['Temperatura2'],data['Temperatura3'],dat
a['Temperatura4'],data['Temperatura5'],data['Temperatura']

t1,t2,t3,t4,t5,y =
np.array(t1),np.array(t2),np.array(t3),np.array(t4),np.array(t5),
np.array(y)

t1,t2,t3,t4,t5,y = t1.reshape(-1,1), t2.reshape(-1,1),
t3.reshape(-1,1), t4.reshape(-1,1), t5.reshape(-1,1),y.reshape(-
1,1)

X= np.concatenate((t1,t2,t3,t4,t5), axis=1)

print(X)
```

Anexo 14.

```
sns.lmplot(x = 'Rotational speed [rpm]',
           y = 'Torque [Nm]',
           hue = 'Machine failure',
           data = data)
plt.show()
```