UNIVERSIDAD DE PIURA

FACULTAD DE INGENIERÍA

Estado del arte de la inteligencia artificial y su aplicación en el mantenimiento

Tesis para optar el Título de
Ingeniera Mecánico - Eléctrica

María Gratzia Tume Fuentes

Asesor:
Mgtr. Ing. Jorge Arturo Yaksetig Castillo

Piura, abril de 2022

UNIVERSITAS STUDIORUM PIURENSIS

A Dios y la Virgen, que siempre cuidan a mi familia.

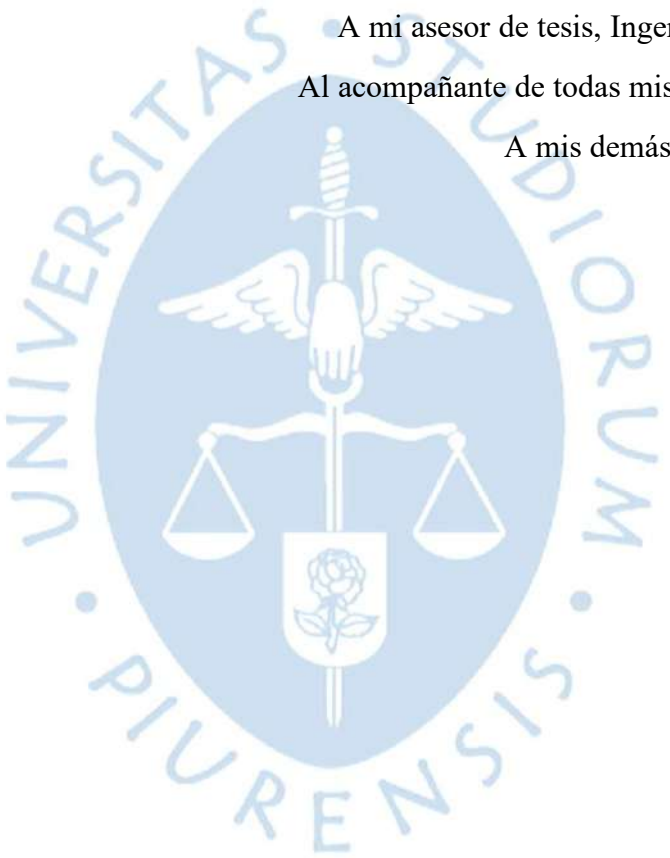
A mis padres, Pedro y Lila, por su guía y apoyo incondicional.

A mis hermanos, Lila y Ricardo.

A mi asesor de tesis, Ingeniero Jorge Yaksetig.

Al acompañante de todas mis amanecidas, Rocky.

A mis demás familiares y amigos.



UNIVERSITAS STUDIORUM PIURENSIS

Resumen

En la industria en general, existen activos simples y complejos. Los activos simples pueden ser gestionados por operarios quienes monitorean sus parámetros de funcionamiento y están pendientes de alguna falla para intervenir y reparar o reemplazar. Sin embargo; los activos complejos, que suelen venir equipados con una serie de dispositivos mecánico-eléctricos-electrónicos y de comunicación, cuando presentan una falla puede resultar en un evento catastrófico por ello deben alertar al operador cuando detectan señales premonitorias de falla para que éste intervenga. Estos atributos predictivos son de gran ayuda para la gestión del mantenimiento. A pesar de ellos, hay objeciones a la función mantenimiento. Es así que para mejorar este orden de cosas aparece la Inteligencia Artificial (IA), cuyos inicios datan del año 1950 y que ha dejado de ser parte de nuestra imaginación, hoy en día se encuentra alrededor de nosotros, en nuestro día a día y mucha gente aún se da cuenta de ello.

La metodología aplicada fue de carácter teórico, se tuvo en cuenta que aún no se aprecia mucho el desarrollo de la inteligencia artificial aplicada al mantenimiento en nuestro país. Se culmina con un caso de estudio que tiene establecido los métodos y procedimientos para el estudio experimental que han aplicado en su industria, el cual ha podido ser desarrollado gracias al avance de los softwares, sistemas de información y la algorítmica. Lo que se busca es relevar la importancia de la IA en el mantenimiento industrial, para garantizar el funcionamiento de las máquinas y, cuando fallan, reducir los tiempos para ejecución de alguna intervención

En las industrias y empresas peruanas se recomienda empezar con la implementación de softwares relacionadas con la Inteligencia Artificial, ya que esto permitirá reducir costos y aumentar la producción usando menos mano de obra. Los constantes desarrollos en Big Data, la comunicación de máquina a máquina y la tecnología en la nube han abierto nuevas posibilidades para investigar la información derivada de los activos industriales. Es viable el monitoreo en tiempo real, gracias a los sensores, actuadores y otros parámetros de control.

UNIVERSITAS STUDIORUM PIURENSIS

Tabla de contenido

Tabla de figuras

UNIVERSITAS STUDIORUM PIURENSIS

Introducción

El mantenimiento en una planta industrial comprende una serie de actividades técnicas y administrativas para mantener los activos en su operación normal y, cuando fallan, devolverles su funcionalidad de diseño.

El desarrollo tecnológico actual, ha llevado a una nueva revolución industrial que se conoce como industria 4.0 que plantea la manufactura con todos los procesos y activos interconectados. La industria 4.0 se basa en una serie de disciplinas avanzadas como el Big Data, Internet Industrial de las cosas (IoT, por sus siglas en Inglés), Sistemas expertos, Robótica, Visión Artificial, Procesamiento de Lenguaje Natural, *Planning,* entre otras que se concatenan con apropiados algoritmos de *Machine Learning*, que junto con la sensórica conforman una nueva área de conocimiento que se conoce como Inteligencia Artificial que, aplicados al mantenimiento, permitirá a las empresas contar con grandes cantidades de datos funcionamiento de los activos y procesos, los cuales convenientemente analizados y procesados ayudarán a realizar pronósticos y detectar fallas y anomalías de forma oportuna lo cual naturalmente conduce hacia una optimización de la gestión del mantenimiento.

En este trabajo se hablará de las disciplinas involucradas en esta nueva área de conocimiento que es la inteligencia artificial.

UNIVERSITAS STUDIORUM PIURENSIS

Capítulo 1
La inteligencia artificial

1.1 Desarrollo histórico de la inteligencia artificial

La conferencia de *Darmouth* desarrollada en el verano de 1956 fue el germen de la Inteligencia Artificial (de ahora en adelante IA), aquí motivó a generaciones de científicos a explorar el potencial de tecnologías de información y su compatibilidad con los humanos. Se reunieron diversos investigadores como John McCarthy, quien los lideraba, Marvin Minsky, Claude Shannon y Nathalie Rochester, despertando el interés de otros investigadores entre ellos: Trenchard More de Princeton, Arthur Samuel de la IBM, Ray Solomonoff y Oliver Selfridge ambos del MIT y Allen Newel junto a Herbert Simon del Carniege Tech.

McCarthy propuso el nombre de IA por dos razones, la primera es la idea de duplicar la inteligencia humana (creatividad, automejora y el uso del lenguaje) y la segunda, por ser una rama de la informática cuyo objeto es perseguir la fabricación de máquinas que funcionen automáticamente en medios complejos y cambiantes. El nombre quedó adecuado a esos tiempos.

En 1962 Frank Rosemblatt publicó el libro Principios de Neuro dinámica: Perceptrones y la teoría de los mecanismos cerebrales, en el cual muestra un algoritmo de redes neuronales con una sola capa y pesos variables confería a un dispositivo electrónico, construido de acuerdo con principios biológicos, la habilidad de aprender (Aprendizaje automático).

En ese mismo año David Hubel y Torsten Wiesel publicaron “Campos receptivos, binoculares interacción y arquitectura funcional en la corteza visual del gato” (visión artificial), el cual informaba por primera vez las propiedades de respuesta de las neuronas individuales grabadas con un microelectrodo.

En 1969 Marvin Minsky y Seymour Papert publicaron el libro “Perceptrones” donde se indicaba las limitaciones computacionales de una neurona artificial, esto tuvo un efecto negativo en la financiación de proyectos de investigación en estos temas.

Sin embargo, en 1979, Geoffrey Hinton y James Anderson organizaron los modelos paralelos del taller de memoria asociativa en La Jolla, California, esto trajo una nueva generación de pioneros en redes neuronales que comenzaron a estudiar nuevamente la obra de Rosenblatt.

En 1987, tiene lugar la primera conferencia de Sistemas de Procesamiento de Información Neuronal (NIPS) y el taller se desarrolló en Denver Tech Center, reuniendo investigadores de diversos campos que exploran redes neuronales biológicas y artificiales. Hubo una amplia gama de temas y desde entonces se abrieron diversas corrientes de investigación sobre aprendizaje automático, IA y estadísticas.

David McAllester en 1998 indicó: "La IA se fundó en parte en el marco de una rebelión en contra de las limitaciones de los campos existentes como la teoría de control o la estadística, y ahora abarca estos campos" pero ahora es parte del ámbito de los métodos científicos.

En el 2005, Raymond Kurzweil predijo que las máquinas alcanzarán un nivel de inteligencia humano en el 2029.

Para el 2007 la IA ingresa a la comunidad a través de los teléfonos inteligentes.

En el 2012, Google crea un superordenador capaz de aprender a identificar gatos, así como caras y cuerpos humanos, mediante algoritmos de aprendizaje profundo (*Deep Learning*) lo cual demuestra el verdadero poder de las nuevas herramientas tecnológicas.

Entre los años 2014 y 2015 la IA logra un mejor acercamiento al comportamiento humano. En el 2014 un *bot* computacional (Eugene Goostman) fue capaz de engañar a un tercio de los jueces a los que se les sometió durante el test de Turing (prueba de la capacidad de una máquina para exhibir un comportamiento inteligente similar al de un ser humano), haciéndose pasar por un niño ucraniano de 13 años.

Yann LeCun, científico informático, director de investigación de IA en Facebook, que trabaja en los campos de aprendizaje automático, visión por computadora, robótica móvil y neurociencia computacional realizó uno de los primeros ejercicios de *deep learning*. en efecto, pensó en un sistema computacional que pudiera diferenciar entre perros y gatos, y que pudiera corregirse cuando no acierte hasta encontrar la distinción entre unos y otros en poco tiempo.

Rajan Goyal, uno de los actuales pioneros de la IA con especialidad en *big data* y *deep learning,* piensa en un sistema que haga que una máquina aprenda por sí misma en lugar de enseñarle, de ese modo el tiempo de desarrollo del sistema se reduciría drásticamente. Remarca que en la medida en que una máquina inteligente sepa más, recopilará información de manera más sencilla y rápida.

En los últimos años, la IA ha tenido amplios avances, como el caso de los investigadores del grupo *Soft Computing* y Sistemas de Información Inteligentes, de la Universidad de Granada, de la mano de investigadores del grupo Sistemas Inteligentes y Minería de Datos de la Universidad de Jaén han logrado desarrollar un software que permitirá disponer de sistemas de IA como soporte a la toma de decisiones para entornos de amenaza y conflicto. (Universidad de Granada, 2020)

Otro de los desarrollos de la IA es *Alpha Go*, programa informático desarrollado por *Deep Mind Technologies*, que derrotó a Lee Sedol, campeón profesional del juego de mesa de estrategia intuitiva y de muchas combinaciones, denominado *Go*.

Por otro lado, en el campo de la medicina, el año 2017 *Alpha Go*, desarrolló un motor de inteligencia artificial que superó a los radiólogos diagnosticando el cáncer de pulmón en su primera etapa. Mediante el uso de redes neuronales se ha podido replicar las capacidades humanas como: el reconocimiento de patrones y la identificación de imágenes haciendo uso de miles de imágenes de pruebas médicas.

Como desarrollo más reciente de la IA, en enero del 2020 un *startup* británico *Exscientia Limited* trabajando en conjunto con una farmacéutica japonesa presentaron una medicina obtenida en un tiempo equivalente al 20% del tiempo promedio necesario, la DSP-1181 para tratar pacientes con desorden obsesivo compulsivo. Fue creada usando algoritmos que evaluaron las diversas composiciones potenciales, comparándolos con una variada base de datos de diversos parámetros. Esta misma empresa viene trabajando en posibles medicamentos para el tratamiento del cáncer y las enfermedades cardiovasculares; se espera tener una molécula lista para ensayos clínicos para fines del 2020. La ilustración 1 grafica la cronología que hemos narrado anteriormente.

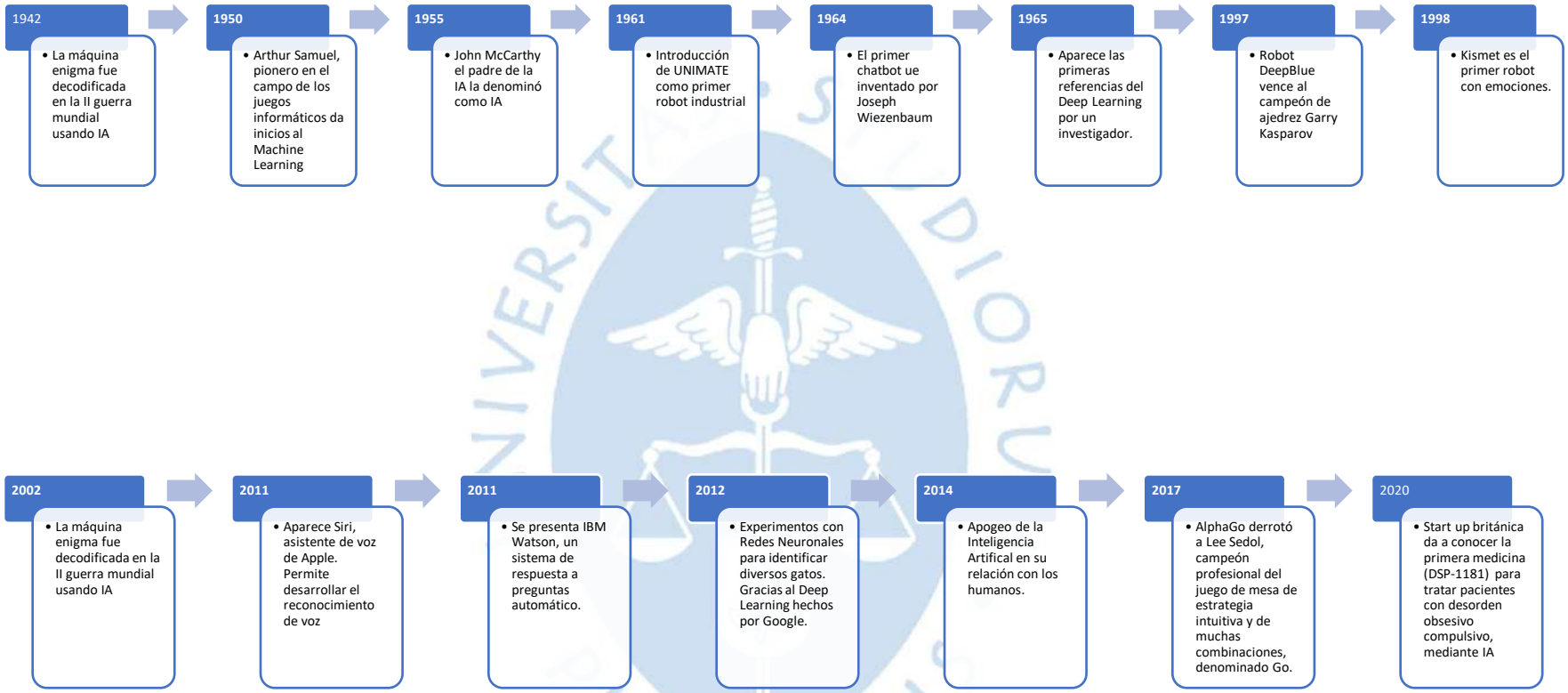


Figura 1. Cronología de los avances de la inteligencia artificial de los siglos 20 y 21

1.2 Disciplinas que comprende la inteligencia artificial

La IA es un amplio campo de estudio que incluye una serie de teorías, métodos y tecnologías que convergen en dispositivos a los cuales se les provee de softwares adecuados para que puedan percibir un entorno, procesar lo percibido y producir un resultado óptimo y consistente.

Para proveer IA a un sistema de modo que pueda aprender, entender, razonar y tomar decisiones es necesario apoyarse en diversas disciplinas como:

a) las ciencias de la computación,

b) la lógica,

c) las neurociencias y

d) la matemática.

que se describirán a continuación:

1.2.1 Ciencias de la computación

Las ciencias de la computación han tenido una evolución en el tiempo; sus inicios datan del siglo XVII, en efecto, en 1805 aparece la primera máquina programable, un telar, desarrollado por Joseph Marie Jacquard, el cual utilizaba tarjetas perforadas para almacenar información sobre los parámetros bordados.

A mitad del siglo XIX, Charles Babbage diseñó dos máquinas: la calculadora mecánica y la calculadora analítica. La primera fue construida y era capaz de calcular tablas de funciones numéricas por el método de las diferencias. La segunda, que podía ejecutar programas de computación no fue construida. Por eso se le considera como “El padre de la computación”.

En 1940, Alan Turing construyó el primer computador operacional de carácter electromecánico con la finalidad de descifrar mensajes alemanes en la segunda guerra mundial. Por estos años, entre 1940 y 1942, aparece el primer computador electrónico ABC, fue creado por John Atanasoff junto a su discípulo Clifford Berry en la Universidad Estatal de Iowa.

En 1991 se construyó la “Calculadora Analítica” que diseñó Babbage a mitad del siglo XIX y se presentó en el museo de la ciencia de Londres. Esta máquina era mucho más ambiciosa ya que contenía memoria direccionable, programas almacenados, saltos incondicionales. Este fue el primer dispositivo dotado de los elementos necesarios para realizar una computación universal. A partir de ahí han aparecido varias generaciones de dispositivo hardware cada vez con mayor velocidad de procesamiento y capacidad de almacenamiento.

Todo este conocimiento que abarcan las bases teóricas de la información y la computación se conoce como las ciencias de la computación que aplicada en dispositivos

los hace capaces de correlacionar una gran cantidad de datos recibidos y procesarlos con el objetivo de producir información válida para la toma de decisiones.

La IA se apoya en las ciencias de la computación para el desarrollo de procesos que permiten a ciertos dispositivos percibir su entorno (recibiendo entradas normalmente a través de sensores), procesar tales percepciones en una arquitectura física y/o computacional (a la cual se le ha almacenado previamente un conocimiento) y actuar (promocionar salidas) conforme a ciertos principios de optimización o maximización de rendimiento. En resumen, dotar a tales dispositivos de cierta racionalidad (inteligencia) para alcanzar algún objetivo o finalidad. (Ciencias de la computación, s.f.)

1.2.2 Lógica

El ser humano a lo largo de su historia siempre se ha planteado retos, comenzó creando dispositivos, equipos que multipliquen su fuerza física (limitada por su naturaleza) y su velocidad y capacidad de desplazamiento. Posteriormente buscó dispositivos que amplíen su capacidad intelectual y velocidad de procesamiento de información, inventando para ello el computador.

Seguidamente, se planteó el reto de crear máquinas pensantes, es decir, que razonen, resuelvan problemas, demuestren teoremas, perciban visualmente, reconozcan la voz, etc. Para esto se acudió al formalismo de la lógica y su forma de tratar los procesos mentales para modelarlos a través de procesos artificiales que, utilizando un dispositivo computacional y algoritmos especializados (redes neuronales, por ejemplo), "sigan la trayectoria aproximada" del proceso mental para ofrecer salidas adecuadas que permitan alcanzar un objetivo.

En ese sentido, John McCarthy, sostenía que el camino para obtener la inteligencia de las máquinas pasaba por un enfoque formal riguroso, en el cual los actos que componen la inteligencia son reducidos a una serie de relaciones o axiomas lógicos que pueden expresarse de forma precisa en términos matemáticos. Esto naturalmente no es tarea fácil, pero es el camino que sigue la IA para dotar, cada vez más, de mejores características de inteligencia a las máquinas. (Familia, 2014)

1.2.3 Neurociencia

La neurociencia estudia el cerebro. Este viene siendo estudiado desde varios siglos atrás, desde el siglo XVIII en que Paul Broca localizó áreas del cerebro responsable de las funciones cognitivas y Camilo Golgi que desarrolló una técnica de observación de neuronas individuales del cerebro. Luego con el desarrollo del encefalograma y las imágenes de resonancia magnética se tiene más detalle de la actividad cerebral. Estos avances tienen el objetivo científico de entender el funcionamiento del cerebro, el cual se calcula que se compone 86000 millones de neuronas y entre 100 y 500 billones de conexiones entre ellas. Investigaciones recientes se acercan más a:

- Entender los mecanismos de conducta humana.
- Estudiar la actividad cerebral casi en tiempo real mediante la magneto-encelografía.

El mayor desarrollo de la neurociencia y la evolución de la IA generará un beneficio mutuo entre ambas; en efecto, esto permitirá reducir la brecha entre la inteligencia humana y la inteligencia artificial.

En la actualidad la IA viene desplegando todo su potencial para construir conocimiento inspirado en las facultades mentales de aprender, entender, razonar y tomar decisiones. (InNeurociencia, 2018)

1.2.4 Matemáticas

Las técnicas de IA en general se basan en diversos aspectos de la matemática en particular en:

- Elementos de álgebra lineal (vectores, matrices y operaciones entre ellos)
- Análisis de funciones de variables reales (derivadas, máximos, mínimos e integrales)
- Geometría
- Fundamentos de la teoría de la probabilidad y estadística básica, descriptiva e inferencial.

El conocimiento de estos aspectos, son el inicio del camino del aprendizaje automático de las máquinas. Los algoritmos utilizados en modelos de IA utilizan la notación vectorial/matricial y las diversas operaciones entre matrices. Ahondaremos específicamente en dos de tales campos: álgebra lineal y la estadística

El álgebra lineal es la matemática de los datos. La notación matemática permite describir con precisión operaciones sobre los datos con operadores específicos. El álgebra lineal fue desarrollada originalmente para resolver sistemas de ecuaciones lineales. Se trata de casos en los que hay más ecuaciones que variables desconocidas. Como resultado, son difíciles de resolver aritméticamente porque no hay una solución única porque no hay una línea o un plano que pueda ajustar los datos sin ningún error.

El método de los mínimos cuadrados, conocidos por su papel en la solución de modelos de regresión lineal, son bastante utilizados y puede sistematizarse mediante algoritmos que tienen como base el conocimiento de algebra lineal, y buen desempeño en el campo de la IA. (Gonzalez, 2020)

Un profesional de la IA es capaz de leer y escribir notación vectorial y matricial; los algoritmos se desarrollan valiéndose de esa notación. El lenguaje de programación que actualmente brilla con luz propia es *Python* ofreciendo formas eficientes de implementar soluciones de manera corta y eficiente de algoritmos que hacen uso del álgebra lineal.

Las estadísticas y el análisis de datos es otro aspecto de las matemáticas que se ocupan de ordenar y comprender los datos. Las estadísticas modernas utilizan la notación

del algebra lineal en sus métodos, desde vectores para las medias y varianzas de los datos, hasta matrices de covarianzas que describen las relaciones entre múltiples variables Gaussianas.

Los resultados de algunas colaboraciones entre el álgebra lineal y la estadística son aplicados en la IA como, por ejemplo: el análisis de componentes principales utilizado para las reducciones de los conjuntos de datos, la factorización de matrices y sus diferentes métodos algunos de los cuales dependiendo de sus fortalezas y capacidades se pueden utilizar como herramientas para la IA, como la descomposición del valor singular, utilizado también para la reducción de datos.

La relación entre la estadística y la IA es biunívoca; en efecto, las aplicaciones de la IA en la estadística tienen como objetivo integrar diversos contrastes, estimaciones, transformaciones y modelos para conseguir una aproximación coherente y total en análisis de datos, estableciendo estrategias que dirijan el proceso de modelización, de elección de técnicas y transformaciones a aplicar, y de ayuda a la interpretación de los resultados. (Ocerín, s.f.)

1.3 La algorítmica y la inteligencia artificial

Cuando se pretende establecer la relación ente la matemática y la IA aparecen los algoritmos como un puente entre ellos. Un algoritmo es una secuencia de instrucciones que representan un modelo de solución para cierto tipo de problemas. O bien como un conjunto de instrucciones que realizadas en orden conducen a obtener la solución de un problema. (UNNE, s.f.)

Una vez diseñado el algoritmo se realiza un programa cuyo desarrollo requiere un conocimiento de las técnicas de programación. El programador experto, Luis Joyanes, dice que los algoritmos son más importantes que los lenguajes de programación o las computadoras, ya que, un lenguaje de programación es sólo un medio para expresar un algoritmo y una computadora es solo un procesador para ejecutarlo.

La IA es una combinación de algoritmos planteados de manera que confieren a las máquinas capacidades como: aprender a partir de un conocimiento previo y/o conocimiento adquirido de la experiencia, ajustarse a nuevas aportaciones de datos y realizar tareas de una manera semejante a la forma en que las realizaría el ser humano.

En resumen, la IA se construye mediante algoritmos (con capacidades matemáticas de aprendizaje) y los datos para entrenarlos. Existe diversos algoritmos para la IA, se describirá en los siguientes apartados los algoritmos de aprendizaje automático, aprendizaje profundo. (Redacción España, 2020)

1.3.1 El aprendizaje automático (*machine learning*)

El *machine learning* o aprendizaje automático es una disciplina que pertenece al ámbito de la IA, cuyo objetivo es crear máquinas o sistemas que "aprenden automáticamente". Aprender en este contexto significa identificar patrones complejos entre millones de datos. El sistema que realmente aprende es un algoritmo que recibe los datos, los revisa, los procesa y es capaz de predecir comportamientos futuros, reconocer imágenes, etc. Aprender automáticamente significa que estos sistemas mejoran de forma autónoma con el tiempo, sin intervención humana.

Según el profesor Tom Mitchell*, machine learning* es una consecuencia natural de la intersección de la informática y la estadística. En realidad, se asocia con una serie de disciplinas tal como se señala en la figura 2.

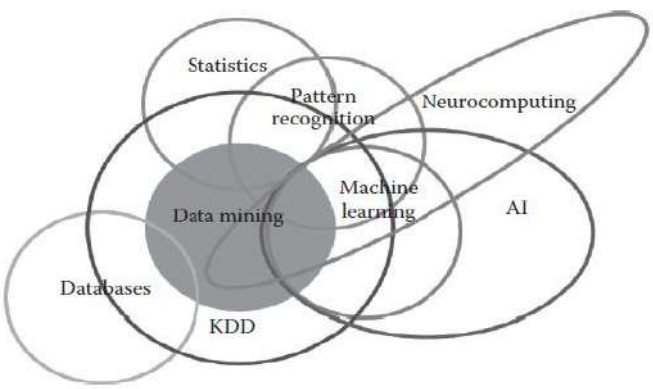


Figura 2. Diferentes Disciplinas y su relación con el *Machine Learning*
Fuente: http://blogs.sas.com/

1.3.2 El aprendizaje profundo (*deep learning*)

El aprendizaje profundo (*deep learning*) es un subconjunto del aprendizaje automático (*machine learning*), se define como un algoritmo automático estructurado o jerárquico que emula el aprendizaje humano para obtener ciertos conocimientos. No requiere de reglas programadas previamente, sino que el propio sistema es capaz de aprender por sí mismo para efectuar una tarea a través de una fase previa de entrenamiento. Generalmente está compuesto por redes neuronales artificiales entrelazadas en un cierto número de niveles o capas jerárquicas, para el procesamiento de la información. (Smart Panel, s.f.)

El primer nivel de esta jerarquía expone que la red aprende algo simple y luego envía esta información al siguiente nivel. En el siguiente nivel se toma esta información sencilla, la combina, compone una información algo un poco más compleja, y se lo pasa al tercer nivel, y así sucesivamente. para que realice tareas como las que realizan los seres humanos, como el reconocimiento del habla, la identificación de imágenes o hacer predicciones. (Moreno, s.f.)

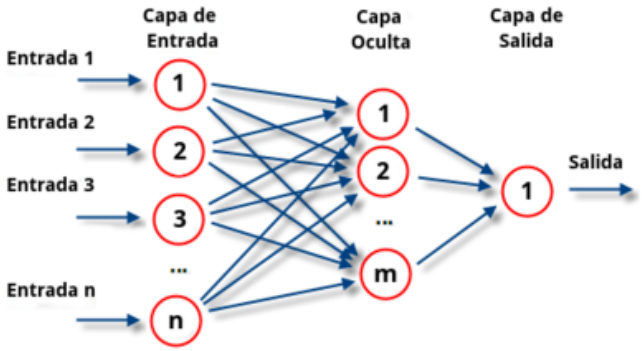


Figura 3. Primer nivel: red aprende algo simple, siguiente nivel: se toma la información sencilla y se combina, tercer nivel: continúa pasando la información
Fuente: https://core.ac.uk/download/pdf/44310843.pdf

Los algoritmos de *deep learning* tienen una naturaleza iterativa, esta complejidad aumenta el número de capas y los grandes volúmenes de datos que se necesitan para entrenar a las redes. Los métodos de *deep learning*, tienen una capacidad de mejorar y adaptarse continuamente a cambios en el patrón de información implícito, esto presenta una gran oportunidad para introducir un comportamiento más dinámico a la analítica. (Software y Soluciones de Analítica, s.f.)

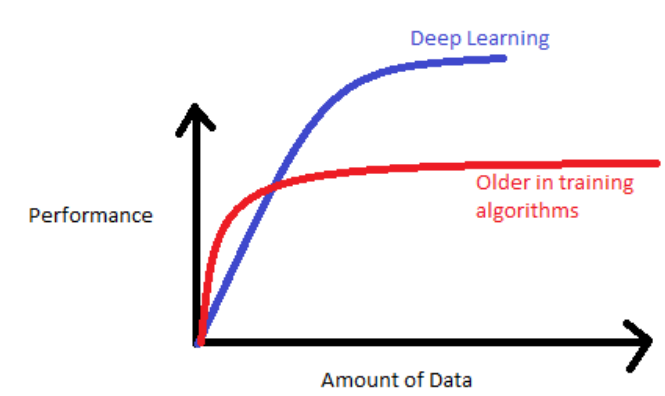


Figura 4. Escalabilidad del *deep learning*
Fuente: *www.slideshare.net/ExtractConf/andrew-ng-chief-scientist-at-baidu*

Las técnicas de *deep learning* se emplean en aplicaciones como: el lenguaje natural hablado y escrito, reconocimiento de voz, visión por computador, reconocimiento de cara, interpretaciones semánticas o los traductores inteligentes. A nivel industrial, donde se maneja gran cantidad de datos, la información resultante y oportuna resulta en una gran ventaja comparativa.

Entre los tipos de redes que utiliza *deep learning* se encuentran: redes prealimentadas, redes convolucionales, redes recurrentes, *deep autoencoders* y se describirá brevemente cada una de ellas:

- Redes prealimentadas o *feedforward*: se refiere a arquitecturas profundas y extensas, es decir, que poseen un gran número de capas y neuronas por capas; más de las que se emplean normalmente. Esto permite representar no linealidades complejas con un número mucho menor de unidades y buenas propiedades de generalización. Se usan en aplicaciones de reconocimiento facial, que, llevadas al ámbito industrial, se pueden usar en el diagnóstico de procesos mediante técnicas de visión por computador. (Goodfellow)

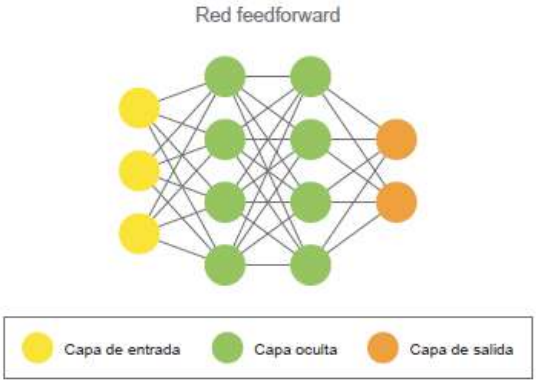


Figura 5.Esquema básico de un feedforward.
Fuente:*https://es.scribd.com/document/421929039/Aprendizaje-Profundo-andrea-pdf*

Los mínimos cuadrados son más conocidos por su papel en la solución de modelos de regresión lineal, pero también desempeña un papel más amplio en una gama de algoritmos de *machine learning*. Por estas razones es necesario el conocimiento de algebra lineal para un mejor desempeño en el campo de la IA. (Gonzalez, 2020)

- Redes convolucionales: Este tipo de redes es similar a las *feedforward*, con la diferencia que incorpora una serie de capas de convolución en su arquitectura, las cuales están compuestas por bancos de filtros cuyos coeficientes son ajustados para optimizar la función de coste. Estas arquitecturas, han mostrado una extraordinaria capacidad de detección de patrones en señales, por lo que se usan con mucho éxito en aplicaciones como el reconocimiento de voz y el análisis de vibraciones. (Sainath, 2013)

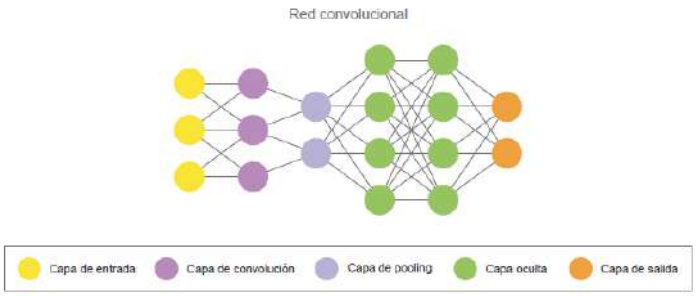


Figura 6. Esquema básico de una red convolucional.
Fuente: *https://es.scribd.com/document/421929039/Aprendizaje-Profundo-andrea-pdf*

- Redes recurrentes: Este tipo de arquitectura trabaja con bucles de realimentación que permite que la información persista y su máximo exponente son las redes LSTM (*Long Short Term Memory).* Estas son capaces de aprender dependencias a corto y largo plazo. Su naturaleza está íntimamente relacionada con secuencias y listas, por lo que es muy efectivo en ese tipo de datos. Llegan a predecir el próximo carácter en una secuencia de texto, lo cual se asimila a la predicción de comportamiento en un sistema de ingeniería. (Sutskever, 2011)

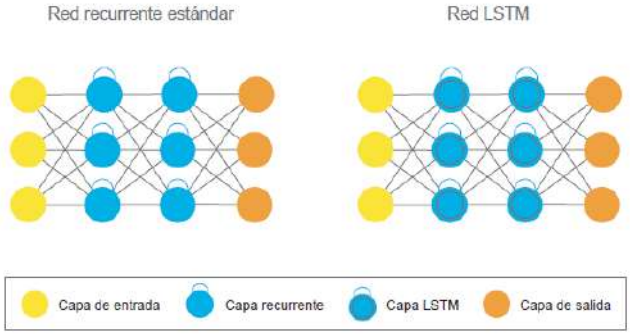


Figura 7.Esquema básico RNN y LSTM.
Fuente:*https://es.scribd.com/document/421929039/Aprendizaje-Profundo-andrea-pdf*

- Deep autoencoders: redes profundas que permiten reproducir la salida de la misma información que se recibe a la entrada y se usa como herramienta de reducción de dimensión, ya que permite incluir un cuello de botella en la red de la dimensión deseada. En las industrias permitirían generar versiones comprimidas de los procesos. (Hinton, 2006)

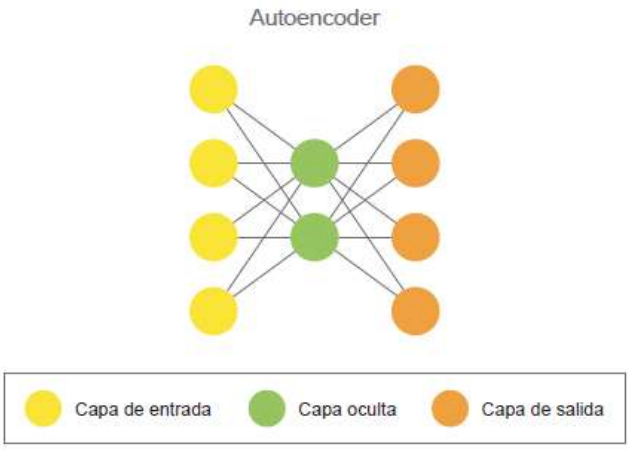


Figura 8. Esquema básico de un autoencoder.
Fuente:*https://es.scribd.com/document/421929039/Aprendizaje-Profundo-andrea-pdf*

1.4 Los datos y la inteligencia artificial

1.4.1 El Big Data

El *Big Data* es un término que describe la ingente cantidad de datos, tanto estructurados como no estructurados, que pueden obtenerse de los equipos o procesos cada día. El gran volumen de datos, su variabilidad, velocidad, veracidad y valor provocan problemas para extraer datos reales y de alta calidad; en ese sentido, los softwares tradicionales no son aparentes para estas exigencias.

El *Big Data* puede generar un gran volumen de datos, sin embargo, uno de los aspectos más importantes es la calidad de los mismos para tomar decisiones que permitan obtener ventajas comparativas; en efecto, si los datos no son de buena calidad se puede incurrir en graves errores. (Power Data, s.f.)

Tareas importantes en esta disciplina son:

- La recopilación y almacenamiento de datos que se debe automatizar,
- El algoritmo que extraiga la información de utilidad lo cual implica un recurso humano, el científico de datos (*data scientist*) cuyo trabajo es extraer conocimiento a partir de los datos para tomar las mejores decisiones.

Erik Brynjolfsson, director del MIT *Initiative on the Digital Economy*, señala que "las empresas que adoptan decisiones basadas en datos logran entre el 5% y el 6% más en productividad y crecimiento en la producción que aquellas que no lo hacen". (ITCL, 2018)

Algunas aplicaciones del *Big Data* pueden ser: por ejemplo: prever crisis económicas, prevenir enfermedades, prever fallas de equipos, trazar la ruta idónea de transporte público o acertar al abrir un local en determinada zona.

Figura 9. Big Data: Algoritmos, tecnología y aplicaciones
Fuente: http://madm.uib.es/wp-content/uploads/2016/06/Jose-Manuel-Benitez-Sanchez-Big-Data-Algoritmos_tecnologia_y_aplicaciones.pdf

1.4.2 Computación en la nube (*Cloud Computing*)

La computación en la nube (o CC, por sus siglas en inglés) es una tecnología que permite acceso remoto a softwares, almacenamiento de archivos y procesamiento de datos a través de un navegador web. El CC se constituye en un factor democratizador y acelerador de la IA, no solo a nivel de desarrollo científico sino, sobre todo de accesibilidad para las empresas y entidades de formación. En efecto, el CC posibilita:

- Disponibilidad de algoritmos ya entrenados para su empleo en soluciones personalizadas
- Acceso a los GPU's (*Graphical Processing Unit*) de amplias capacidades computacionales lo cual permite a cualquiera entrenar sus algoritmos aprovechando la mayor velocidad de trabajo respecto de las máquinas que habitualmente posee una empresa

La IA y la CC se han fusionado para mejorar la calidad de vida de millones de personas. Los asistentes digitales como SIRI, *Google Home* y Alexa de Amazon combinan estas dos tecnologías y muchos usuarios ni se dan cuenta de ello. A nivel empresarial, las capacidades que aporta la IA a través de la CC, ayudan a las empresas a mejorar la administración de sus datos, buscar patrones e información, ofrecer experiencias satisfactorias a sus clientes y optimizar flujos de trabajo. (10Custom, 2020)

1.5 La IA y la industria 4.0

La IA es una disciplina que actúa como motor de cambio y viene marcando la pauta en el avance tecnológico en organizaciones de diversos tamaños. Al existente proceso de automatización, las empresas inevitablemente deberán complementarlo con la digitalización integral de toda su cadena de valor mediante la implantación de nuevos métodos de trabajo, software inteligente, sensores y actuadores.

Hablar de IA en la industria 4.0 implica referirse a:

a) los diversos algoritmos utilizados por *machine learning*, *deep learning*,
b) la visión artificial,
c) los sistemas de reconocimiento de voz,
d) los asistentes virtuales,
e) los *cobots*, etc.

La industria requiere que la IA le agregue valor en términos de:

a) mejora de la productividad,
b) ahorro de tiempo,
c) costos de producción,
d) mayor disponibilidad de los equipos, etc.

En la función mantenimiento, en particular, será el mantenimiento predictivo, que ya utiliza algoritmos para detectar síntomas premonitorios de falla de un activo o proceso, el que apalancará la introducción de la IA para agregar valor.

1.5.1 La industria 4.0 y la hiperconectividad

La hiperconectividad es un concepto que describe la situación actual de la sociedad en la cual la mayoría de sus principales componentes (seres humanos), viven conectados permanentemente a la información mediante diversos dispositivos como el teléfono celular, la internet, la radio, la televisión. También se define como la conectividad permanente entre distintos sistemas y entornos digitales, refiriéndose además a la interacción entre sistemas de información, redes de datos y dispositivos relacionados entre sí a través de internet. (Braun, 2021)

La hiperconectividad se ha convertido en una parte esencial del día a día y permite realizar toda clase de tareas cotidianas. Dentro de la Industria 4.0 la hiperconectividad permite crear un ecosistema digital en la empresa donde todas las áreas involucradas en la creación y elaboración de bienes y/o servicios estén conectados a través de red. (Braun, 2021)

Gran parte de las organizaciones en la actualidad operan en un nivel denominado "Industria 3.0", lo cual significa que sus procesos de producción están conectados dentro de la planta mediante dispositivos de automatización. Por el momento esto sigue siendo suficiente dentro de los márgenes de productividad esperado dentro de las compañías. Sin embargo, con la irrupción de la industria 4.0 será necesario que las empresas implementen un mayor nivel de conectividad para:

- Enlazar todos los procesos dentro y fuera de la planta para agregar valor en términos de productividad, rentabilidad, imagen, confiabilidad.
- Conocer la evolución de los procesos en tiempo real.

Esto ayudará a tomar decisiones rápidas cuando haya oportunidades de mejora. (Braun, 2021)

1.5.2 Tecnologías habilitadoras de la industria 4.0

La industria 4.0, funciona con varios conceptos tecnológicos entre los cuales se incluye:

- el internet industrial de las cosas (IoT, por sus siglas en inglés),
- la robótica colaborativa,
- sistemas ciberfísicos,
- realidad virtual,
- realidad aumentada,
- drones,
- los sistemas ciber físicos,
- fabricación aditiva,

- computación en la nube (*cloud computing*)
- *big data*,
- *block chain*

Los describiremos a continuación.

1.5.2.1 Internet industrial de las cosas (IoT)

El internet de las cosas (IoT, por sus siglas en inglés) permite la conexión de dispositivos mediante sistemas embebidos para comunicarse e interactuar ya sea entre ellos o con dispositivos centralizados. El internet industrial de las cosas (IoT, por sus siglas en inglés) se refiere al uso de la tecnología IoT aplicada al entorno industrial. (Lopez, s.f.)

Cada dispositivo participante en el sistema debe tener una identificación única y la capacidad de intercambiar datos de manera automática y confiable. El conjunto de dispositivos debe estar conectados a la misma red. Todos ellos junto con los sensores, elementos de accionamiento y redes son comúnmente usados en la Industria 4.0.

1.5.2.2 Robótica colaborativa

La robótica es una ciencia que involucra muchas disciplinas. Su objetivo es crear máquinas capaces de hacer o simular comportamientos humanos. (Esneca Business School, 2019)

Los robots colaborativos o *cobots* son robots diseñados y programados para llevar a cabo tareas en colaboración con los trabajadores de la industria 4.0 con la finalidad de liberarles las tareas que son repetitivas y eventualmente peligrosas. Los *cobots* tienen ciertas características que los diferencian de los robots domésticos, a saber: (Esneca Business School, 2019)

- Está diseñado para que sus usuarios puedan programarlo fácilmente
- La programación de sus funciones es regulable, es decir, los operarios pueden ajustar su fuerza y movimientos
- Suelen estar fabricados con materiales ligeros
- Son construidos a base de sensores
- Ayudan a crear productos ajustados a las exigencias de cada cliente.

1.5.2.3 Drones

Un dron es vehículo aéreo no tripulado (VANT) cuyo funcionamiento es básicamente el mismo que el de un avión o un helicóptero. Se ponen en marcha los motores y las aletas se mueven para hacer posible el vuelo. Después con los mandos de control se dirige el vuelo, que dependerá en gran medida de los conocimientos y pericia

del piloto. Algunas veces es posible fijar un rumbo de forma automática. (Universidad Internacional de Valencia, 2018)

Los drones se utilizan para realizar rondas de observación de imágenes que se recogen mediante cámaras de diferente aplicación instaladas en ellos. Con cámaras termográficas instaladas en los drones se puede captar información del estado de las estructuras en las instalaciones mecánicas o eléctricas, esto se enmarca dentro del mantenimiento predictivo o monitoreo de condición. Se puede detectar elementos que tienen fallas y se puede realizar revisiones sin detener la cadena productiva y sin riesgos para las personas. Esto naturalmente trae consigo una serie de beneficios.

También los drones ayudan a la cadena logística ubicando piezas y productos en grandes superficies de almacenaje a través de tecnologías como RFID, la cual es una pequeña etiqueta que se puede adherir a cualquier objeto y que ofrece información sobre su uso o ubicación gracias a unas diminutas antenas que son capaces de transmitir información por radio frecuencia. Sirven además para automatizar tareas en entornos industriales que requieren gran esfuerzo y destreza, reducir costos en desplazamientos de equipos humanos, prevenir fallas antes de que sucedan gracias a los que se conoce como *condition monitoring* y reducir el riesgo por accidente al sustituir operarios por drones en trabajos de alta peligrosidad.

1.5.2.4 Sistemas ciber físicos

Un sistema ciber físico es un ambiente donde existen de manera diferenciada la parte real de un sistema (máquina o equipo) con un modelo virtual funcional del mismo. Entre ambas partes hay interacción. Estos sistemas integran capacidades de computación, almacenamiento y comunicación para controlar e interactuar con uno o más procesos físicos.

Estos sistemas están conectados entre sí y también con servicios remotos de almacenamiento *(cloud computing*) y gestión de datos. La mayor interconexión, adaptabilidad y mayor velocidad de intercambio de información y generación de autocomprobación y auto configuración permite optimizar en todo momento el rendimiento del sistema impulsando la rentabilidad y el mejoramiento de los procesos en la industria. (Factoría del Futuro, 2020)

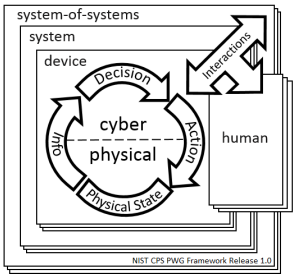


Figura 10. CPS Modo Conceptual
Fuente: https://pages.nist.gov/cpspwg/

Tienen variadas aplicaciones como vemos en la Figura 11 se tiene el equipo real (una aeronave) y en la tableta las diferentes partes del mismo con la finalidad de monitorearlo o monitorear su reparación.

Figura 11. Aeronave cuya reparación se monitoreará
Fuente: Cortesía de Siemens

1.5.2.5 Realidad virtual

La tecnología de la realidad virtual recrea un espacio tridimensional virtual inmersivo, similar a uno real. Si una persona se coloca unas gafas de realidad virtual, se priva totalmente del mundo real que tiene alrededor y puede interactuar dentro de ese mundo virtual mediante dispositivos adecuados (guantes, mandos) que permiten:

- Agarrar objetos,
- Abrir puertas, etc.
- Realizar actividades de mantenimiento óptimas

Permitiendo realizar simulaciones de la realidad como:

- maniobras normales o arriesgadas,
- visitas virtuales a instalaciones industriales,
- simulacros de actuación en casos de emergencia,

Donde que la persona tiene una sensación de presencia real consiguiéndose un aprendizaje mental y locomotriz mucho más eficiente que un curso convencional con videos y manuales. Esto la sitúa como una herramienta de entrenamiento/aprendizaje para trabajadores sin tener que parar la producción. (3R. Industria 4.0, s.f.)

Por otro lado, la realidad virtual es la mejor herramienta para reducir los costes derivados en las tareas de prototipado dado que crea una simulación casi real del futuro producto donde se visualizarán todas sus características como si se le tuviera físicamente delante de los ojos.

1.5.2.6 Realidad aumentada

Es aquella tecnología que permite superponer imágenes generadas digitalmente (computadoras, tabletas, teléfonos celulares, videojuegos, etc) sobre elementos de en entorno físico real, maximizando y enriqueciendo la información que obtenemos de dicho

entorno. Estos dispositivos están hiperconectados a una red protegida, la cual se encarga de gestionar el cruce de datos, alarmas, mensajes, videoconferencias y demás información.

La documentación de los componentes de una instalación es accesible desde la plataforma una vez que ha sido identificado el equipo o componente gracias a la conexión con la fuente de datos del cliente.

Se pude realizar simulaciones en realidad aumentada sobre los equipos de planta, de tal forma que el operario puede accionar determinados mecanismos o cambiar ciertas variables del equipo y se visualizará una simulación 3D sobre el equipo. (Idea Ingeniería, s.f.)

1.5.2.7 Fabricación aditiva

La Fabricación aditiva es un nuevo concepto de producción mediante el cual los materiales (poliméricos, compuestos, metálicos) se depositan capa a capa de una forma controlada generando piezas de diferente y compleja geometría que se adapten mejor a las necesidades de las empresas.

Gracias al diseño asistido por el ordenador o un software de modelado y animación, es posible crear una guía de impresión, a partir de la cual, la máquina va construyendo por capas hasta finalizar el modelado.

Esta fabricación posee un alto potencial para todo tipo de aplicaciones. No solo se contempla la fabricación de piezas intermedias o finales, sino que se puede obtener un gran avance para diversas tareas como realizar planos, maquetas o crear alimentos procesados.

Las ventajas que podemos señalar son (Grupo IGN, 2019):

- Complementa otras técnicas de fabricación tradicionales,
- Permite fabricar piezas de alta complejidad geométrica o formas personalizadas,
- Crear montajes sencillos
- No produce desechos (cuidado del medio ambiente),
- Permite reducir los costes
- Facilita el proceso de diseño
- Producir de forma más sostenible.

1.5.2.8 Cloud Computing

El gran avance de tecnologías de la información ha permitido que el *Cloud Computing* se haya posicionado como una tecnología emergente dentro de la industria 4.0, debido a que ha logrado independizar el hardware del software, permitiendo que grandes

compañías del rubro industrial puedan tener un acceso ubicuo y baja demanda de muchos de los principales servicios que necesitan para funcionar.

Los principales modelos de servicio de *Cloud Computing* en la Industria son:

- IaaS (*Infrastructure as a Service*): Se refiere a la provisión de los sistemas físicos (hardware). Este modelo de servicio abarca diversas dimensiones como el acceso a determinados servidores o la capacidad computacional a la hora de analizar ciertos datos de una gran magnitud. Una característica especial es que los usuarios finales tienen un control total sobre la infraestructura contratada.
- PaaS (*Plataform as a Service*): este otro modelo de servicio del *Cloud Computing* en la Industria 4.0 hace referencia a los entornos de desarrollo desde que los programadores crean, analizan y llevan a cabo las aplicaciones informáticas.
- SaaS (*Software as a Service*): Se encarga de proveer aplicaciones finales a las empresas que lo contratan mediante internet, ya que estás empresas se encuentran alojadas en la infraestructura del proveedor. Se accede a los programas SaaS a través de navegadores web.

Existen diversos tipos de *Cloud* en la Industria 4.0:

- *Cloud* Pública: Presenta la gran ventaja de ofrecer una gran escalabilidad en la empresa, esto quiere decir que la compañía puede utilizar el servicio de Cloud dependiendo de sus propias necesidades, lo cual hace una opción mucho más flexible que las nubes privadas.
- *Cloud* privada: Brinda una gran seguridad, más que todo en aquellas empresas del sector industrial que necesiten tener a salvo todos sus datos.
- *Cloud* híbrida: Las empresas pueden escalar el servicio conforme a sus necesidades, pero manteniendo la seguridad de la *Cloud* Privada.

1.5.2.9 Big Data

Big data es un proceso que analiza e interpreta ingentes cantidades de datos estructurados y no estructurados. Sirve para que los datos almacenados de forma remota puedan ser utilizados por las organizaciones para una informada toma de decisiones.

La información extraída de tales datos sirve para mejorar procesos de operación, estrategias de mantenimiento, incrementar la competitividad, encontrar patrones de comportamiento, entre otras aplicaciones.

La interconexión entre sistemas y ordenadores complementado con la capacidad de recopilación y análisis de grandes cantidades de datos ha hecho posible que existan máquinas inteligentes que puedan tomar decisiones por sí mismas ayudando a mejorar las operaciones en las organizaciones. El *Big Data* ayuda, entre otras cosas, a:

- Mejorar los procesos en almacén: Debido a los sensores y dispositivos portátiles, diversas empresas pueden mejorar la eficiencia operativa al detectar errores humanos, hacer controles de calidad y mostrar rutas óptimas de producción o montaje.
- Eliminar cuellos de botella: Se identifican las variables que puedan afectar el rendimiento sin costo adicional, ayudando así a los fabricantes a identificar el problema.
- Predecir en base a datos históricos recopilados por sensores situaciones anómalas relativas por ejemplo a: comportamiento de mercado, intervención antes de una falla catastrófica para tomar decisiones estratégicas como planes de expansión, etc.

1.5.2.10 Block Chain

El *Blockchain* (Cadena de bloques) es una tecnología o protocolo informático basada en estructura matemática que permite el almacenamiento de información, seguro, anónimo. descentralizado y casi imposible de falsificar.

Los bloques son ficheros (normalmente de texto) que contiene la información que se quiere guardar. Forman una cadena porque cada uno de esos bloques tiene algo de información sobre el bloque anterior. A la información que cada bloque tiene sobre el anterior se le llama HASH y es una especie de número de serie importante para el que se necesita matemática

El 2do ingrediente del BCh es la red de ordenadores, todos los ordenadores de la red guardan una copia de la cadena de bloques lo cual hace muy difícil que se pierda.

La tecnología *Blockchain* tiene la capacidad de mejorar procesos existentes e impactar en la modificación de producción de bienes. Permite mejorar, por ejemplo, la trazabilidad, seguridad, velocidad y transparencia en cualquier sistema.

El *Blockchain* potenciará las capacidades de la Industria 4.0 y tendrá el siguiente impacto:

- Optimización de recursos.
- Garantiza la trazabilidad de un producto.
- Da lugar a nuevos modelos de negocios, en efecto, garantiza intercambios de datos seguros, transparentes y fiables.
- Se conseguirán máquinas autónomas, que serán capaces de autogestionarse, negociar entre ellas y poder así llegar a acuerdos, coordinando con proveedores y clientes de la logística de entrega del producto, tomando decisiones sobre el suministro de materias primas, haciendo pedidos de forma automática, siempre y cuando se den las condiciones establecidas previamente.

UNIVERSITAS STUDIORUM PIURENSIS

Capítulo 2
Mantenimiento industrial

2.1 Evolución histórica del mantenimiento industrial

Desde la aparición del ser humano en la tierra los activos han servido para su supervivencia. Los activos de los hombres prehistóricos eran su lanza, su cueva lo que les ayudaba a cazar y sobrevivir. A estos activos se les hacía mantenimiento correctivo. Esta estrategia correctiva se mantuvo hasta inicios del siglo XIX.

A mediados del siglo XX, la tecnología permitió que se desarrollaran máquinas dentro de una cadena productiva compleja, ya que se había intensificado la producción a gran escala, así como el formato de línea de producción. El rendimiento de las máquinas adquirió importancia para agilizar la producción, y es debido a esta situación, que nace la aplicación del mantenimiento preventivo en la industria.

Alrededor de la década de 1960, el sector de mantenimiento se limitaba a ciertos campos como la electricidad, mecánica, cambio de piezas, lubricación o engrase. El mantenimiento se basaba en la prevención y corrección, no existía una noción de predicción. Se tenía programada la rutina del operario cada cierto tiempo, pero no se tomaba en cuenta las condiciones de operación de cada equipo.

Las actividades preventivas que se realizaban en una gran parte de los equipos o instalaciones no tenían mejoras sustanciales. La irrupción de la electrónica y microelectrónica abre las puertas al mantenimiento predictivo con el cual se puede monitorear con indicadores, sensores y dispositivos cada equipo, así predecir una avería o un mal funcionamiento.

Simultáneamente, en los años 1960 estaba implantándose en Japón una filosofía de mantenimiento que revolucionó la industria mundial, esta fue el TPM (*Total Productive Maintenance*). Surgió bajo la necesidad de realizar un mantenimiento menos costoso que el Mantenimiento Preventivo y la solución fue basada en el concepto del mantenimiento autónomo, el cual tiene sus bases en la capacitación de operarios para realizar las tareas básicas de mantenimiento. La automatización fue de gran ayuda en esta práctica, ya que posibilitaba que los operarios pudiesen leer indicadores y monitorear sus equipos.

En los años 80 del siglo XX en Estados Unidos se documentó, gracias a la metodología predictiva, el RCM (*Reliability Centered Maintenance*) o Mantenimiento Centrado en la Confiabilidad. La práctica se inició en vista a la gran cantidad de accidentes que sufría la aviación comercial en la década de los años 60 en que dos tercios de los accidentes eran causados por fallas en los equipos.

Esa problemática llevo a repensar el programa de mantenimiento que se estaba aplicando hasta ese entonces en la industria de aviación y gracias a ello es que luego de 20 años de investigación se culmina un reporte que presentó al método RCM como solución, puesto que, su objetivo era determinar lo que debía hacerse con la finalidad de asegurar que un elemento físico continúe desempeñando las actividades que se esperaban de él en un contexto determinado.

Alrededor de los años 2000, el mundo industrial cambió de diversas maneras, pero la revolución informática impactó especialmente en la producción, la calidad y el flujo de trabajo. Desde ese entonces inició la adaptación a las nuevas tecnologías y empezó la modernización del mantenimiento industrial. Aparecen los *softwares* de mantenimiento para gestionar los equipos industriales (asignar tareas, realizar monitoreos y diagnóstico de fallas, identificar las prácticas óptimas, mejorar el desempeño a futuro, controlar el cumplimiento de tareas asignadas, registrar el historial de fallas y generar informes). (Mancuzo, 2020)

En la actualidad estamos en la era de la digitalización (Industria 4.0) y el mantenimiento debe adecuarse a este nuevo escenario (cuarta revolución industrial) utilizando todas las tecnologías habilitadoras de la industria 4.0. Este es el reto y es lo que se describirá en el presente trabajo.

2.2 Estrategias del mantenimiento

2.2.1 El mantenimiento correctivo

Es aquel que se realiza después de que ocurre una falla. Se clasifica en mantenimiento correctivo programado y mantenimiento correctivo de emergencia.

El mantenimiento correctivo tiene sus ventajas y desventajas. La ventaja más importante de seguir esta estrategia es la máxima utilización de la vida de la máquina o componente. Por su parte, las desventajas más resaltantes son:

- Una falla puede tener efectos colaterales, inclusive la detención de la producción.
- Dependiendo de la criticidad de la máquina o componente, puede ser costoso.

Dado que tarde o temprano una máquina puede fallar, la organización debe estar preparada para gestionar las fallas en los siguientes aspectos:

- Proveer al personal correspondiente las capacidades necesarias para diagnosticar fallas y efectuar reparaciones en tiempo y forma.
- Elaborar procedimientos y hojas de trabajo fáciles de entender.

- Realizar una adecuada planificación de las tareas para reducir el tiempo medio de reparación (MTTR) y optimizar los recursos.
- Identificar los peligros y evaluar riesgos en los trabajos.
- Analizar la causa raíz de las fallas, de modo de eliminar o reducir su recurrencia.
- Una adecuada planificación del mantenimiento correctivo deberá contemplar:
- Definición del alcance de los trabajos en base al diagnóstico establecido.
- Elaboración de procedimientos y hojas de trabajo.
- Selección de las herramientas, repuestos e insumos necesarios.
- Definición del personal técnico especializado.
- Planificar turnos, grupos de trabajo, refrigerios, descanso y la secuencia de tareas y subtareas.
- Resumir las tarjetas de seguridad, órdenes de trabajo, hojas de trabajo, instructivos, etc.
- Reconocer los estándares de aceptación para la entrada en servicio del equipo.
- Establecer las rutinas de prueba, puesta en marcha y verificación de funcionamiento.
- Confeccionar un diagrama de Gantt si la envergadura del trabajo lo justifica.

En todo tipo de intervención correctiva hay planificación. Cuando el trabajo es sencillo y requiere pocos recursos se puede planificar mentalmente, sin embargo, cuando la tarea es compleja, demanda un tiempo de ejecución elevado y requiere de la participación de diversas especialidades, la intervención debe considerarse como un proyecto y por tanto planificarse como tal.

2.2.1.1 El mantenimiento correctivo programado

Una intervención de mantenimiento correctivo programado se activa debido a:

- observación del operador,
- una inspección de rutina o
- la aparición de un síntoma premonitorio de falla

Poder programar la intervención permite:

- planificarla,
- seleccionar el momento más oportuno,
- recabar los repuestos, la mano de obra y las facilidades necesarias, de este modo se afectará lo menos posible la producción.

2.2.1.2 El mantenimiento correctivo de emergencia

El mantenimiento correctivo de emergencia se realiza cuando ocurre una falla imprevista que detiene la producción o puede ocasionar accidentes o afectar la salud de los trabajadores o una contaminación ambiental que impone la necesidad de reparar el equipo cuanto antes. En resumen, este tipo de mantenimiento implica que se lleve a cabo una reparación rápida, para evitar daños materiales, humanos y pérdidas económicas. (Gestión del mantenimiento cod 7969, s.f.)

2.2.2 El mantenimiento preventivo

El mantenimiento preventivo consiste en reparar un ítem o reemplazar sus componentes en forma periódica, sin importar su estad de condición al iniciar la intervención, y bajo la hipótesis de que el patrón de fallas que rige su comportamiento tiene un periodo de vida útil conocido, alcanzando finalmente los mismos niveles de confiabilidad y calidad originales. (Pistarelli, 2010)

Las actividades de mantenimiento preventivo suelen contemplar lo siguiente:

- Reemplazo (sustitución) de equipos, subconjuntos, componentes o piezas cuando muestren signos de desgaste, corrosión, erosión o fatiga, lo que lleva a que se incremente la probabilidad de falla. El reemplazo busca restituir al equipo a su estado original.
- Conservación, revisión o restauración de ítems para llevarlos a su estado de condición básica inicial.
- Rutinas de inspección y chequeo de recorrida para detectar condiciones anómalas para corregirlas. Tiene altos beneficios y un costo de realización muy bajo.
- Limpieza, Ajuste y Lubricación: Ciertos activos requieren acciones de conservación para mantenerlos dentro de cierto estado de condición básica. Se lleva a cabo mediante rutinas. Aunque estas actividades no cambian la propensión a la falla tiene beneficios extraordinarios y es de muy bajo costo.
- Calibración: Se realiza cierto ajuste de parámetros en instrumentos del proceso. Esta acción contempla medir, controlar y ajustar los parámetros de proceso de acuerdo a los patrones certificados.

El mantenimiento preventivo persigue los siguientes objetivos:

- Aumentar la disponibilidad de las máquinas al disminuir las detenciones no programadas.
- Minimizar averías imprevistas de los equipos.
- Mejorar el aprovechamiento de mano de obra por medio de la programación de tareas.
- Mejorar la calidad de productos y servicios.
- Disminuir el riesgo para el personal en las operaciones de producción y mantenimiento.
- Minimizar los gastos debido a las reparaciones de emergencia.
- Disminuir el impacto ambiental.

La intervención preventiva debe realizarse en base a criterios técnicos y económicos. Si se realiza con mayor frecuencia de la necesaria se puede inducir errores humanos (ajustar un perno sin el torque adecuado, etc.) que en lugar de conseguir un mejor desempeño de la máquina la deja en mal estado. Por otro lado, la intervención implica un costo que debe justificarse con la mejora del desempeño de la máquina.

El mantenimiento preventivo se puede clasificar en:

a. basado en el tiempo/uso (horas de operación de un equipo, kilómetros recorridos de un vehículo, ciclos de trabajo de una válvula, cantidad de producción de un compresor de gas, etc.) o
b. basado en la condición (pérdida de presión, elevación de temperatura, incremento de vibraciones, incremento de la corriente, etc).

2.2.2.1 Mantenimiento preventivo basado en tiempo/uso

El mantenimiento basado en el tiempo (MBT), consiste en reemplazar las piezas de equipos de acuerdo con los intervalos establecidos en base a las recomendaciones del productor o a la experiencia de los técnicos en donde se define cada cuánto tiempo las piezas deben ser cambiadas antes de que se averíen.

2.2.2.2 Mantenimiento preventivo basado en condición

El mantenimiento basado en la condición (MBC), consiste en reemplazar las piezas de equipos en función de los resultados de un examen realizado a ciertos intervalos con dispositivos que relevan ciertas condiciones como, por ejemplo:

i. Niveles de vibración
ii. Niveles de sonido
iii. Estado del lubricante
iv. Fugas

que definirán cuando se debe intervenir para evitar averías.

2.2.3 El mantenimiento proactivo

Mediante El mantenimiento proactivo es una estrategia enfocada en la identificación y corrección de las causas que originan las fallas en equipos, componentes e instalaciones industriales. Esta técnica implementa soluciones que atacan la causa raíz que puede provocar fallas. (TMV)

Beneficios del mantenimiento Proactivo:

- Permite analizar la evolución de una falla sintomática reversible prácticamente desde la aparición de la causa raíz, debido a su sensibilidad de detección prematura.
- Provee información para trabajar sobre las causas y no efectos de los fallos.
- Reduce los gastos provocados por las fallas sintomáticas irreversibles.
- Ofrece el panorama muy acotado del estado de los componentes para decidir el momento más oportuno de su reemplazo o reparación.

- Prolonga la vida de equipos en los cuales la causa raíz hubieran generado una falla sintomática irreversible. (Pistarelli, 2010)

2.3 Metodologías de optimización del mantenimiento

La existencia de las estrategias de mantenimiento correctiva y preventiva si bien es cierto son necesarias, no son suficientes para garantizar la continuidad de funcionamiento de los activos deben ser reforzadas mediante la aplicación de ciertas metodologías como la del mantenimiento centrado en la confiabilidad (*Reliability, Centered Maintenance*-RCM, por sus siglas en Inglés) y el mantenimiento productivo total (*Total Productive Maintenance*-TPM, por sus siglas en Inglés) las que describiremos brevemente.

2.3.1 Mantenimiento centrado en la confiabilidad (RCM)

El mantenimiento centrado en la confiabilidad (en adelante RCM, *Reliability Centered Maintenance)*, aparece para encontrar respuestas a las fallas que ocurrían en los activos a pesar de que a estos se les aplicara escrupulosamente las tareas tradicionales de mantenimiento preventivo.

Se puede definir como un método estructurado, deductivo y participativo que define la estrategia de mantenimiento apropiada para cada equipo actuando en su contexto operativo real.

Se puede decir que una estrategia apropiada es una agrupación de tareas capaz de evitar que sucedan modos de falla o reducir drásticamente sus consecuencias de manera eficiente.

La metodología RCM sigue u procedimiento:

I. Conformar un equipo multidisciplinario de trabajo.
II. Jerarquizar los activos
III. Identificar las funciones de los activos
IV. Deducir las fallas funcionales y modos de falla,
V. Identificar las causas de tales fallas y
VI. Deducir las consecuencias de las fallas
VII. Elaborar un diagrama de decisión

Del diagrama de decisión se derivan:

a) tareas preventivas (reemplazo o sustitución programada, rutinas de lubricación ajuste y limpieza),
b) tareas predictivas (detección de fallas incipientes, fallas ocultas) o
c) tareas proactivas
d) rediseño del proceso.

El objetivo es mejorar la confiabilidad operacional.

Los modos de falla cuyas consecuencias son capaces de afectar la seguridad de las personas o el medio ambiente tienen prioridad en esta metodología. En estos casos se propone alternativas que reducen el riesgo de accidente y aseguran un desarrollo sustentable. Se trata de un proceso de mejora de carácter proactivo, porque no es preciso que sucedan las fallas para empezar a prevenirlas. (Pistarelli, 2010)

El RCM no solo busca resolver problemas de confiabilidad, va mucho más allá y además introduce un cambio cultural en los conceptos a tener en cuenta al momento de pensar en mantenimiento. Introduce la idea de que el mantenimiento tiene como finalidad asegurar que las funciones de un sistema en su contexto operativo sin concentrarse únicamente en una parte del mismo como objeto aislado.

Los diversos beneficios de usar RCM son:

- Preservar las funciones del sistema productivo teniendo en cuenta su contexto operativo.
- Establecer rutinas de mantenimiento más efectivas para el activo buscando hacer exclusivamente lo necesario.
- Eliminar el riesgo de accidentes o reducirlo a niveles tolerables y satisfacer normativas, leyes o reglamentaciones vigentes referidas al cuidado del medio ambiente.
- Aumentar confiabilidad operacional.
- Mejorar la calidad de los productos elaborados o de los servicios prestados.
- Satisfacer a los usuarios del equipo o a los clientes de servicio.
- Promover el trabajo en equipo y el espíritu participativo de los colaboradores.
- Crear un lenguaje común en el ámbito del mantenimiento y de la planta.
- Impulsar rediseños para evitar o predecir aquellos modos de falla que puedan amenazar la seguridad de las personas, el medio ambiente o producción de activos físicos.

2.3.2 Mantenimiento productivo total (TPM)

También conocido como *Total Productive Maintenance*. Este tipo de mantenimiento se desarrolló en Japón, específicamente en el área automotriz, con la finalidad de mejorar los procesos productivos y alcanzar altos niveles de calidad.

La implementación de TPM, va más allá del área de mantenimiento, en efecto, impulsa la participación de todos los niveles y sectores de la organización, se trata de gestionar a la empresa como un TODO, por lo que todas las personas que la conforman deben adquirir el compromiso de apoyarlo.

El objetivo del TPM es aumentar la eficacia integral de los equipos y con ello la competitividad y flexibilidad de la empresa. Entre otras cosas persigue: eliminar borrar los defectos, ya sean físicos y metodológicos, para alcanzar calidad y eficiencia supremos.

El TPM necesita que una persona asuma diversos compromisos y pueda lograr una evolución mental de enfrentar el trabajo. Debe existir un cambio cultural desde las más

altas esferas de la empresa hasta el colaborador de menos rango, donde la alta jerarquía debe comprometerse con el proyecto y poder transmitir sus expectativas a los niveles inferiores.

La visión del TPM se traduce en:

- cero tasas de incidentes.
- cero tasas de defectos.
- cero tasas de averías, fallas y desviaciones.
- cero pérdidas de índole.

Una organización que pretenda aplicar la metodología TPM, necesita conducir los esfuerzos con procesos que articulen los diversos pilares que comprende, estos son.

- mantenimiento autónomo.
- mantenimiento planeado
- mejora enfocada
- gestión temprana o inicial.
- mantenimiento para la calidad.
- capacitación y Desarrollo.
- gestión de los sectores administrativos.
- higiene, Salud y Medio Ambiente.

2.3.3 Optimización del mantenimiento preventivo (*Preventive Maintenance Optimized*)

El método de optimización de mantenimiento preventivo (PMO por sus siglas en inglés) ha sido concebido para revisar los requerimientos actuales de mantenimiento, el historial de fallas y además la información técnica de los activos en operación y proponer mejoras en estos aspectos es decir realizar aquellas tareas que realmente agreguen valor. Esta metodología no es aplicable solo en la fase de diseño del equipo sino para activos en operación y tiene como finalidad que la organización del mantenimiento no caiga en el círculo vicioso del mantenimiento correctivo (ver Figura 12)

Figura 12. Ciclo Reactivo del Mantenimiento
Fuente: https://www.researchgate.net/publication/320540199

La metodología PMO, reconociendo la importancia de la función del sistema, toma como punto de partida el plan de mantenimiento elaborado para: determinar las tareas y frecuencias actuales de mantenimiento. La implementación exige experiencia y conocimiento técnico del personal de planta, para realizar mejoras en distintos aspectos como: la determinación de nuevas tareas y frecuencias, la asignación de recursos, todo ello para mejorar la eficacia de los planes de mantenimiento.

La fuerza fundamental del PMO se centra en incorporar acciones de mantenimiento que agreguen valor. Steve Turner recomienda una serie de pasos para la implementación de PMO:

- Establecer funciones y tareas.
- Analizar los modos de falla.
- Racionalizar y revisar los procedimientos.
- Análisis funcional basado en confiabilidad.
- Evaluar las consecuencias.
- Determinar las políticas de mantenimiento.
- Agrupar y revisar los procesos funcionales.
- Aprobar e implementar los programas.
- Programa de Vida y Mejoramiento Continuo.

La implementación del PMO se basa en la criticidad de los activos físicos y sistemas que se encuentran en la planta. La criticidad se puede obtener jerarquizando los equipos. Existen diversos criterios para la criticidad entre ellos el impacto en la producción, la seguridad, el medio ambiente, los costos. Cada uno de ellos impacta en el cumplimiento de los objetivos estratégicos de la organización. La prioridad en la aplicación de la metodología depende de la criticidad.

En resumen, el PMO se encarga de analizar los programas de mantenimiento de manera muy detallada y específica, lo cual permite que se lleve a cabo un control minucioso de todas las actividades realizadas en un equipo. Es así que el proceso de implementación del PMO transforma la cultura de una empresa reactiva a una cultura dinámica que aumente la confiabilidad. (Ponce-Mostacero, 2018)

2.3.4 Mantenimiento basado en riesgo (RBM)

En toda tarea de mantenimiento existe riesgos implícitos por esta razón aparece la metodología de mantenimiento basado en el riesgo, que busca reducirlo o eliminarlo para evitar incidentes, daños a la salud o al medio ambiente. Esto es mucho más importante en aquellas pantas de alto riesgo tales como las de hidrocarburos, nucleares, petroquímicas). La definición de riesgo contempla dos aspectos: la probabilidad de ocurrencia del evento (en nuestro caso, la falla) y las consecuencias de la falla.

El mantenimiento basado en riesgo puede tener impactos en:

- Planificación y Gestión de activos: cuando se aplica una jerarquización de equipos basada en riesgos conjuntamente con una definición clara de los compromisos de producción a corto, mediano y largo plazo, la planificación aporta valor al negocio.
- Organización: cuando se aplica una estrategia de mantenimiento basada en riesgo no se debe descuidar el ciclo de mantenimiento, es decir, identificación de tareas, planificación, programación, ejecución y cierre.
- Actividades del ciclo de vida: una buena gestión de los equipos potencia la eficacia en las diferentes etapas del ciclo de vida del activo (diseño, fabricación, instalación, operación, mantenimiento y enajenación) para la selección o renovación.
- Conocimiento de los Activos: el desconocimiento o la incertidumbre sobre la condición de los activos, es un elemento potenciador del riesgo y al mismo tiempo es una de las causas de mayor destrucción de valor en una organización.
- Revisión y Riesgo: el proceso de gestión de riesgo debe ser una parte integrante de la gestión del negocio, debe integrarse en la cultura y en las prácticas del quehacer diario y adaptarse a los procesos de negocio de la organización. (Medina Nuñez, 2018)

Aplicar el mantenimiento basado en riesgos tiene los siguientes beneficios:

- Evita la falla catastrófica de un equipo crítico.
- Permite modelar los mecanismos de deterioro de los equipos.
- Permite direccionar los recursos hacia los equipos claves del proceso.
- Permite definir los puntos de monitoreo de condición de los equipos.
- Mejorar el proceso de planeación de las campañas de inspección.
- Reduce el costo de inspecciones a lo largo de1 ciclo de vida del activo.
- Se alcanzan reducciones significativas de costos asociados al alcance de las paradas de plantas programadas.
- Reducción de un alto porcentaje del número de equipos a intervenir en una parada programada para mantenimiento.
- Extensión de los intervalos entre paradas de plantas para mantenimiento.
- Reducción del tiempo de duración de una parada de planta para mantenimiento.

El mantenimiento basado en riesgo es una gestión más completa que el mantenimiento basado en condición, pues agrega el riesgo calculando la probabilidad de falla del componente y estimando la consecuencia de la falla. (Medina Nuñez, 2018)

Capítulo 3
El mantenimiento y la inteligencia artificial

3.1 El mantenimiento en la industria 4.0

En la actualidad gracias a las nuevas tecnologías que se han desarrollado tanto en sensorización, comunicaciones, almacenamiento de datos y potenciamiento de cálculos existen nuevas mejoras para el servicio de mantenimiento de los activos.

La estrategia de mantenimiento que por el momento se acomoda más con la IA es el predictivo, en efecto, con éste se analizan datos operacionales que ayudan a determinar la condición del activo para poder predecir cuándo se realiza el mantenimiento respectivo.

Todo mantenedor tiene su visión y la más alta es que sus activos estén siempre disponibles. La indisponibilidad de un activo afecta la planificación de la producción, con todos los efectos negativos colaterales que trae consigo. La avería de un activo crítico afecta el desempeño de la planta, y puede tener consecuencias indeseables que incluso pueden llegar a comprometer su existencia.

Una de las opciones para mejorar la gestión de mantenimiento es la recopilación de datos de ciertas variables en los equipos. Esto comienza con la instalación de sensores en puntos estratégicos. El análisis de los datos recopilados por los sensores brinda información de calidad para tomar decisiones asertivas en cuanto a intervenciones de mantenimiento. Si esos datos son procesados mediante algoritmos convenientemente entrenados, pueden ayudar a detectar fallas y alertar sobre su presencia. Esto finalmente conlleva a una mejora de la competitividad sin una gran inversión adicional.

La metodología más general que se lleva a cabo en el mantenimiento cuando va de la mano con la IA en la Industria 4.0 es la siguiente:

- Primero, planifica la estrategia de mantenimiento predictivo en un activo determinado.
- Se le añade, la monitorización, a través de sensores o dispositivos IoT.
- La información obtenida de los sensores define si un activo tiene sus diversos parámetros correctos. Aquí se une el conocimiento del activo y mantenimiento predictivo.
- Si los parámetros están fuera de estándar los diferentes algoritmos comunicarán la variación del estado del activo podrán identificar lo que está pasando.

- Posteriormente, se activa el proceso de creación de alertas y avisos acerca de la probabilidad de una falla o anomalía en el sistema.
- Finalmente, programar una intervención en el activo a través de la generación automática de órdenes de trabajo y optimización de los planes de mantenimiento. (Corredera, 2019)

3.2 Apoyo de las tecnologías habilitadoras al mantenimiento

En esta sección se explica como el mantenimiento evoluciona a lo largo del tiempo. adecuando sus estrategias en función del avance tecnológico. En la actualidad, el reto es adecuar tales estrategias con la IA y el trabajador de mantenimiento también debe evolucionar adicionando a su conocimiento técnico otras habilidades como las de aprendizaje continuo, trabajo en equipo y reacción rápida a los cambios, es decir, convertirse en un trabajador 4.0

Para tener claridad acerca de la función mantenimiento en las actuales circunstancias se describirán como las diversas tecnologías habilitadoras de la IA (Internet de las cosas, robótica colaborativa, drones, sistemas ciber-físicos, realidad virtual, realidad aumentada, entre otras) pueden adecuarse al Mantenimiento.

3.2.1 Internet industrial de las cosas

El término internet de las cosas (IoT, por sus siglas en inglés) es tan reciente que aún no tiene una clara definición y probablemente hay personas que no comprenden qué es concretamente, sin embargo, se puede decir de manera sencilla que se refiere a un inmenso número de objetos físicos (dispositivos) que se conectan a internet (vía cable, wifi, bluetooth, etc) para compartir datos con otros objetos físicos, haciendo su trabajo sin intervención humana.

Algunos de los objetos físicos son de uso cotidiano en los seres humanos como los teléfonos móviles y otros de uso en entonos industriales para monitorear variables en máquinas (temperatura, vibración, consumo energético, etc), detectar patrones, hacer recomendaciones y de esa manera ayudar a tomar decisiones, entre ellas, decisiones acerca de la gestión de su mantenimiento. Cuando esta interconexión se realiza en equipos industriales el término cambia y se convierte en internet industrial de las cosas (IoT, por sus siglas en inglés).

Esto parte necesariamente de la instalación de sensores en los dispositivos, los cuales generan una gran cantidad de datos que, recolectados, procesados y analizados proveen información relevante para la toma de decisiones, por ejemplo, si hay algún síntoma premonitorio de falla en una máquina, generar un aviso que sea notificado a las personas correspondientes, generar pedidos de repuestos para intervenirla a tiempo. Todo este proceso puede ser acompañado desde un dispositivo móvil. Esto sería el mundo del mantenimiento 4.0, los beneficios pueden ser muchos.

3.2.2 Robótica colaborativa

Tradicionalmente, los robots industriales eran máquinas enjauladas de grandes dimensiones con las que los humanos no podían interactuar, ya que eran pesadas, no tenían consciencia de su entorno y suponían un peligro para la seguridad. Una evolución han sido los robots colaborativos (*cobots*) los cuales pueden trabajar junto a los humanos para realizar ciertas tareas. (Automated, 2018)

Los *cobots* son mucho más fáciles de programar y se adaptan a trabajar con las personas. Vienen equipados con funciones de seguridad que les permiten detenerse o ralentizar su velocidad cuando hay un ser humano cerca, esto reduce el riesgo de colisión o peligro de seguridad. Las funciones integradas de los *cobots* son las que hacen factible la ayuda a los técnicos al momento de reducir el riesgo de que se produzcan periodos de inactividad imprevistos.

Se han diseñado *cobots* para realizar tareas repetitivas y complejas con el objeto de reducir la fatiga de los humanos; por otro lado, pueden realizar tareas realizadas en entornos peligrosos, por ejemplo, en un equipo sobrecalentado.

Gracias a la IA y el aprendizaje automático los *cobots* pueden aprender a medida que desarrollan su trabajo y eso les ha permitido desarrollar la capacidad de tomar sus propias decisiones, alertando a los humanos de cualquier posible avería. Esto los hace muy adecuados para las labores de mantenimiento de activos antes de que la avería repercuta negativamente en la producción. Es una forma de mejorar el mantenimiento preventivo en la fábrica.

Actualmente se ha venido trabajando en el desarrollo del ARMAR-6, un robot humanoide y autónomo que puede ayudar a reducir el tiempo empleado en las tareas de mantenimiento en la fábrica. Está conformado por un sistema de tres cámaras para detectar y reconocer humanos, identificar la voz para poder entender y procesar las órdenes, y manos con pinzas para coger objetos.

Se aspira a que en el futuro el robot sea capaz de determinar las intenciones del técnico para poder intervenir en el momento más oportuno; sería una gran ayuda para los técnicos en las labores de mantenimiento.

3.2.3 Drones

El uso de drones es fundamental en el mantenimiento sobre todo en labores de inspección en lugares inaccesibles, además evita que trabajadores tengan que acceder lugares peligrosos, por ejemplo, inspección en líneas eléctricas energizadas, en grandes alturas, etc.

Los drones brindan un acceso más fácil a los datos, menor costo y riesgo, y la posibilidad de documentar las condiciones de los activos en una forma automatizada. Conforme progresa la tecnología en el uso de drones se puede gestionar con facilidad las instalaciones, infraestructuras y activos.

Las principales ventajas del uso de drones son:

- Ahorro de costos.
- Reducción de los accidentes.
- Mejora de los servicios al incrementar la fiabilidad de los equipos.

La información recolectada por los drones puede ser compartida con un software de gestión de mantenimiento que incluya registros históricos de estados previos, normas de mantenimiento y diversas especificaciones junto con instrucciones de reparación, diagramas, información de garantías y demás datos para: monitorear las condiciones de los activos y mejorar los programas de mantenimiento y/o reparación minimizando las paradas.

3.2.4 Sistemas ciber físicos

Un Sistema Ciber-Físico (CPS por sus siglas en inglés), es todo aquel dispositivo que integra capacidades de computación, almacenamiento y comunicación para controlar e interactuar con un proceso físico. Los Sistemas Ciber-Físicos están, normalmente, conectados entre sí y también con servicios remotos de almacenamiento y gestión de datos. (PodCast Industria 4.0, 2016)

Cada activo tendrá un doble en el espacio cibernético ayudando a predecir, primero, y prevenir, después, fallas potenciales. En un entorno de automatización industrial inteligente como el que propician los sistemas ciber físicos que caracterizan la industria 4.0, las máquinas y equipos no sólo son capaces de llevar a cabo análisis que les permitan el autoconocimiento, sino que incluyen capacidades de autoevaluación, autocomprobación, autoconfiguración y también de llevar a cabo su propia optimización para obtener más inteligencia y un mayor rendimiento. (OBS Business School , 2018)

La arquitectura que se debe centrar en cómo habilitar los activos para utilizar datos e información para crear conocimiento y sabiduría, para ello se necesitan 5 pasos:

1. Propiciar una conexión inteligente: desde el nivel de máquina o componente, lo primero es tener claro cómo adquirir datos de manera eficiente y confiable. Un protocolo de comunicación y el diseño de un esquema de red de fábrica robustos basado en métodos conocidos de comunicación sin cables, como *Bluetooth* o *Wi-Fi*, por ejemplo, es el principio. Además, hay que garantizar la calidad y transparencia de los datos, requisito indispensable cuando el objetivo es lograr que los sistemas de máquinas sean más inteligentes.
2. Convertir datos a información: en un entorno industrial, los datos pueden provenir de diferentes recursos, incluidos controladores, sensores, sistemas de fabricación (ERP, MES, SCM y sistema CRM) o registros de mantenimiento, entre otros. Estos datos deben convertirse en información significativa para una aplicación en el mundo real, por lo que es preciso implementar las capacidades que harán posible extraer valor de los bits de información recogidos.

3. Construir una gran base de conocimiento para cada activo: una vez que es posible recolectar información de sus sistemas, hay que garantizar la robustez del nivel cibernético, utilizando esta información para crear un avatar cibernético para dicho activo.
4. Proporcionar las soluciones para convertir las señales de los activos en información: la automatización industrial avanza hacia un modelo cognitivo, en el que el activo mismo puede aprovechar este sistema de monitorización en línea para diagnosticar sus posibles fallos y conocer su potencial degradación. Este tipo de sistemas pueden utilizar algunos algoritmos de predicción específicos para anticipar errores, problemas, ineficiencias o interrupciones dentro de un plazo temporal.
5. Configurar el sistema para que sea capaz de enviar oportunamente información sobre los problemas detectados: el conocimiento extraído puede enviarse al sistema de gestión empresarial para que usuarios y gerentes puedan tomar la decisión correcta. Al mismo tiempo, el activo puede ajustar su carga de trabajo o su programa de fabricación para reducir las pérdidas provocadas por el funcionamiento incorrecto y, finalmente, lograr un sistema resistente. (OBS Business School , 2018)

3.2.5 Realidad virtual

La tecnología de la realidad virtual recrea un espacio tridimensional virtual inmersivo, similar a uno real. Si una persona se coloca unas gafas de realidad virtual, se priva totalmente del mundo real que tiene alrededor y puede interactuar dentro de ese mundo virtual mediante dispositivos adecuados (guantes, mandos).

La realidad virtual puede aplicarse en diversos aspectos del mantenimiento, por ejemplo, en entrenamiento, en efecto, la implementación de dispositivos de realidad virtual permite a los técnicos de mantenimiento tener experiencias realistas sobre las características y dimensiones de los activos. Asimismo, podría simular reparaciones y resolver incidencias ficticias.

Las ventajas de la aplicación de la realidad virtual en el mantenimiento son:

- Ahorrar tiempo en la búsqueda de información y solución de problemas.
- Mejorar el entrenamiento creando aplicaciones que incluyen también textos explicativos para formar empleados.
- Acceso al historial de datos, revisar fechas, datos y recibir una alarma preventiva cuando estén cerca de cumplir su fecha prevista o recibir avisos sobre la temperatura o el estado de desgaste de sus componentes.
- Etiquetar virtualmente los elementos para que no exista ningún tipo de error de identificación. Además, permite que se encuentre cualquier tipo de elemento en sistemas complejos sin dificultad. (Grupo Retailgas, 2020)
- Se puede gestionar intervenciones en tiempo real mediante teleasistencia en *streaming* en la que expertos pueden conectarse con los mantenedores u operarios eliminando

viajes de técnicos o expertos y dejando trazabilidad con registro gráfico de una reparación. Esto reduce la brecha de habilidades.

- Reduce tiempos de inactividad de los equipos al permitir que los operadores resuelvan los problemas como expertos especializados incrementando la disponibilidad de los equipos.
- Se puede acceder al historial de mantenimiento de cualquier componente de la instalación, introduciendo y gestionando intervenciones en campo desde los dispositivos

3.2.6 Realidad aumentada

La realidad aumentada es una tecnología que conecta al mundo real con lo digital. Lo cual consiste en superponer imágenes generadas digitalmente sobre elementos de un entorno real. La realidad aumentada permite crear una experiencia especializada en el entorno de trabajo, en cualquier momento y desde cualquier lugar. Cuando surja un problema se podrá tener conexión con especialistas que ayudarán a diagnosticar el problema y poder así brindar una solución rápida y efectiva. De esta forma se podrá reducir el tiempo de inactividad de la máquina y el tiempo de reparación lo que conlleva a ahorrar esfuerzo y dinero.

Las ventajas de la Realidad Aumentada en el Mantenimiento son:

- Información en tiempo real: Gracias a la conexión de la plataforma con variadas fuentes de datos.
- Gestión del Mantenimiento: Es posible acceder al histórico de mantenimientos completos para poder introducir y gestionar incidencias desde la propia plataforma.
- Acceso a la documentación: Gracias a la plataforma implementada, una vez que se identifica el elemento o equipo, se pueden acceder a los documentos.
- Simulación de Procesos: Desde un aplicativo se pueden realizar simulaciones sobre los equipos de planta, así el operario puede accionar determinados mecanismos o cambiar variables del equipo para poder así visualizar una simulación 3D.
- Teleasistencia: Mediante el *streaming* se pueden gestionar incidencias en tiempo real de tal forma que el operario se pueda conectar con el equipo técnico. (IDEA Ingeniería, s.f.)

Los usos de la realidad aumentada en las estrategias de mantenimiento son:

- Mantenimiento preventivo: La realidad aumentada ayuda a prevenir fallas e incidentes, algunos ejemplos de usos son instrucciones de montaje al operador para asegurar la calidad del proceso, instrucciones de inspección, instrucciones detalladas para procedimientos nuevos o desconocidos y verificación de cumplimiento de proceso.
- Mantenimiento correctivo: Estos procesos se llevan a cabo para reparar los daños encontrados en el mantenimiento preventivo y la realidad aumentada se puede usar para instrucciones de servicio y brindar asistencia remota en dar mantenimiento sin importar que el experto no esté en el lugar.

- Mantenimiento predictivo: Esta técnica consiste en pronosticar el punto de falla de una máquina antes de que suceda. La realidad aumentada permite conocer datos en tiempo real y conocer el rendimiento de la maquinaria para predecir futuras fallas.

3.2.7 Fabricación aditiva

También conocida como fabricación por adición, se le conoce como un nuevo concepto de producción por el cual el material a elección (plástico o metal) se deposita capa a capa de una forma controlada. Esta técnica permite producir diversos diseños geométricos personalizados según las necesidades de cada sector.

Este sector de producción por adición ha incrementado en los últimos tiempos debido a la rapidez, precisión y ahorro que brinda a los usuarios. Con esta tecnología se elabora una gran variedad de componentes para diversos sectores productivos, principalmente de sectores de la salud y la industria aeroespacial.

Actualmente, existen distintas tecnologías de fabricación aditiva que se distinguen por los materiales que usan como filamentos, hilos, polvos, líquidos o resinas, y además por las diversas formas de unir estos materiales, ya sean mediante calor, haz de luz, láser o soldadura.

En comparación con las técnicas de fabricación tradicionales, este tipo de tecnología tiene enormes beneficios como la reducción de procesos intermedios como la producción de utillajes, ya que permite la obtención de piezas hasta un 90% más rápido.

La fabricación aditiva en el mantenimiento:

- Permite la creación de una cadena de productos en menor tamaño, en función a la demanda.
- Disminuye el coste inicial de la inversión para poder fabricar un nuevo producto.
- Facilita el proceso de diseño, eliminando restricciones tradicionales del mismo.
- Permite la creación de diversos montajes con una variedad de materiales y propiedades a pesar de tener un ensamblado sencillo.
- Reducir los costos de producción a pesar de que existan dificultades geométricas en el diseño.
- Identifica las diferencias y permite personalizar cada producto sin encarecerlo.
- Integra mecanismos en las piezas que se han producido sin realizar los respectivos montajes, calibrados ni ajustes.
- Elimina los desechos del material, facilitando un mejor cuidado del medio ambiente y la reducción de costes. (Dynatec, 2020)

La fabricación aditiva en el panorama industrial está permitiendo a las empresas adoptar soluciones innovadoras, personalizadas y tecnológicamente avanzadas para satisfacer las demandas de sus clientes.

3.2.8 Cloud Computing

El Cloud Computing es un servicio que consiste en utilizar la capacidad de procesamiento y de almacenamiento de servidores y computadoras repartidos por todo el mundo, unidos entre sí a través de internet. Las empresas pueden alojar allí grandes cantidades de información de sus actividades sin necesidad de invertir en una infraestructura (hardware, software, seguridad, soporte, etc) para acceder y disponer de la misma.

La gran ventaja de este potente servicio es que resulta muy barata, solo se necesita desarrollar una capa de software que permita gestionar los recursos y las herramientas de programación. (Ibérica Multimedia, 2008)

El *cloud computing* en el mantenimiento de las industrias tiene las siguientes características:

- Se paga solo por los recursos que usan la empresa, sin costos adicionales, evitando así inversiones en infraestructura o costos por adquisición de licencias.
- Externalización: Se puede externalizar la gestión de recursos informáticos a un proveedor especializado, lo cual permite que las empresas eliminen los costos asociados a la instalación y mantenimiento de una infraestructura informática propia.
- Multifuncionalidad: Cada empresa elige las funciones del sistema que necesita en cada momento.
- Multiusuario: Autoriza que diferentes usuarios de una empresa puedan usar los servicios de una misma plataforma.
- Auto servicio bajo demanda: Los diversos usuarios pueden consultar las capacidades del servicio sin tener que consultar al proveedor.
- Acceso sin restricciones: Se permite acceder a cualquier lugar / momento y con cualquier dispositivo con acceso a la red. (Aina Tecnología, 2012)

3.2.9 Big Data

La digitalización de las señales de campo trae consigo una vasta cantidad de datos a lo que se llama Big Data. Estos datos provienen de los sensores de campo, dispositivos IoT, actuadores instalados en la maquinaria o procesos industriales, y se almacenan en grandes repositorios de información llamados *Data Warehouse* que se pueden analizar e interpretar mediante técnicas apropiadas de *machine learning* para una toma de decisiones informada y generar valor en la organización

De este modo, detectar tendencias y comportamientos de los activos permitirá predecir fallas antes de que ocurran (a esto se le conoce como análisis predictivo), planificar reparaciones, evitar el estancamiento de las operaciones y eliminar la posibilidad de una falla catastrófica. Con todo ese apoyo se reducirán los costos de mantenimiento y se mejorará el tiempo medio entre fallas (MTBF; por sus siglas en inglés), la confiabilidad de los activos y disponibilidad de los sistemas

Hay diversas variables que podrían ser relevantes para predecir si una máquina fallará a corto plazo, como por ejemplo los tiempos de un determinado proceso, lecturas de voltajes o corrientes, niveles de humedad o presión. Estos datos obtenidos de diferentes fuentes de datos, apoyados con un modelo de Analítica Predictiva permitirán aprender los patrones que suelen preceder las fallas de una máquina para llevar a cabo un proceso de mantenimiento preventivo sin que la producción se vea afectada. A mayor cantidad de datos de entrenamiento, mejor se ajustará el modelo.

Lo atributos de los datos son los siguientes: Volumen, Variedad, velocidad, veracidad, visibilidad y valor, se describe a continuación cada uno de ellos:

- Volumen de data: Se refiere a la cantidad de datos que genera una organización por segundo. Estos datos provienen de diversas fuentes virtuales como: dispositivos electrónicos. En forma de texto, imágenes, videos, etc.
- Velocidad de data: la velocidad con la que el Big Data analiza los datos es muy alta, esto permite a las empresas recopilar información instantánea incrementando la velocidad de producción y transmisión de datos.
- Variedad de data: tiene que ver con todos los lugares donde los datos pueden ser almacenados y extraídos
- Valor de data: El objetivo del Big Data es agregar valor a la organización mediante un análisis preciso de los datos.
- Veracidad: De los millones de datos que se generan, muchos pueden ser falsos, es preciso filtrarlos en el análisis.
- Visibilidad: Es importante contar con las herramientas de visualización que permiten de una manera fácil leer los análisis realizados.

3.2.10 Block Chain

El *Blockchain* es un tipo de tecnología a base de datos diseñada de una manera radicalmente diferente. Brinda un excelente potencial de poder simplificar de manera drástica la contabilidad, cadenas de suministros, sistemas de registros, seguimiento de activos, etc. Las diversas empresas que aprovechan estas ventajas que brindan la base de datos *blockchain* reconocen grandes eficiencias operativas e incluso reconocen un cambio fundamental en la manera que se relaciona con los clientes. (Carretero Toro, s.f.)

Este tipo de tecnología permite crear un certificado de nacimiento para cada una de las piezas y de los componentes de un activo, de dar acceso y actualizarlo cada vez que un activo es verificado. Por otro lado, permite el acceso a esta información a las distintas partes implicadas y solo a aquellos datos a los que estén autorizados a conocer.

El *Blockchain* tiene alta potencialidad y se puede manifestar mejor en el sector mantenimiento. Se ha pronosticado que para el 2023, el 30% de las empresas de fabricación con más de 5 mil millones de dólares en ingresos habrán implementado proyecto piloto utilizando *blockchain* frente a menos del 5% actual.

El uso de esta tecnología ha proporcionado en cada área de mantenimiento como habilitar los detalles del producto, vincular el producto al proveedor, iniciar servicios anuales, ejecutar solicitudes de servicio, captura de registros de servicio y generar factura, es impecable.

Además, ha permitido que las organizaciones rastreen problemas con un fabricante de producto, componente o material específico, garantiza seguridad y mejorar el servicio y disminuir el costo de mantenimiento, así como también garantiza la información crítica del dispositivo y los detalles de su servicio se mantengan de forma segura en un libro mayor.

Blockchain como un libro mayor descentralizado también ayuda en el mantenimiento de las reglas. En lugar de almacenar y extraer el estado de los activos, también es posible almacenar las reglas que impulsan el mantenimiento de los activos, en línea con los procesos industriales.

Las actualizaciones de firmware por aire (OTA), que son comunes en los dispositivos de IoT con capacidades inalámbricas, enfrentan la amenaza de piratería, ya que el acceso físico puede no ser necesario en tales casos. La mayoría de estos marcos utilizan una arquitectura centralizada para actualizar una cantidad potencialmente grande de dispositivos, ampliando así el panorama de amenazas.

Alternativamente, se puede aprovechar un marco de *blockchain* con contratos inteligentes para proteger la integridad del proceso de actualización del firmware. Dichos sistemas se pueden emplear en ciudades inteligentes o escenarios con una gran cantidad de dispositivos y proveedores de servicios donde los nodos están autenticados, las comunicaciones protegidas y las condiciones de actualización especificadas y aplicadas a través de contratos inteligentes. (R, 2020)

Capítulo 4
Caso de implementación de la inteligencia artificial en la función mantenimiento

Los gestores del mantenimiento de equipos tienen hoy en día una gran oportunidad para implementar aplicaciones basadas en IA. En los diferentes sectores como minería, industria, telecomunicaciones, energía, etc se pueden hacer uso de aplicaciones para monitorear el desempeño de los diferentes equipos que forman parte de una unidad de procesamiento. Las tecnologías basadas en Inteligencia artificial permiten detectar diversos fenómenos que ocurren dentro de los equipos y que pueden no ser perceptibles por los sentidos humanos. Además, tiene la capacidad de ser predictiva, es decir, precisar deficiencias en los sistemas, que pueden llevar posteriormente a fallas en un equipo con las consecuencias que ellas conllevan.

En ese sentido explicaremos un caso de IA aplicada al mantenimiento el cual tiene que ver con la simulación de la degradación de un turboventilador. Donde toda la información de la data se ha obtenido del repositorio de la NASA.

4.1 Planteamiento y explicación del caso

CASO:

Se ha decidido trabajar con una data del repositorio de la NASA. La cual obtiene los datos de la simulación de la degradación de un turboventilador. Esta simulación se llevó a cabo utilizando C-MAPSS de la NASA, el cual es un programa de multiplataforma creado en colaboración entre el *Institute for Human and Machine Cognition* (IHMC). El conjunto de datos fue proporcionado por *Prognostics CoE* (Centro de Pronósticos por Excelencia) en el Centro de investigaciones de la NASA AMES.

Se tratará de investigar sobre el diagnóstico y pronóstico multimedia real que interpretan los diversos datos adquiridos por una red de sensores distribuidos y utiliza los flujos de datos para tomar decisiones críticas, lo cual permite proporcionar avances significativos en una amplia gama de aplicaciones. Estos sensores identifican las condiciones estresantes por las que puede pasar un activo, las cuales pueden ser: alta presión, alta temperatura, vibraciones, alto campo de irradiación, entre otras

El activo o sistema cuando es impuesto a estas condiciones va generando un daño en su integridad y funcionalidad, para lo que se requiere un monitoreo constante debido a que puede llegar a generar situaciones negativas para la seguridad y salud de los trabajadores.

Un ejemplo de situación negativa serían las plantas de energía nuclear, las averías inesperadas pueden ser extremadamente costosas y desastrosas, ya que inmediatamente dan como resultado la pérdida de producción de energía, el costo de mantenimiento correcto, la reducción de la confianza del público y, posiblemente, lesiones y muertes humanas. Para poder reducir y tratar de eliminar tales problemas, es necesario evaluar con precisión el estado actual del sistema y predecir con precisión las vidas útiles remanente (RUL) de los componentes operativos, subsistemas y sistemas en los sistemas de ingeniería de alto riesgo.

En general, los enfoques de pronóstico se pueden clasificar en enfoques basados en modelos, en datos y en híbridos. La aplicación de enfoques generales de pronóstico basados en modelos se basa en la comprensión de la sistemática de fallas y los modelos de degradación del sistema subyacente.

El mantenimiento y la gestión de ciclo de vida de estos sistemas de ingeniería de alto riesgo para minimizar el coste, maximizar la disponibilidad y prolongar la vida útil, es una de las áreas de aplicación beneficiarias. Es por ello que en este trabajo de investigación se evaluarán algunos algoritmos como: *Random Forest, Gradiant Boostin y Super Vector Machine* para evaluar cuál de los algoritmos presenta un menor R cuadrado y sería el mejor para procesar la data adquirida en relación al RUL y los respectivos ajustes que brinda la información dada.

4.2 Solución y posibles respuestas

Los Para poder realizar este código se ha utilizado diversos algoritmos como *Random Forest,* validación cruzada, mínimos cuadrados, *Support Vector Machine* y *Gradient Boostin*.

El único trabajo relevante que conocemos proviene de donde solo se emplean dos algoritmos basados en datos como algoritmos de miembros y sus ponderaciones se determinan empíricamente en función de un error de entrenamiento de una sola vez sin un esquema sistemático para la estimación de errores y la validación del rendimiento.

La mayoría de las prácticas de pronóstico basadas en datos seleccionan un solo algoritmo con la mejor precisión del grupo de algoritmos mientras descartan los demás. Este enfoque no solo desperdicia los recursos dedicados a desarrollar diferentes algoritmos, sino que también adolece de la falta de robustez.

Las implementaciones exitosas de los algoritmos de pronóstico requieren la extracción de las firmas de las condiciones de salud y el conocimiento previo de la salud de las señales sensoriales de entrenamiento / pruebas masivas de las unidades del sistema de ingeniería. Para ello, este estudio utilizará un sistema genérico de índices de salud que se compone de dos índices de salud diferenciados: índice de salud física (PHI) e índice de salud virtual (VHI). En general, el PHI usa una señal física dominante como una métrica de salud directa y, por lo tanto, es aplicable solo si las señales sensoriales están directamente relacionadas con la física de fallas.

En esta parte del código se cargan los paquetes necesarios para visualizar la data disponible. Se usa *print* (os.listdir()) para escribir el contenido del archivo, el comando *np.random.seed* para crear números aleatorios que tengan un límite al momento de correr el código.

```
# load necessary packages and view available data
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import keras
%matplotlib inline
sns.set()
print(os.listdir("/content/drive/My Drive/data3/CMAPSSData"))
# Setting seed for reproducability
np.random.seed(1234)
PYTHONHASHSEED = 0
```

El comando str(i) seleccionará un número del 1 al 4 tratando de formar como una oración para indicar las operaciones. Se usarán 3 operaciones y habrá 20 sensores ya que Python cuenta -1. A los *column_names* indicará los nombres de las columnas y se le sumarán las variables para darle un orden.

```
# the files did not contain headers. Here we create labels based on documentation
target_var = ['Target_Remaining_Useful_Life']
index_columns_names =  ["UnitNumber","Cycle"]
op_settings_columns = ["Op_Setting_"+str(i) for i in range(1,4)]
sensor_columns =["Sensor_"+str(i) for i in range(1,22)]
column_names = index_columns_names + op_settings_columns + sensor_columns
print(column_names)
```

En esta parte del código se define la data como *train* y se usará la librería Panda para la lectura y procesamiento de datos ya que es necesario que se lea en csv, por ello lo separa en puntos y comas.

Con el comando test se cargará la data del test, pero separándola.

El comando *prin*t(*train shape; test shape*), indica respectivamente el nombre y las dimensiones de la data.

Mientras que el comando *head* (10) se encarga de imprimir las 5 primeras filas de data de entrenamiento que tenga unit number =1.

```
# load data
train= pd.read_csv('/content/drive/My Drive/data3/CMAPSSData/train_FD001.txt', sep=" ", heade
r=None)
test = pd.read_csv('/content/drive/My Drive/data3/CMAPSSData/test_FD001.txt', sep=" ", header=
None)
print("train shape: ", train.shape, "test shape: ", test.shape)
```

```
# drop pesky NULL columns
train.drop(train.columns[[26, 27]], axis=1, inplace=True)
test.drop(test.columns[[26, 27]], axis=1, inplace=True)
# name columns
train.columns = column_names
test.columns = column_names
#train.head(10)
#test.head(10)
train[train['UnitNumber'] == 1].head(5)
test[test['UnitNumber'] == 1].head(5)
```

Luego de haber hecho la respectiva limpieza de data, se corrió el código y se obtuvieron los siguientes resultados. Se logró ordenar la data y nombrar los encabezados, donde el *train shape*: (20631,28) indica que deben coincidir 28 columnas.

```
train shape:  (20631, 28) test shape:  (13096, 28)
```

	UnitNumber	Cycle	Op_Setting_1	Op_Setting_2	Op_Setting_3	Sensor_1	Sensor_2	Sensor_3
0	1	1	0.0023	0.0003	100.0	518.67	643.02	1585.29
1	1	2	-0.0027	-0.0003	100.0	518.67	641.71	1588.45
2	1	3	0.0003	0.0001	100.0	518.67	642.46	1586.94
3	1	4	0.0042	0.0000	100.0	518.67	642.44	1584.12
4	1	5	0.0014	0.0000	100.0	518.67	642.51	1587.19

Figura 13. Sensores del 1 al 3, *Unite Number, Cycle y OP Settings* del 1 al 3. Luego de entrenar la data y tener la matriz de (20631,28)

Sensor_4	Sensor_5	Sensor_6	Sensor_7	Sensor_8	Sensor_9	Sensor_10	Sensor_11	Sensor_12
1398.21	14.62	21.61	553.90	2388.04	9050.17	1.3	47.20	521.72
1395.42	14.62	21.61	554.85	2388.01	9054.42	1.3	47.50	522.16
1401.34	14.62	21.61	554.11	2388.05	9056.96	1.3	47.50	521.97
1406.42	14.62	21.61	554.07	2388.03	9045.29	1.3	47.28	521.38
1401.92	14.62	21.61	554.16	2388.01	9044.55	1.3	47.31	522.15

Figura 14. Sensores del 1 al 3, *Unite Number*, *Cycle y OP Settings* del 4 al 12. Luego de entrenar la data y tener la matriz de (20631,28)

Sensor_13	Sensor_14	Sensor_15	Sensor_16	Sensor_17	Sensor_18	Sensor_19	Sensor_20	Sensor_21
2388.03	8125.55	8.4052	0.03	392	2388	100.0	38.86	23.3735
2388.06	8139.62	8.3803	0.03	393	2388	100.0	39.02	23.3916
2388.03	8130.10	8.4441	0.03	393	2388	100.0	39.08	23.4166
2388.05	8132.90	8.3917	0.03	391	2388	100.0	39.00	23.3737
2388.03	8129.54	8.4031	0.03	390	2388	100.0	38.99	23.4130

Figura 15. Sensores del 1 al 3, *Unit Number Cycle* y *OP settings* del 13 al 21 Luego de entrenar la data y tener la matriz de (20631,28)

En esta parte del código se trata de encontrar el máximo valor de ciclos que tengan valor. Busca así agregar otra columna. Se usa el comando *train merge* para fusionar una línea con otra con la máxima cantidad de ciclos. No los renombra, solo los mezcla. Vida útil actual = número máximo de ciclos en unidad – ciclo actual.

```
# this section calculates Remaining Useful Life (RUL) in T- minus notation for the training data
# find the last cycle per unit number
max_cycle = train.groupby("UnitNumber")["Cycle"].max().reset_index()
max_cycle.columns = ['UnitNumber', 'MaxOfCycle']
# merge the max cycle back into the original frame
train_merged = train.merge(max_cycle, left_on='UnitNumber', right_on='UnitNumber', how='inner')
# calculate RUL for each row
Target_Remaining_Useful_Life = train_merged["MaxOfCycle"] - train_merged["Cycle"]
train_with_target = train_merged["Target_Remaining_Useful_Life"] = Target_Remaining_Useful_Life
# remove unnecessary column
train_with_target = train_merged.drop("MaxOfCycle", axis=1)
train_with_target[train_with_target['UnitNumber'] == 1].head(5)
```

A continuación, se muestran los resultados de la vida útil restante en relación a T menos para los datos de entrenamiento. Se quiere encontrar el último ciclo por unidad (*unit number*):

	UnitNumber	Cycle	Op_Setting_1	Op_Setting_2	Op_Setting_3	Sensor_1	Sensor_2	Sensor_3
0	1	1	-0.0007	-0.0004	100.0	518.67	641.82	1589.70
1	1	2	0.0019	-0.0003	100.0	518.67	642.15	1591.82
2	1	3	-0.0043	0.0003	100.0	518.67	642.35	1587.99
3	1	4	0.0007	0.0000	100.0	518.67	642.35	1582.79
4	1	5	-0.0019	-0.0002	100.0	518.67	642.37	1582.85

Figura 16. Data entrenada con los sensores del 1 al 3, *unit number*, ciclo y operaciones de configuración

Sensor_4	Sensor_5	Sensor_6	Sensor_7	Sensor_8	Sensor_9	Sensor_10	Sensor_11	Sensor_12
1400.60	14.62	21.61	554.36	2388.06	9046.19	1.3	47.47	521.66
1403.14	14.62	21.61	553.75	2388.04	9044.07	1.3	47.49	522.28
1404.20	14.62	21.61	554.26	2388.08	9052.94	1.3	47.27	522.42
1401.87	14.62	21.61	554.45	2388.11	9049.48	1.3	47.13	522.86
1406.22	14.62	21.61	554.00	2388.06	9055.15	1.3	47.28	522.19

Figura 17. Data entrenada con los sensores del 4 al 12, *unit number*, ciclo y operaciones de configuración

Sensor_13	Sensor_14	Sensor_15	Sensor_16	Sensor_17	Sensor_18	Sensor_19	Sensor_20	Sensor_21
2388.02	8138.62	8.4195	0.03	392	2388	100.0	39.06	23.4190
2388.07	8131.49	8.4318	0.03	392	2388	100.0	39.00	23.4236
2388.03	8133.23	8.4178	0.03	390	2388	100.0	38.95	23.3442
2388.08	8133.83	8.3682	0.03	392	2388	100.0	38.88	23.3739
2388.04	8133.80	8.4294	0.03	393	2388	100.0	38.90	23.4044

Figura 18. Data entrenada con los sensores del 13 al 21, *unit number*, ciclo y operaciones de configuración

Target_Remaining_Useful_Life
191
190
189
188
187

Figura 19. Tiempo de vida útil restante

En la siguiente parte del código se usará el comando sns el cual es una abreviación del *seaborn, pair grid* es un comando para realizar gráficas y *query* es un comando que se usa para indicar los *unit numbers* menores a 15.

```
# use seaborn to visualize featuresto target (RUL)
explore = sns.PairGrid(data=train_with_target.query('UnitNumber < 15') ,
            x_vars=target_var,
            y_vars=sensor_columns + op_settings_columns,
            hue="UnitNumber", size=3, aspect=2.5)
explore = explore.map(plt.scatter, alpha=0.5)
explore = explore.set(xlim=(400,0))
explore = explore.add_legend()
```

Posteriormente con ayuda del comando *sns.PairGrid*, se graficarán los valores de la vida útil restante (eje x) y los sensores + las operaciones configuradas (eje y):

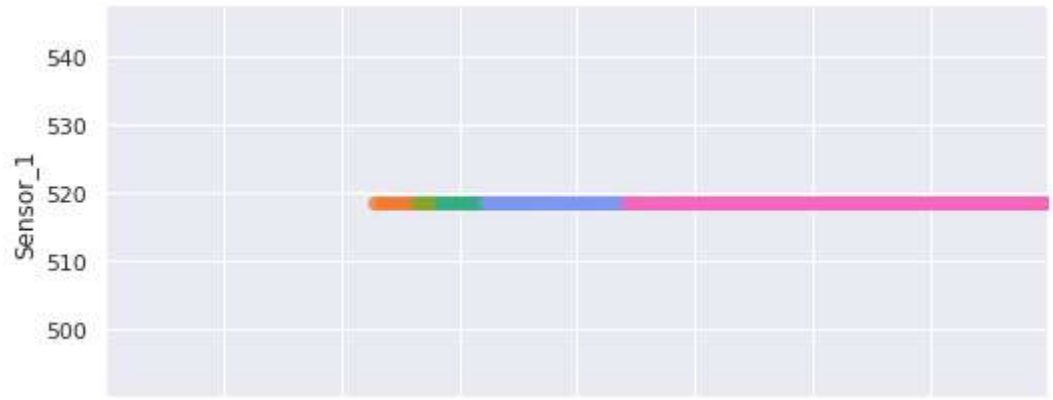


Figura 20. Gráfica obtenida del sensor 1 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

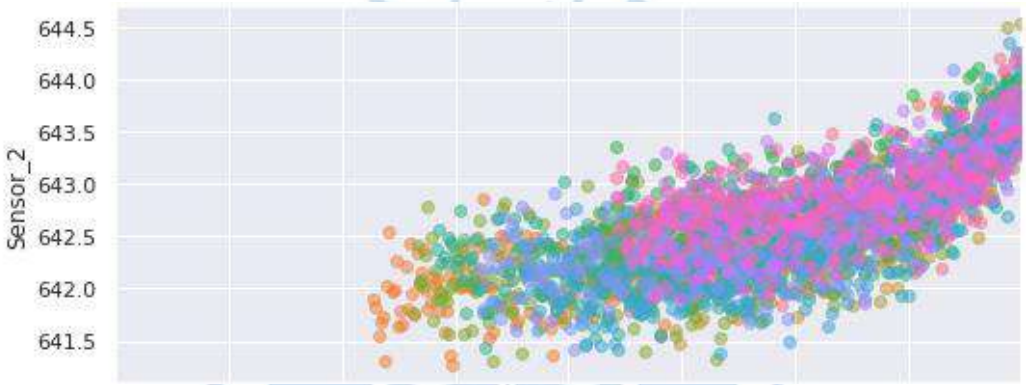


Figura 21. Gráfica obtenida del sensor 2 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

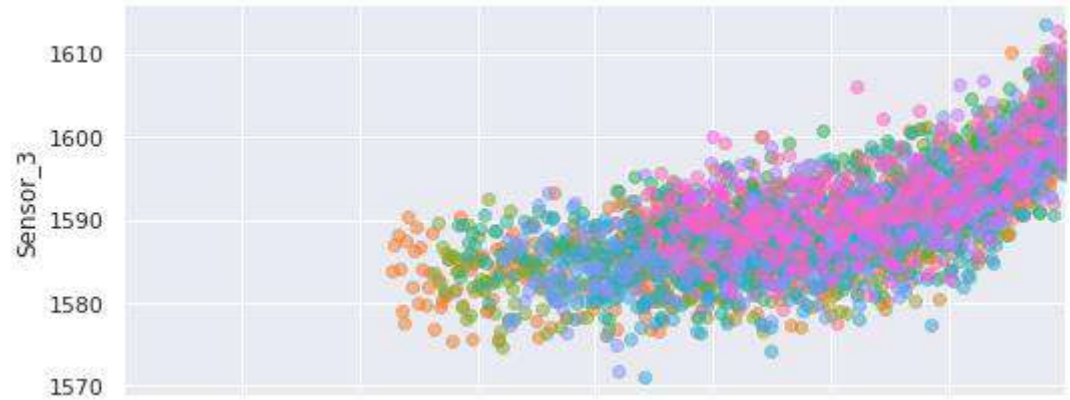


Figura 22. Gráfica obtenida del sensor 3 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

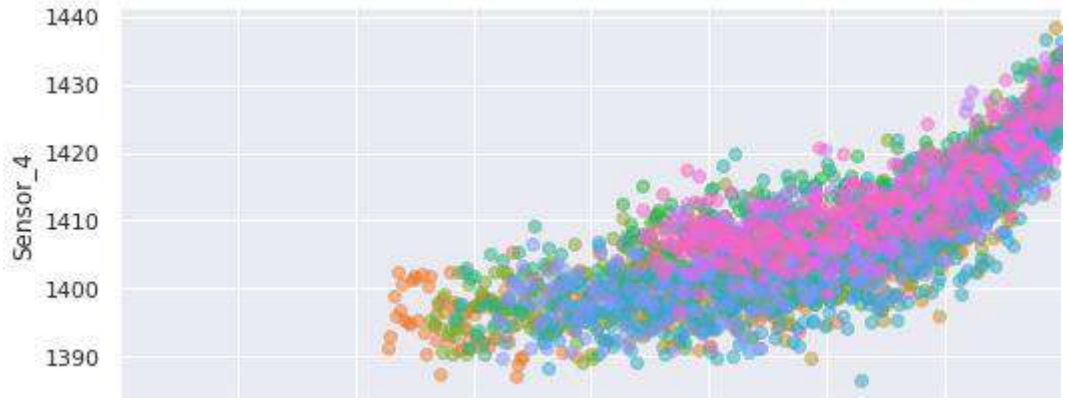


Figura 23. Gráfica obtenida del sensor 4 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

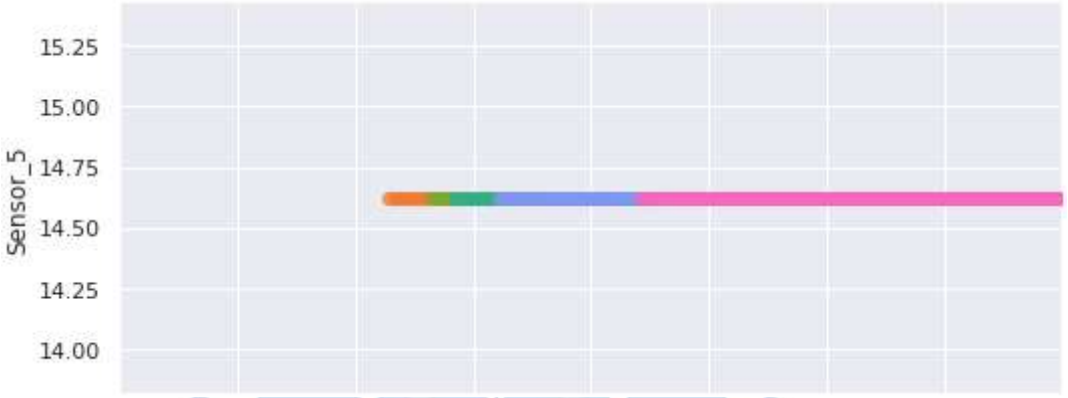


Figura 24. Gráfica obtenida del sensor 5 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

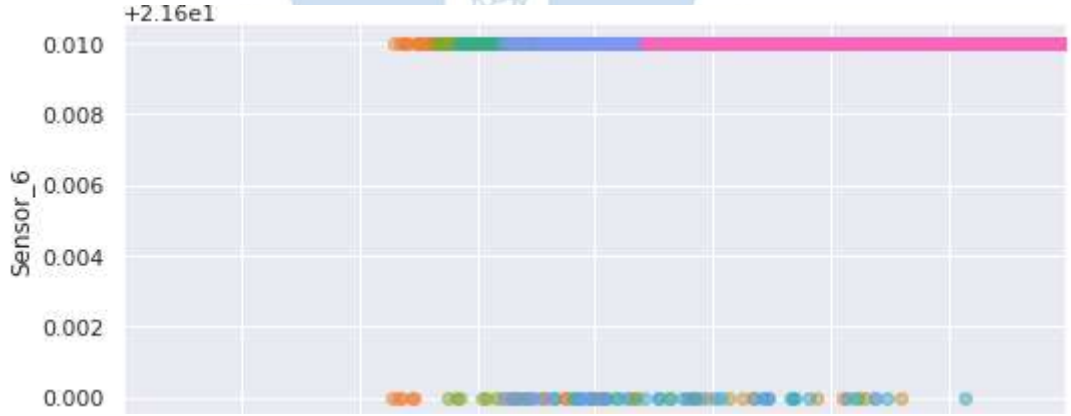


Figura 25. Gráfica obtenida del sensor 6 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

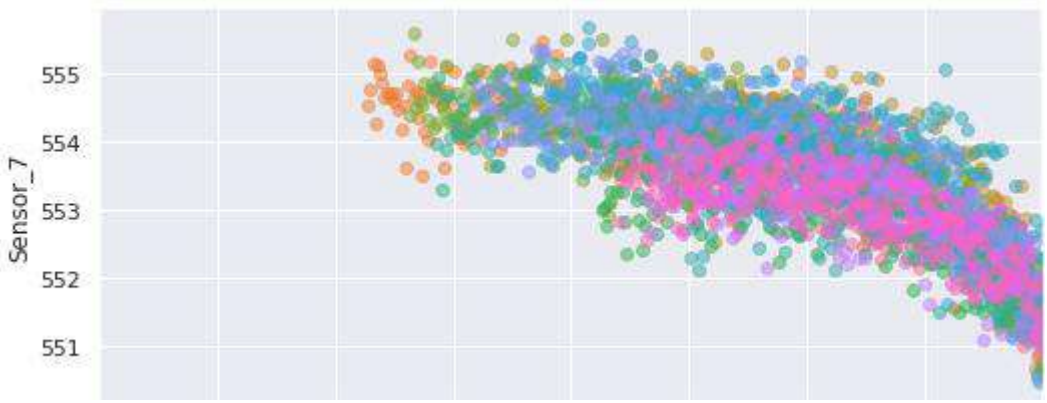


Figura 26. Gráfica obtenida del sensor 7 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

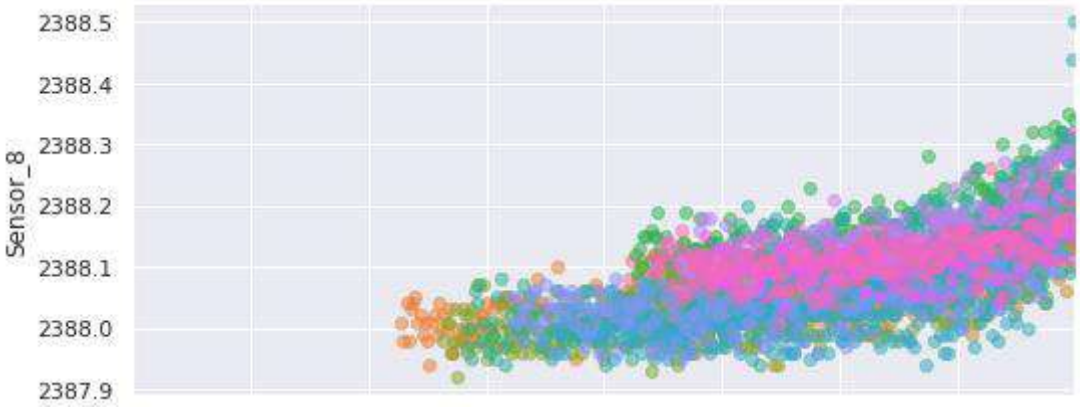


Figura 27. Gráfica obtenida del sensor 8 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

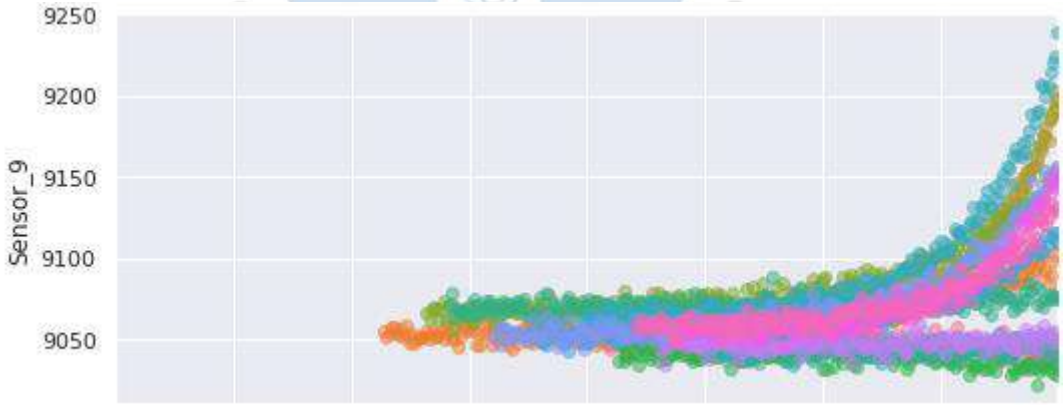


Figura 28. Gráfica obtenida del sensor 9 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

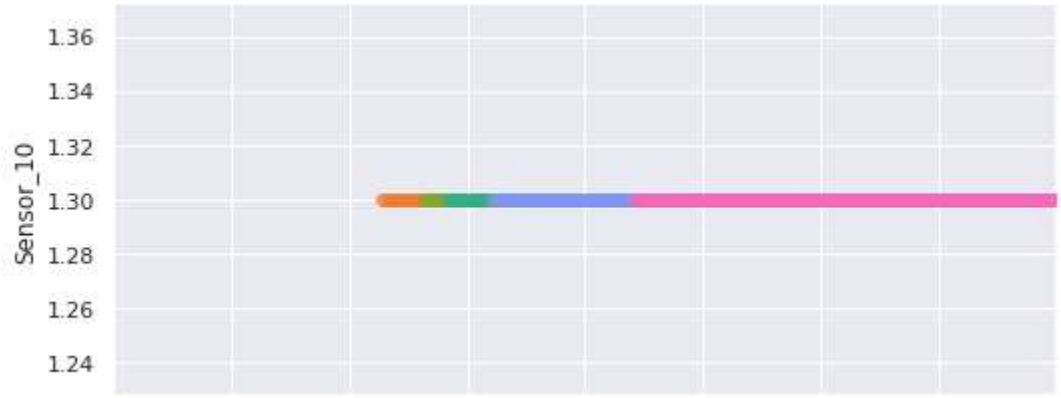


Figura 29. Gráfica obtenida del sensor 10 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

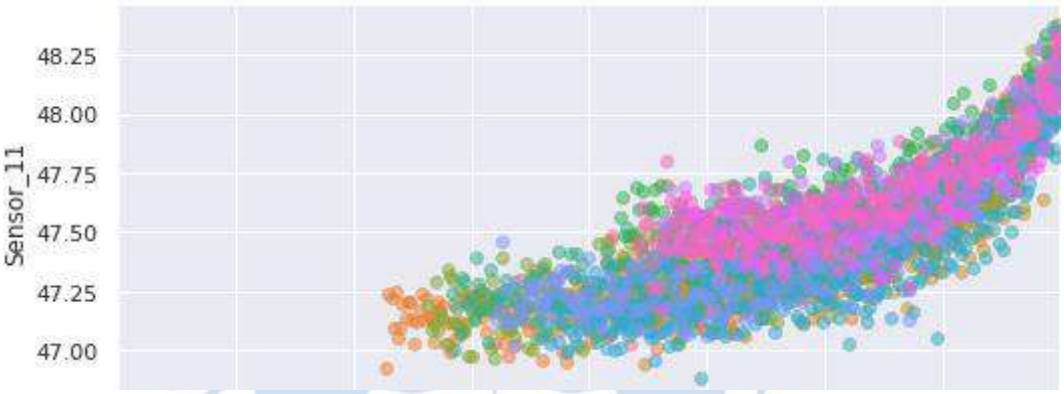


Figura 30. Gráfica obtenida del sensor 11 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

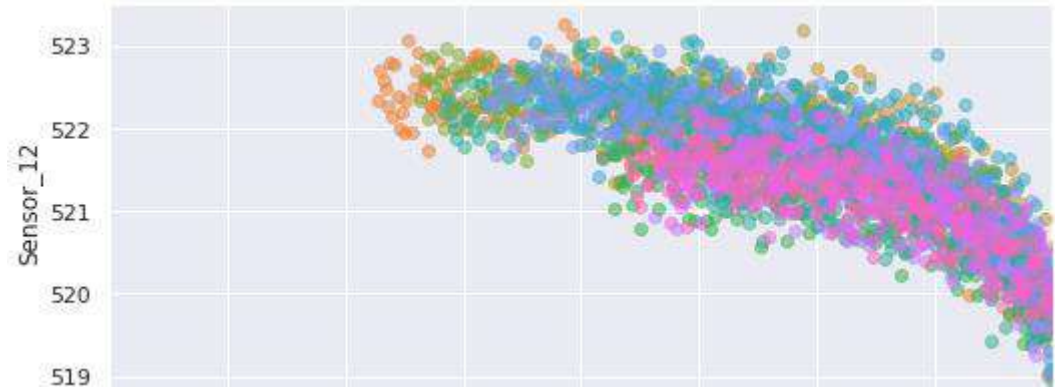


Figura 31. Gráfica obtenida del sensor 12 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

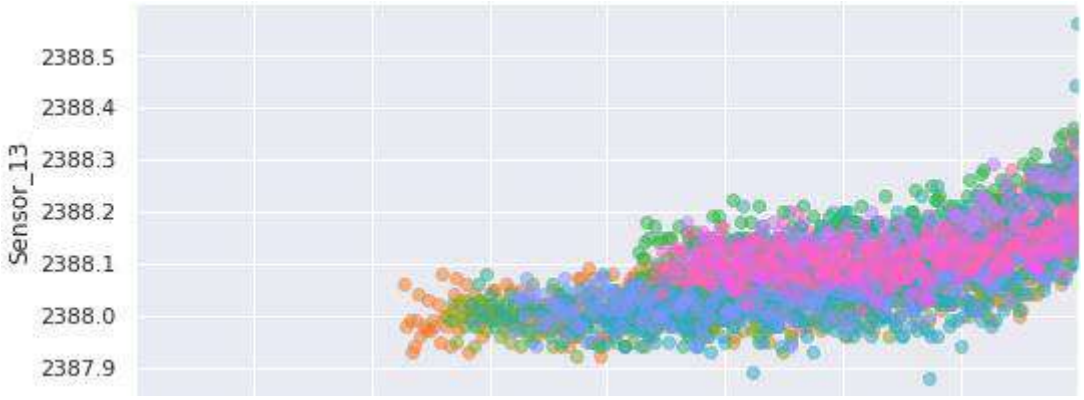


Figura 32. Gráfica obtenida del sensor 13 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

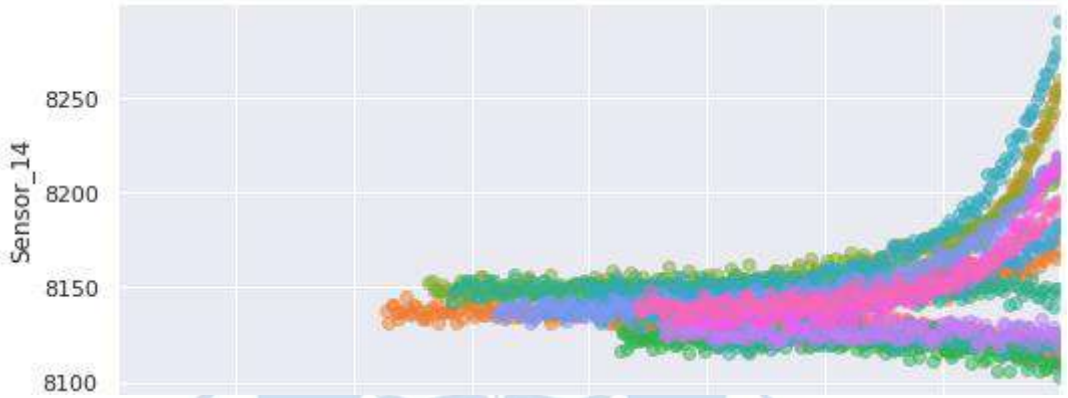


Figura 33. Gráfica obtenida del sensor 14 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

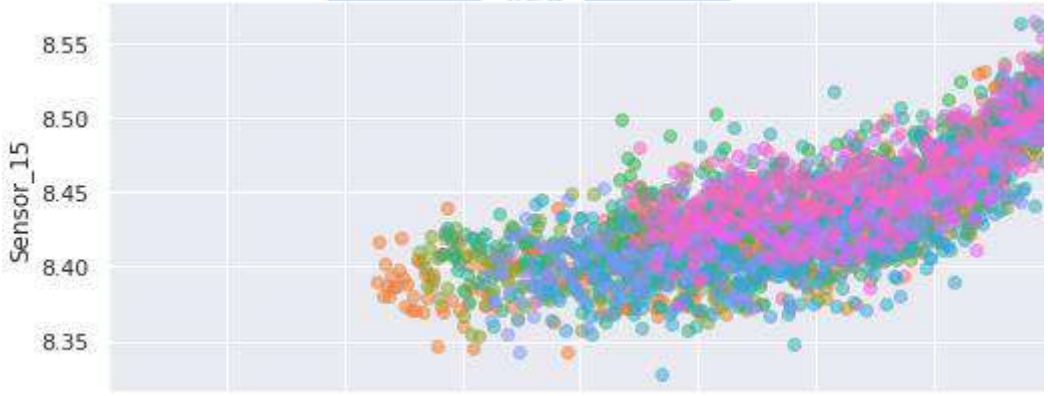


Figura 34.Gráfica obtenida del sensor 15 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

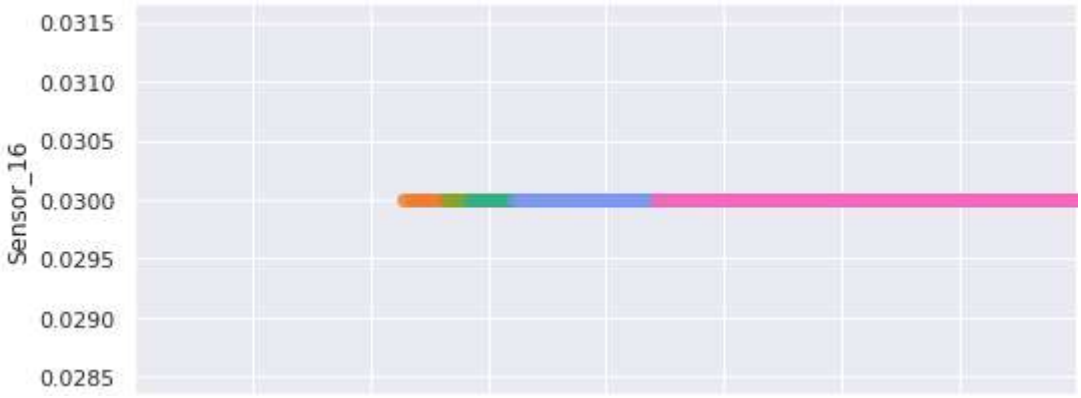


Figura 35.Gráfica obtenida del sensor 16 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

Figura 36. Gráfica obtenida del sensor 17 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

Figura 37. Gráfica obtenida del sensor 18 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

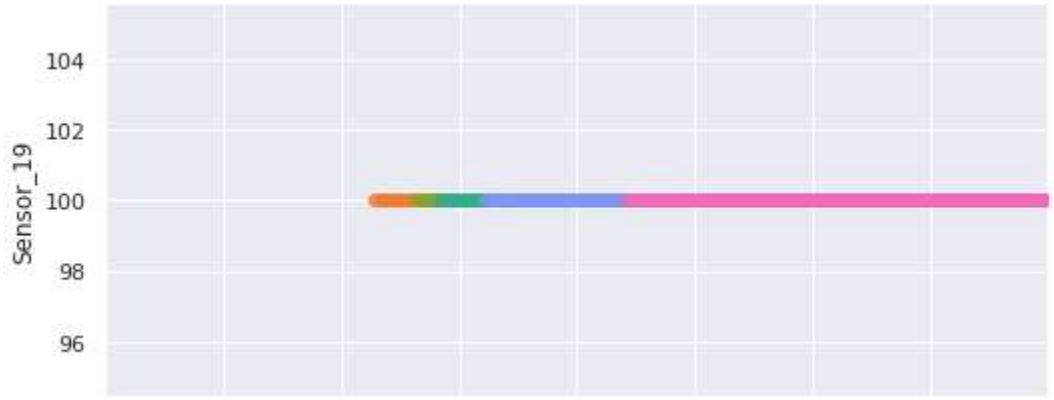


Figura 38. Gráfica obtenida del sensor 19 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

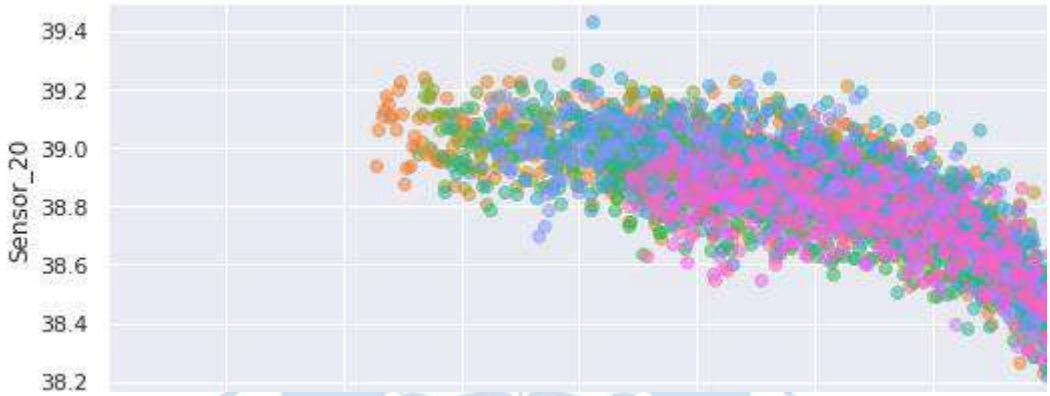


Figura 39. Gráfica obtenida del sensor 20 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

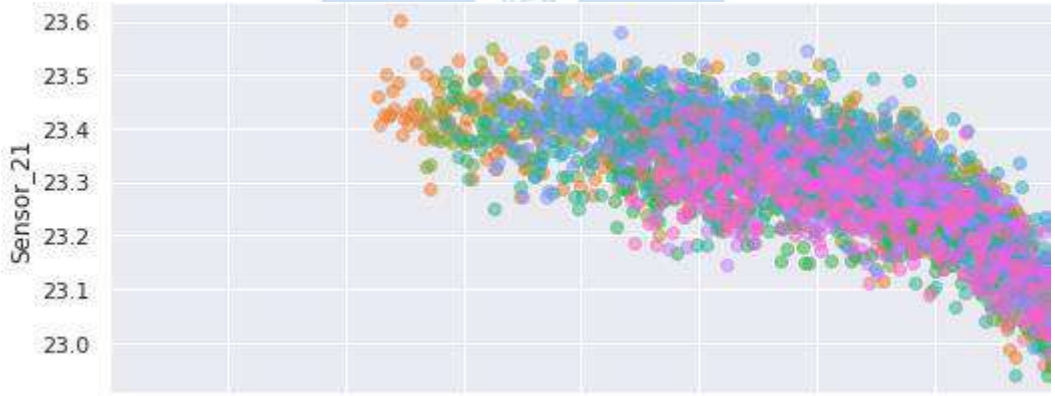


Figura 40. Gráfica obtenida del sensor 21 + operaciones configuradas (eje y) vs. Vida útil restante (eje x)

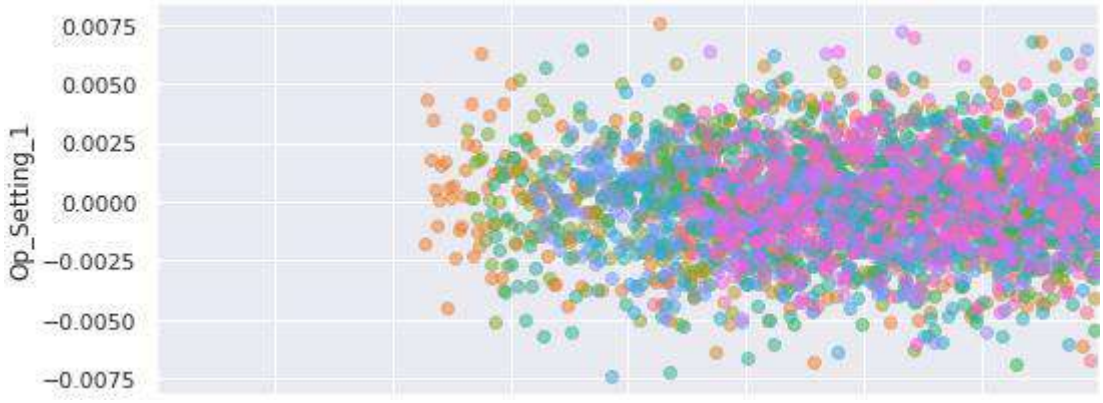


Figura 41. Gráfica de operaciones configuradas 1 (eje y) vs. Vida útil restante (eje x)

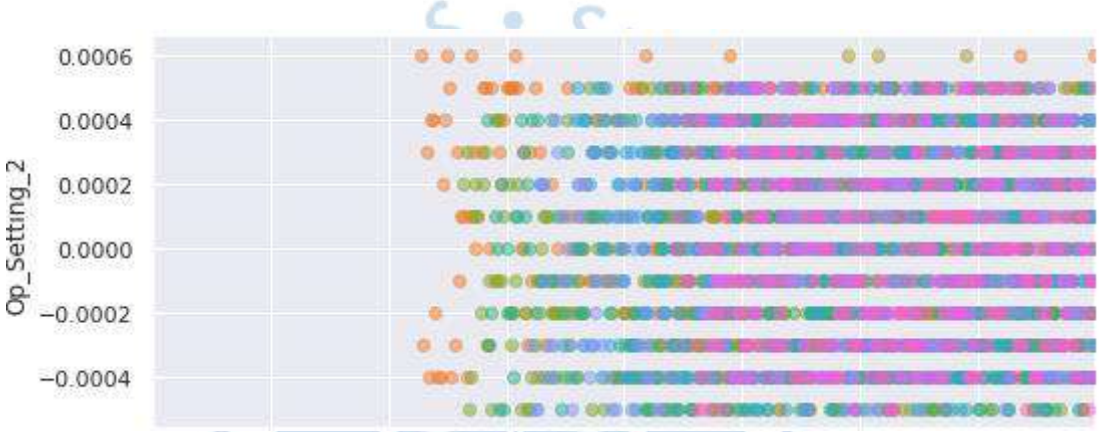


Figura 42. Gráfica obtenida de operaciones configuradas 2 (eje y) vs. Vida útil restante (eje x)

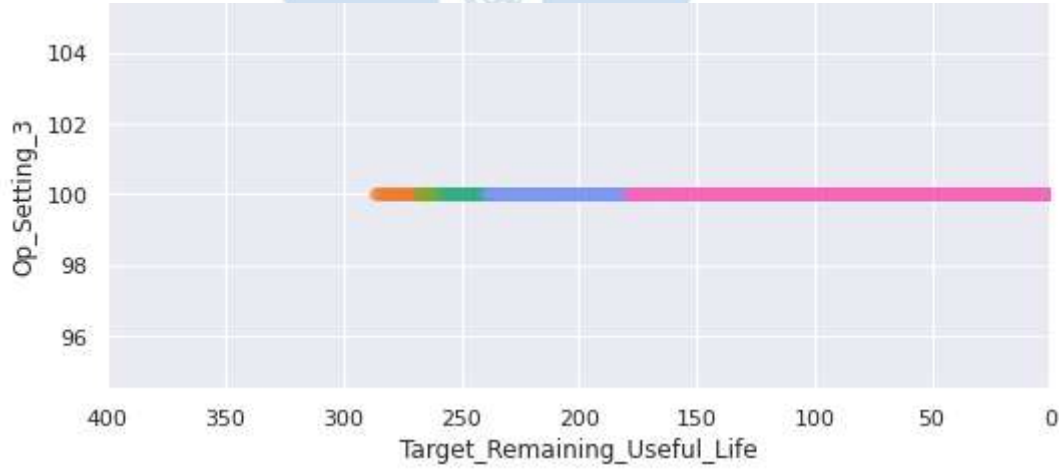


Figura 43. Gráfica obtenida de operaciones configuradas 3 (eje y) vs. Vida útil restante (eje x)

Como la configuración de operaciones 3 está estable, se decidió graficar los sensores más activos con la configuración de operaciones 1 y 2. Donde el eje “x” será de las operaciones 1 y 2, y el eje “y” coge los sensores a excepción del 1,5 y 6 porque no cambian respecto a las operaciones para luego redimensionar la data.

```
# operational setting 3 is stable, let's visualize op setting 1 and 2 against some of the most active se
nsors
g = sns.pairplot(data=train_with_target.query('UnitNumber < 15'),
                 x_vars=["Op_Setting_1","Op_Setting_2"],
                 y_vars=["Sensor_2", "Sensor_3", "Sensor_4", "Sensor_7", "Sensor_8", "Sensor_9", "Sensor_11
", "Sensor_12", "Sensor_13", "Sensor_14", "Sensor_15", "Sensor_17", "Sensor_20", "Sensor_21"],
                 hue="UnitNumber", aspect=1)
```

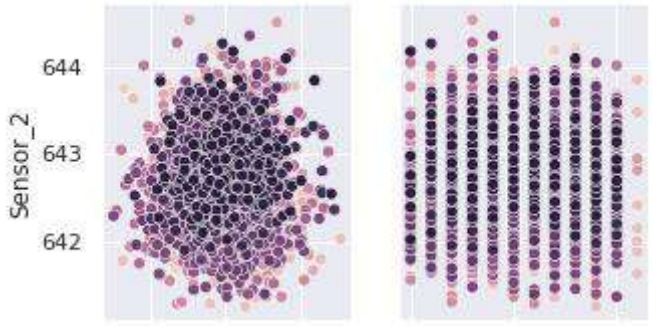


Figura 44. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 1

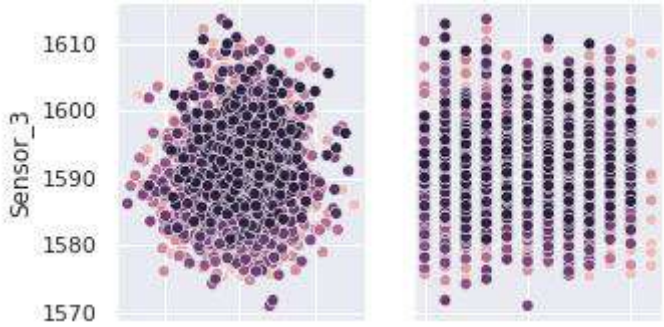


Figura 45. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 3

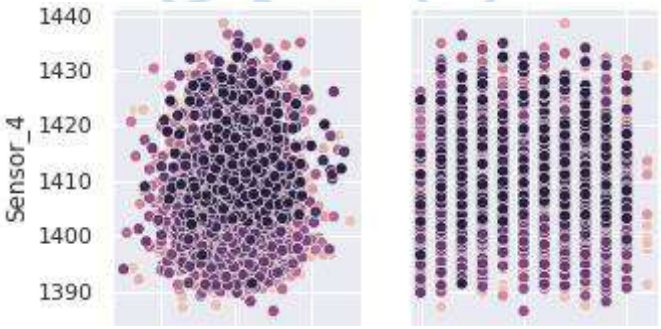


Figura 46. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 4

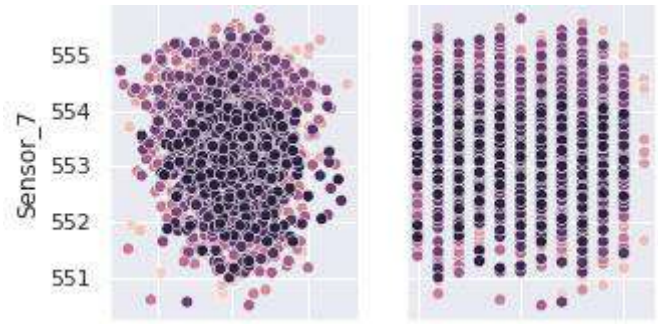


Figura 47. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor

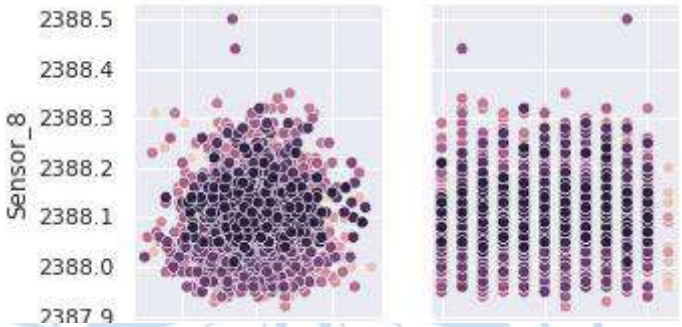


Figura 48. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 8

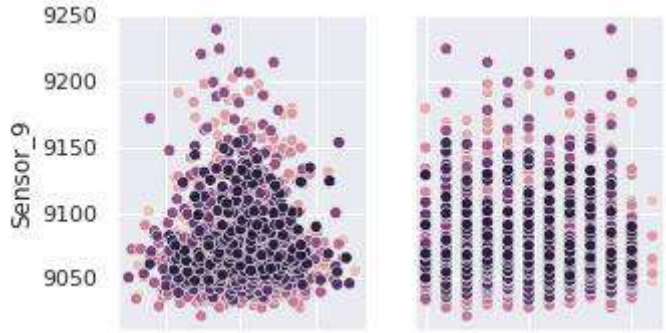


Figura 49. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 9

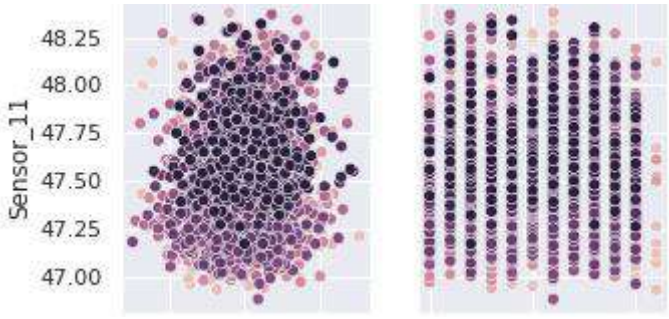


Figura 50. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 11

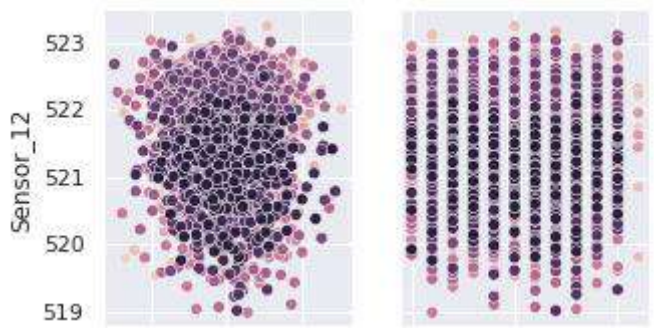


Figura 51. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 12

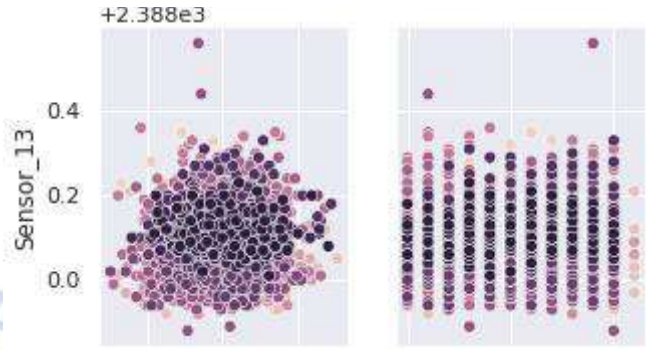


Figura 52. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 13

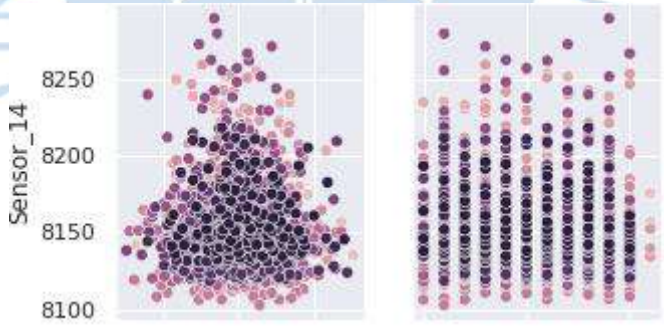


Figura 53. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 14

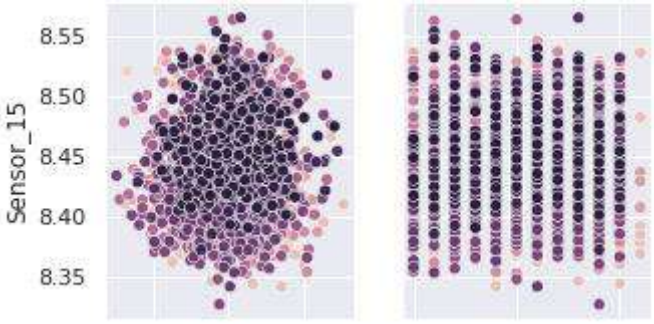


Figura 54. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 15

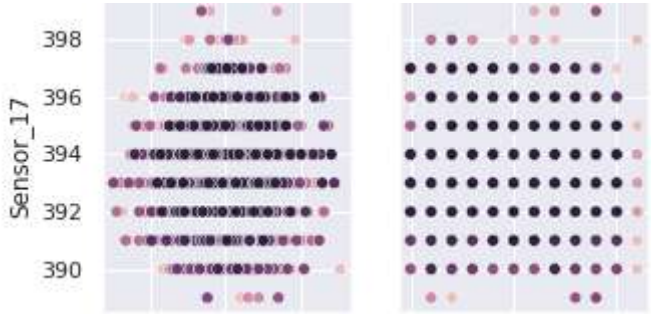


Figura 55. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 17

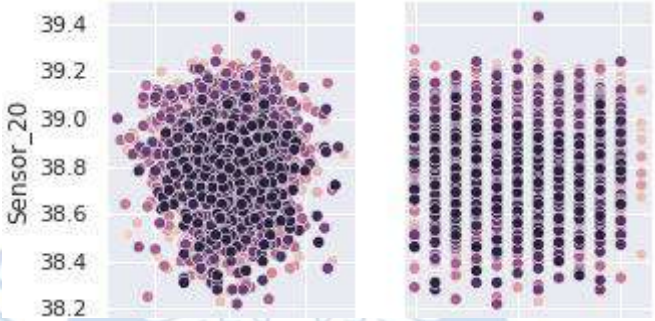


Figura 56. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 20

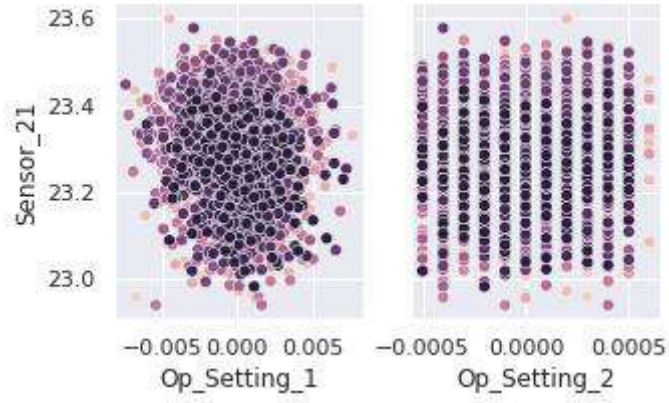


Figura 57. Gráfica de sensores vs. operaciones. Eje x operaciones 1 y 2, Eje y sensor 21

A continuación, se limpian los datos sobrantes para poder ajustar la data y se usa *random forest* para obtener la data más relevante. Así es como se crea un *random forest* de 200 árboles con 15 niveles de profundidad.

En esta parte del código se usan los comandos *train_no_drop* para eliminar lo que está y el comando *train_no_leakage* para la nueva data redimensionada. Por otro lado, el comando *print* se encarga de imprimir la data redimensionada. La variable y menos *unit number*, *cycle* y *op settings* y la variable *x* contiene todo lo que no contenga el URL (elimina el *target useful lifes*).

```
# now it's time to clear out target leakage
    print(train_with_target.shape)
    leakage_to_drop = ['UnitNumber', 'Cycle', 'Op_Setting_1', 'Op_Setting_2', 'Op_Setting_3']
    train_no_leakage = train_with_target.drop(leakage_to_drop, axis = 1)
    print(train_no_leakage.shape)
    # set up features and target variable
    y = train_no_leakage['Target_Remaining_Useful_Life']
      X = train_no_leakage.drop(['Target_Remaining_Useful_Life'], axis = 1)
```

En la siguiente parte del código, se usa el comando *esemble* para poder armar un *random forest* y así determinar las características más significativas con un número de árboles. Mientras que el n_estimadores=200 es el número de árboles completos y su comando max_depth = 15 indica la máxima oportunidad.

Se hace un entrenamiento y predicción de acuerdo a x. Al correr este código se indica al final que cuando haya terminado de correr escriba complete.

```
    # I like to use a simple random forest to determine some of the most important/meaningful feature
    s. Can be used as feature selection
    # create an exhuastive random forest (200 trees up to 15 levels deep)
    from sklearn import ensemble
    rf = ensemble.RandomForestRegressor()
    single_rf = ensemble.RandomForestRegressor(n_estimators = 200, max_depth = 15)
    single_rf.fit(X, y)
    y_pred = single_rf.predict(X)
    print("complete")
```

Esta parte del código indica la importancia de cada gráfico, al correr se obtiene que el sensor 18 no varía, el sensor con más relevancia es el 11 sin embargo su índice es el 10. Usa el comando *importance* para indicar las características de cada sensor, el comando *np.argsort* permite ordenar el nivel de importancia de la gráfica y el comando *plt.show* para mostrar la gráfica. Finalmente, el comando *important features* para indicar la lista de variables más importantes.

```
    # graph feature importance
    import matplotlib.pyplot as plt
    importances = single_rf.feature_importances_
    indices = np.argsort(importances)[::-1]
    feature_names = X.columns
    f, ax = plt.subplots(figsize=(11, 9))
    plt.title("Feature ranking", fontsize = 20)
    plt.bar(range(X.shape[1]), importances[indices], color="b", align="center")
    plt.xticks(range(X.shape[1]), indices) #feature_names, rotation='vertical')
    plt.xlim([-1, X.shape[1]])
    plt.ylabel("importance", fontsize = 18)
    plt.xlabel("index of the feature", fontsize = 18)
    plt.show()
```

```
# list feature importance
important_features = pd.Series(data=single_rf.feature_importances_,index=X.columns)
important_features.sort_values(ascending=False,inplace=True)
print(important_features.head(10))
```

Se procede a graficar la data más relevante:

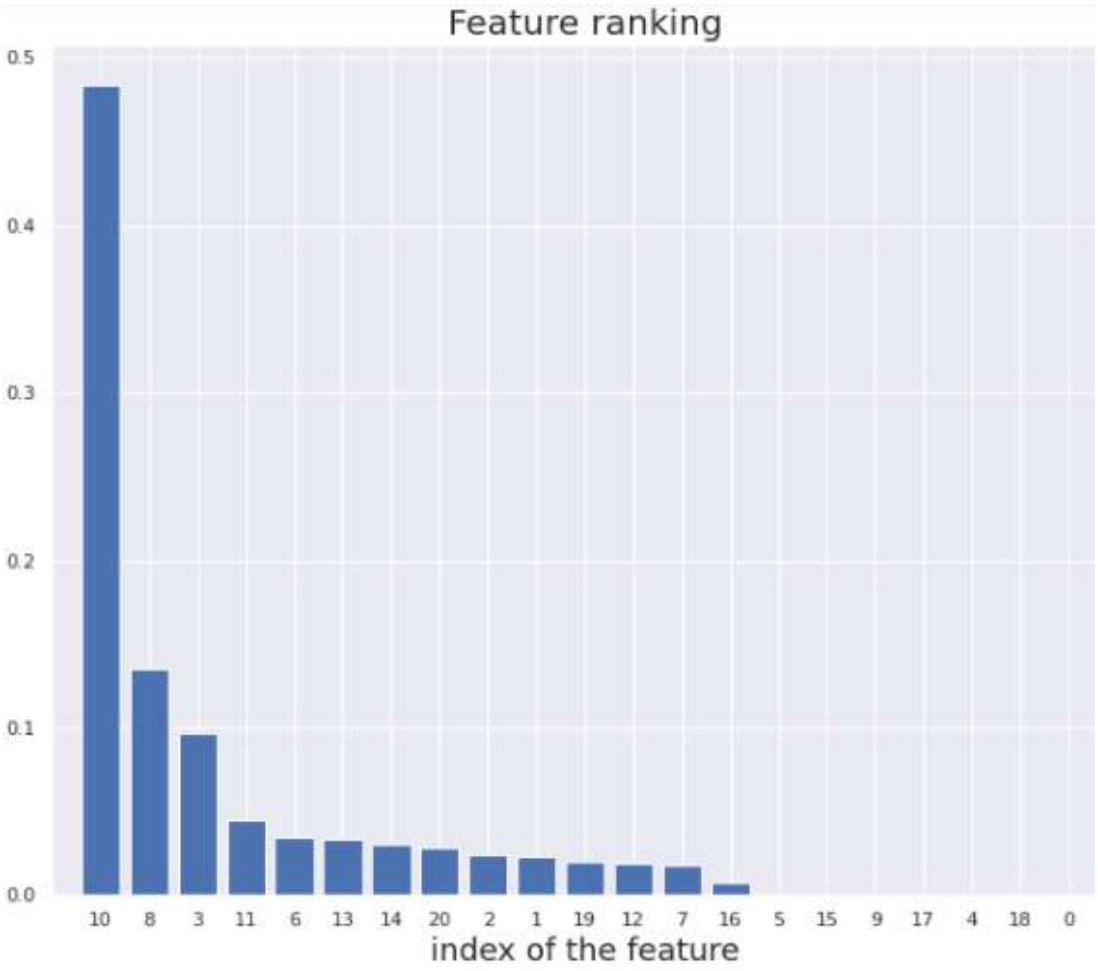


Figura 58. Gráfica en relación a la importancia

```
Sensor_11    0.483372
Sensor_9     0.135095
Sensor_4     0.097162
Sensor_12    0.044909
Sensor_7     0.034436
Sensor_14    0.033587
Sensor_15    0.030271
Sensor_21    0.027977
Sensor_3     0.024597
Sensor_2     0.023624
dtype: float64
```

Figura 59. Sensores más relevantes

Esta parte del código se encarga de imprimir los primeros 10 valores ya que se consideran los más importantes. El comando *vars_to_drop* se usa para indicar las variables

que no sirven, el comando + str(i) se encarga de almacenar los valores sin importancia, el comando *train_final=train_no_leakage.drop* se usara para redimensionar porque la matriz indica que se eliminan 6 variables.

```
# based on the graphs as well as random forest feature importance, I will exclude sensors without
much valuable information
print(train_no_leakage.shape)
vars_to_drop = ["Sensor_"+str(i) for i in [5, 15, 9, 17, 4, 18]]
train_final = train_no_leakage.drop(vars_to_drop, axis = 1)
print(train_final.shape)
train_final.head(5)
```

Y se procede a excluir los sensores con poca información relevante:

```
(20631, 22)
(20631, 16)
```

	Sensor_1	Sensor_2	Sensor_3	Sensor_6	Sensor_7	Sensor_8	Sensor_10	Sensor_11
0	518.67	641.82	1589.70	21.61	554.36	2388.06	1.3	47.47
1	518.67	642.15	1591.82	21.61	553.75	2388.04	1.3	47.49
2	518.67	642.35	1587.99	21.61	554.26	2388.08	1.3	47.27
3	518.67	642.35	1582.79	21.61	554.45	2388.11	1.3	47.13
4	518.67	642.37	1582.85	21.61	554.00	2388.06	1.3	47.28

Figura 60. Sensor del 1 al 11 de la nueva data entrenada para excluir a los de poca información relevante

Sensor_12	Sensor_13	Sensor_14	Sensor_16	Sensor_19	Sensor_20	Sensor_21	Target_Remaining_Useful_Life
521.66	2388.02	8138.62	0.03	100.0	39.06	23.4190	191
522.28	2388.07	8131.49	0.03	100.0	39.00	23.4236	190
522.42	2388.03	8133.23	0.03	100.0	38.95	23.3442	189
522.86	2388.08	8133.83	0.03	100.0	38.88	23.3739	188
522.19	2388.04	8133.80	0.03	100.0	38.90	23.4044	187

Figura 61. Sensor del 12 al 21 de la nueva data entrenada para excluir a los de poca información relevante

En esta parte del código identifica los campos categóricos y numéricos, así mismo les asigna nombre. Tomando así la data más relevante.

```
# identify categorical and numeric fields
from sklearn import preprocessing
categorical = train_final.select_dtypes(include=['object'])
numeric = train_final.select_dtypes(exclude=['object'])
print(categorical.columns.values)
# create dummy variables (if any categorical fields)
for name, values in categorical.items():
  print(name)
```

```
    dummies = pd.get_dummies(values.str.strip(), prefix = name, dummy_na=True)
    numeric = pd.concat([numeric, dummies], axis=1)
# imputation (if any NULL values)
for name in numeric:
    print(name)
    if pd.isnull(numeric[name]).sum() > 0:
        numeric["%s_mi" % (name)] = pd.isnull(numeric[name])
        median = numeric[name].median()
        numeric[name] = numeric[name].apply(lambda x: median if pd.isnull(x) else x)
y = numeric['Target_Remaining_Useful_Life']
X = numeric.drop(['Target_Remaining_Useful_Life'], axis = 1)
```

Se realiza el entrenamiento usando *random forest*, donde el comando *test train* y *test size* se encargan de definir la división de los datos de entrenamiento y prueba. Mientras que el *k fold* se usará para la validación cruzada con un k=5 y *min_sample_leaf* se usa para saber la mínima cantidad de hojas en el *random forest* y el comando *max_depth* se usa para los valores diversos. Finalmente, el *Grind Search* para la búsqueda de los mejores hiperparámetros (profundidad, hojas, etc.)

```
# random forest regression
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn.ensemble import RandomForestRegressor
rf = ensemble.RandomForestRegressor()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', rf) ])
# tune the model
my_min_samples_leaf = [2, 10, 25, 50, 100]
my_max_depth = [7, 8, 9, 10, 11, 12]
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_rf = GridSearchCV(estimator=pipeline
                    , cv=cv
                    , param_grid =dict(model__min_samples_leaf = my_min_samples_leaf, model__max_de
pth = my_max_depth)
                    , scoring = 'neg_mean_squared_error'
                    , verbose = 1
```

```
                    , n_jobs = -1
                   )
optimized_rf.fit(X_train, y_train)
# show the best model estimators
print(optimized_rf.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_rf.predict(X_test)
print("Random Forest Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("Random Forest Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("Random Forest r-squared: ", r2_score(y_test, y_pred))

Fitting 5 folds for each of 30 candidates, totalling 150 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed:  2.5min
[Parallel(n_jobs=-1)]: Done 150 out of 150 | elapsed:  9.0min finished
Pipeline(memory=None,
     steps=[('standardize',
          StandardScaler(copy=True, with_mean=True, with_std=True)),
         ('model',
          RandomForestRegressor(bootstrap=True, ccp_alpha=0.0,
                        criterion='mse', max_depth=10,
                        max_features='auto', max_leaf_nodes=None,
                        max_samples=None,
                        min_impurity_decrease=0.0,
                        min_impurity_split=None,
                            min_samples_leaf=2, min_samples_split=2,
                            min_weight_fraction_leaf=0.0,
                        n_estimators=100, n_jobs=None,
                        oob_score=False, random_state=None,
                        verbose=0, warm_start=False))],
     verbose=False)
Random Forest Mean Squared Error:  1772.264806735246
Random Forest Mean Absolute Error:  29.89143812976181
Random Forest r-squared:  0.6250111062357075
```

En esta parte del código se usa *Elastic Net GLM* para poder realizar la respectiva regresión y así poder aplicar la validación cruzada con regularización de L1 y L2. Se definen todos los parámetros y se imprimen los parámetros del mejor estimador. Aquí también se imprime el error cuadrático medio y el error absoluto medio. El comando pipeline contiene el modelo de regresión.

```
# Elastic Net GLM
# create holdout
import numpy as np
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=1234)
```

```
# choose the model
from sklearn.linear_model import ElasticNet
glm_net = ElasticNet()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', glm_net) ])
# tune the model
my_alpha = np.linspace(.01, 1, num=5)
my_l1_ratio = np.linspace(.01, 1, num=3)
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_glm_net = GridSearchCV(estimator=pipeline
                    , cv=cv
                    , param_grid =dict(model__l1_ratio = my_l1_ratio, model__alpha = my_alpha)
                    , scoring = 'neg_mean_squared_error'
                    , verbose = 1
                    , n_jobs = -1
    optimized_glm_net.fit(X_train, y_train)
# show the best model estimators
print(optimized_glm_net.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_glm_net.predict(X_test)
print("GLM Elastic Net Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("GLM Elastic Net Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("GLM Elastic Net r-squared: ", r2_score(y_test, y_pred))
Fitting 5 folds for each of 15 candidates, totalling 75 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
Pipeline(memory=None,
      steps=[('standardize',
           StandardScaler(copy=True, with_mean=True, with_std=True)),
          ('model',
           ElasticNet(alpha=0.01, copy_X=True, fit_intercept=True,
                 l1_ratio=0.01, max_iter=1000, normalize=False,
                 positive=False, precompute=False, random_state=None,
                 selection='cyclic', tol=0.0001,
                 warm_start=False))],
      verbose=False)
GLM Elastic Net Mean Squared Error:  2043.0311342906373
```

```
GLM Elastic Net Mean Absolute Error:  34.60051745516692
GLM Elastic Net r-squared:  0.5677203643258366
[Parallel(n_jobs=-1)]: Done 75 out of  75 | elapsed:    1.6s
```

Se procede ahora a usar el *Support Vector Machine* para seguir comparando el r cuadrado y encontrar el mejor algoritmo.

```
 Support Vector Machines
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn import svm
from sklearn.svm import SVR
svm = svm.SVR()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', svm) ])
# tune the model
my_C = [1]
my_epsilon = [.05, .1, .15]
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_svm = GridSearchCV(estimator=pipeline
                     , cv=cv
                     , param_grid =dict(model__C = my_C, model__epsilon = my_epsilon)
                     , scoring = 'neg_mean_squared_error'
                     , verbose = 1
                     , n_jobs = -1
                    )
optimized_svm.fit(X_train, y_train)
# show the best model estimators
print(optimized_svm.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_svm.predict(X_test)
print("SVM Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("SVM Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("SVM r-squared: ", r2_score(y_test, y_pred))
```

```
Fitting 5 folds for each of 3 candidates, totalling 15 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  15 out of  15 | elapsed:  2.6min finished
Pipeline(memory=None,
     steps=[('standardize',
          StandardScaler(copy=True, with_mean=True, with_std=True)),
         ('model',
            SVR(C=1, cache_size=200, coef0=0.0, degree=3, epsilon=0.15,
           gamma='scale', kernel='rbf', max_iter=-1, shrinking=True,
           tol=0.001, verbose=False))],
     verbose=False)
SVM Mean Squared Error:  1856.8127708430852
SVM Mean Absolute Error:  30.282585144178867
SVM r-squared:  0.6071218227548569
```

Igualmente se procede a programar con el algoritmo de *Gradient Boosting* para evaluar el r cuadrado y compararlo con los obtenidos previamente.

Gradient Boosting

```
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn.ensemble import GradientBoostingRegressor
gb = ensemble.GradientBoostingRegressor()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', gb) ])
# tune the model
my_alpha = [.5, .75, .9]
my_n_estimators= [500]
my_learning_rate = [0.005, .01]
my_max_depth = [4, 5, 6]
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_gb = GridSearchCV(estimator=pipeline
                    , cv=cv
                    , param_grid =dict(model__max_depth = my_max_depth, model__n_estimators = my_n
_estimators,
```

```
                        model__learning_rate = my_learning_rate, model__alpha = my_alpha)
                  , scoring = 'neg_mean_squared_error'
                  , verbose = 1
                  , n_jobs = -1
                  )
optimized_gb.fit(X_train, y_train)
# show the best model estimators
print(optimized_gb.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_gb.predict(X_test)
print("Gradient Boosting Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("Gradient Boosting Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("Gradient Boosting r-squared: ", r2_score(y_test, y_pred))
Fitting 5 folds for each of 18 candidates, totalling 90 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed: 13.4min
[Parallel(n_jobs=-1)]: Done  90 out of  90 | elapsed: 26.2min finished
Pipeline(memory=None,
     steps=[('standardize',
          StandardScaler(copy=True, with_mean=True, with_std=True)),
         ('model',
          GradientBoostingRegressor(alpha=0.75, ccp_alpha=0.0,
                                    criterion='friedman_mse', init=None,
                                    learning_rate=0.01, loss='ls',
                                    max_depth=6, max_features=None,
                                    max_leaf_nodes=None,
                                    min_impurity_decrease=0.0,
                                    min_impurity_split=None,
                                    min_samples_leaf=1,
                                    min_samples_split=2,
                                    min_weight_fraction_leaf=0.0,
                                    n_estimators=500,
                                    n_iter_no_change=None,
                                    presort='deprecated',
                                    random_state=None, subsample=1.0,
                                    tol=0.0001, validation_fraction=0.1,
                                    verbose=0, warm_start=False))],
     verbose=False)
Gradient Boosting Mean Squared Error:  1768.620476402149
Gradient Boosting Mean Absolute Error:  29.921852278069807
Gradient Boosting r-squared:  0.6257822005975242
```

Luego de haber identificado los sensores más relevantes, se compara con los diversos algoritmos mencionados anteriormente, obteniendo así la gráfica con la predicción del tiempo de vida útil restante vs. el predicho.

```
# plot actual vs predicted Remaining Useful Life for the best model (GBM)
fig, ax = plt.subplots()
ax.scatter(y_test, y_pred, edgecolors=(0, 0, 0))
ax.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
ax.set_xlabel('Actual RUL')
ax.set_ylabel('Predicted RUL')
ax.set_title('Remaining Useful Life Actual vs. Predicted')
plt.show()
```

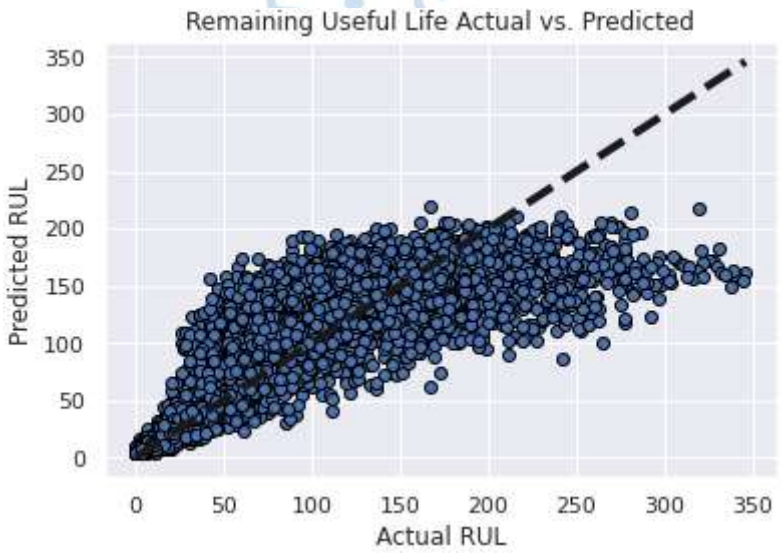


Figura 62. Gráfica de regresión del random foresr y la importancia de cada sensor. Predicted RUL vs. Actual RUL

Ahora se verá en como convertir los resultados previos en una solución de clasificación. ¿Se podrá identificar con seguridad cuándo un activo dentro de sus últimos 15 ciclos?

Se genera una nueva etiqueta para una nueva data de entrenamiento de manera binaria (1,0). Si es menor a 15 ciclos = 1, de lo contrario = 0

En esta parte del código se agregan los sensores previamente obviados.

	Sensor_1	Sensor_2	Sensor_3	Sensor_4	Sensor_5	Sensor_6	Sensor_7
20626	518.67	643.49	1597.98	1428.63	14.62	21.61	551.43
20627	518.67	643.54	1604.50	1433.58	14.62	21.61	550.86
20628	518.67	643.42	1602.46	1428.18	14.62	21.61	550.94
20629	518.67	643.23	1605.26	1426.53	14.62	21.61	550.68
20630	518.67	643.85	1600.38	1432.14	14.62	21.61	550.79

Figura 63. Sensores del 1 al 7 con nueva etiqueta para nueva data de entrenamiento

Sensor_8	Sensor_9	Sensor_10	Sensor_11	Sensor_12	Sensor_13	Sensor_14
2388.19	9065.52	1.3	48.07	519.49	2388.26	8137.60
2388.23	9065.11	1.3	48.04	519.68	2388.22	8136.50
2388.24	9065.90	1.3	48.09	520.01	2388.24	8141.05
2388.25	9073.72	1.3	48.39	519.67	2388.23	8139.29
2388.26	9061.48	1.3	48.20	519.30	2388.26	8137.33

Figura 64. Sensores del 8 al 14 con nueva etiqueta para nueva data de entrenamiento

Sensor_15	Sensor_16	Sensor_17	Sensor_18	Sensor_19	Sensor_20	Sensor_21
8.4956	0.03	397	2388	100.0	38.49	22.9735
8.5139	0.03	395	2388	100.0	38.30	23.1594
8.5646	0.03	398	2388	100.0	38.44	22.9333
8.5389	0.03	395	2388	100.0	38.29	23.0640
8.5036	0.03	396	2388	100.0	38.37	23.0522

Figura 65. Sensores del 15 al 21 con nueva etiqueta para nueva data de entrenamiento

Target_Remaining_Useful_Life	Target_15_Cycles
4	1
3	1
2	1
1	1
0	1

Figura 66. Vida útil restante y la etiqueta de 15 ciclos ya que, con más, fallaría el modelo

Al obtener los valores del tiempo de vida útil restante (RUL) muy dispersos y posteriormente aplicarlos los algoritmos de *Random Forest*, Validación cruzada, *Support Vector Machine* y *Gradient Boostin*. Teniendo como mejor resultado de precisión para el RUL el algoritmo de *Gradient Boostin* con un R cuadrado de 0.6257 ya que se aproxima más a 1.

A continuación, se procederá a generar el código del *Random Forest,* pero ahora se comparará los ciclos, los cuales no pueden ser mayor a 15 y al ser este el algoritmo más rápido se ha decidido trabajar con este algoritmo obteniendo un r cuadrado de 0.6607 en esta comparación de ciclos con una precisión de 82.0%

```
# random forest regression
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn import ensemble
from sklearn.ensemble import RandomForestClassifier
rf = ensemble.RandomForestClassifier()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                           , ('model', rf) ])
# tune the model
my_min_samples_leaf = [2, 25, 50]
my_max_depth = [8, 9, 10, 12]
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_rf = GridSearchCV(estimator=pipeline
```

```
                , cv=cv
                , param_grid =dict(model__min_samples_leaf = my_min_samples_leaf, model__max_de
pth = my_max_depth)
                , scoring = 'roc_auc'
                , verbose = 1
                    , n_jobs = -1
optimized_rf.fit(X_train, y_train)
# show the best model estimators
y_pred_proba = optimized_rf.predict_proba(X_test)[:, 1]
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_rf.predict(X_test)
print(optimized_rf.best_estimator_)
print("Random Forest Mean Squared Error 2: ", mean_squared_error(y_test, y_pred))
print("Random Forest Mean Absolute Error 2: ", mean_absolute_error(y_test, y_pred))
print("Random Forest r-squared 2: ", r2_score(y_test, y_pred))
Fitting 5 folds for each of 12 candidates, totalling 60 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed:   58.8s
[Parallel(n_jobs=-1)]: Done  60 out of  60 | elapsed:  1.3min finished
Pipeline(memory=None,
     steps=[('standardize',
          StandardScaler(copy=True, with_mean=True, with_std=True)),
         ('model',
          RandomForestClassifier(bootstrap=True, ccp_alpha=0.0,
                                 class_weight=None, criterion='gini',
                                 max_depth=9, max_features='auto',
                                 max_leaf_nodes=None, max_samples=None,
                                 min_impurity_decrease=0.0,
                                 min_impurity_split=None,
                                 min_samples_leaf=2, min_samples_split=2,
                                 min_weight_fraction_leaf=0.0,
                                 n_estimators=100, n_jobs=None,
                                 oob_score=False, random_state=None,
                                 verbose=0, warm_start=False))],
     verbose=False)
Random Forest Mean Squared Error 2:  0.02180760843227526
Random Forest Mean Absolute Error 2:  0.02180760843227526
  Random Forest r-squared 2:  0.6607852269925203
```

	precision	recall	f1-score	support
0	0.99	0.99	0.99	3842
1	0.82	0.87	0.85	285
accuracy			0.98	4127
macro avg	0.91	0.93	0.92	4127
weighted avg	0.98	0.98	0.98	4127

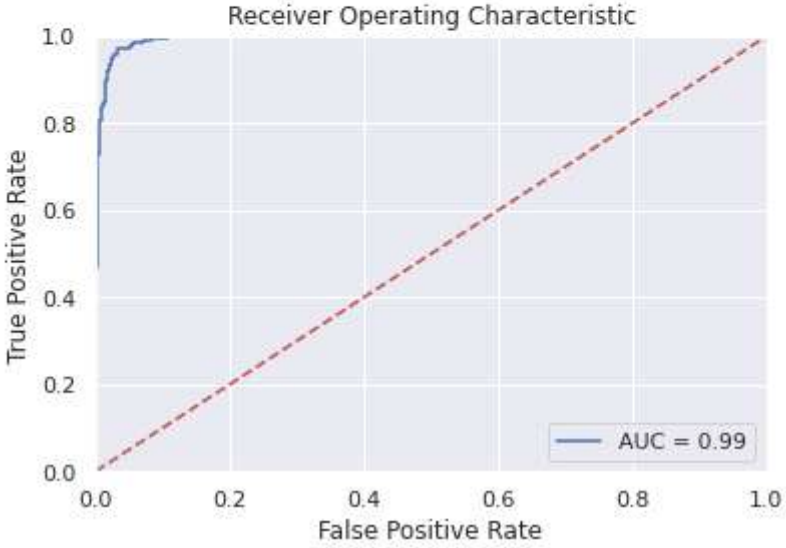


Figura 67. Gráfica final con la vida menor o igual a 15 ciclos, ya no se usa el RUL

Conclusiones

La existencia y desarrollo de grandes avances de la IA, la cual ha sido aplicada a sistemas de producción, han hecho que en el día a día la industria logre el objetivo de poder mejorar su competitividad, ya que es lo que se venía buscando constantemente. Sin embargo, en la mayoría de los casos de desplazar gran cantidad de mano de obra trae consigo un deterioro social, lo cual se puede ver reflejado en los indicadores de desempleo y pobreza.

Gracias a que se ha podido llevar a cabo el estudio del comportamiento inteligente de las máquinas, se ha podido crear y desarrollar una tecnología que permita crear y dar forma a diversas conductas que la conllevan a actuar como una persona. El plan es que a futuro lleguen a desarrollar diversas conductas incluso mejor que los humanos.

Es importante recalcar que la inclusión de la IA en el mantenimiento industrial es considerada como una herramienta que permite facilitar las operaciones, en cuanto a la reducción de tiempos para ejecución. Es una gran oportunidad para que los trabajadores profesionales puedan mejorar y desempeñarse en obtener un mejor conocimiento para los diversos activos que existen

Se ha logrado determinar que los datos se fragmentan entre organizaciones, especialmente cuando se retienen en sistemas aislados, además sin la implementación de la tecnología *blockchain*, la organización receptora debe confiar en el valor de los datos recibidos. A diferencia de que con *blockchain*, las partes involucradas pueden compartir datos en tiempo real, junto con el historial y las modificaciones al mismo.

En las industrias y empresas peruanas se recomienda empezar con la implementación de softwares relacionadas con la Inteligencia Artificial, ya que esto permitirá reducir costos y aumentar la producción usando menor mano de obra. Se puede ir empezando en pequeños activos, los más críticos, y poco a poco ir desarrollando un plan de mantenimiento preventivo que se pueda implementar de la mano con la Inteligencia Artificial.

Los constantes desarrollos en Big Data, la comunicación de máquina a máquina y la tecnología en la nube han abierto nuevas posibilidades para investigar la información derivada de los activos industriales. Es viable el monitoreo en tiempo real, gracias a los sensores, actuadores y otros parámetros de control.

El comando *grid search* permite realizar diversas combinaciones, desde la mínima cantidad de hojas y a su vez con 7. Así como en su máxima profundidad donde busca la mejor combinación para el mejor modelo. Se entrena con una *x* y *y* de entrenamiento. Se usa mucho ya que permite encontrar el mejor hiperparámetro.

El comando pipeline contiene el modelo de regresión, ya que imprime el mejor estimador para la red, lo cual permite comparar y llegar a la conclusión de que el valor r cuadrado mejor es el de *Gradient Boostin* se compara el RUL frente a los otros algoritmos.

En la comparación de los 15 ciclos el *Random Forest* es el que presenta mejores resultados al ser usado con una amplia base de datos, así como con data faltante ya que crea estimaciones de estas. Gracias a la ayuda de algunos algoritmos como el SVM se pudieron extrapolar los datos para poder realizar la evaluación final de los sensores vs la vida menor o igual a 15 ciclos.

Glosario de términos

1 Inteligencia Artificial: Programa de computación diseñado para realizar determinadas operaciones que se consideran propias de la inteligencia humana, como el autoaprendizaje.

2. *Big data*: Es un término que describe el gran volumen de datos, tanto estructurados como no estructurados, que inundan los negocios cada día. Pero no es la cantidad de datos lo que es importante. Lo que importa con el *Big Data* es lo que las organizaciones hacen con los datos. (Power Data, s.f.)

3. Redes Neuronales: Una red neuronal es un método de computación que simula el funcionamiento neuronal durante el aprendizaje. Uno de los retos de la inteligencia artificial es el de resolver problemas de reconocimiento de patrones. es decir, una de las capacidades en las que la maquina inteligente debe imitar al cerebro humano es en la capacidad de asociar un conjunto de rasgos con una idea o concepto. (Caparros, 1994)

4. Aprendizaje profundo: El aprendizaje profundo es una rama del aprendizaje automático. A diferencia de los algoritmos tradicionales de aprendizaje automático, muchos de los cuales tienen una capacidad finita de aprendizaje independientemente de cuántos datos adquieran, los sistemas de aprendizaje profundo pueden mejorar su rendimiento al poder acceder a un mayor número de datos, o lo que es lo mismo, hacer que la máquina tenga más experiencia. (NetApp, 2021)

5. Malware: es un término general para referirse a cualquier tipo de "*malicious software*" (software malicioso) diseñado para infiltrarse en su dispositivo sin su conocimiento. Hay muchos tipos de malware y cada uno busca sus objetivos de un modo diferente. Sin embargo, todas las variantes comparten dos rasgos definitorios: son subrepticios y trabajan activamente en contra de los intereses de la persona atacada. (Avast Academy, 2021)

UNIVERSITAS STUDIORUM PIURENSIS

Referencias bibliográficas

10Custom. (18 de setiembre de 2020). Obtenido de https://customprofessionalhosting.com/noticias/como-la-inteligencia-artificial-esta-mejorando-el-cloud-computing/

3R. Industria 4.0. (s.f.). Obtenido de https://industria40.me/realidad-virtual/

Aina Tecnología. (2012). Obtenido de https://www.ainia.es/tecnoalimentalia/tecnologia/las-6-ventajas-de-usar-el-cloud-computing-en-la-empresa-las-conoces/

Allmatic. (2012). *Projekt Knestel*. Projekte von Allmatic Spannsysteme.

Allmatic. (2019). *Gripp Version TITAN 2 M MECHANISCH*. Obtenido de https://www.allmatic.de/shop/produkte/ALLMATIC_GRIPP_Baureihen/Version_TITAN_2_K_M_L/Version_TITAN_2_M_MECHANISCH.html

Allmatic. (2019). *NC Version LCH 125 HYDRAULISCH*. Obtenido de https://www.allmatic.de/shop/produkte/Automatisierbares_Spannen/NC_HYDRAULISCHE_Version_LCH/NC_Version_LCH_125_HYDRAULISCH.html

Automated. (10 de octubre de 2018). Obtenido de https://www.euautomation.com/es/automated/article/de-que-forma-pueden-mejorar-los-robots-colaborativos

Avast Academy. (19 de mayo de 2021). Obtenido de https://www.avast.com/es-es/c-malware

Braun, S. J. (2021). Obtenido de http://sebastianbrau.com/la-importancia-de-la-hiperconectividad-en-la-industria-4-0/

Caparros, M. E. (1994). *Redes neuronales: concepto, fundamentos y aplicaciones*. Obtenido de https://www.seqc.es/download/revista/388/1224/1297816671/1024/cms/Qu%C3%ADmica%20Cl%C3%ADnica%201994;13%20(5)%20221-228.pdf/

Carretero Toro, F. J. (s.f.). Obtenido de https://desarrolloonline.com/wp-content/uploads/2018/10/Los-usos-de-la-blockchain-crecen.pdf

Chumán, J. (2017). *Diseño e implementación de un banco de pruebas automático para las mordazas mecánicas en Allmatic (tesis de licenciatura en Ingeniería Mecánico-Eléctrica).* Universidad de Piura, Facultad de Ingeniería. Programa Académico de Ingeniería Mecánico-Eléctrica. Piura, Perú.

Ciencias de la computación. (s.f.). Obtenido de https://sites.google.com/site/cienciasdelacompucion/inteligencia-artificial

Corredera, F. J. (10 de marzo de 2019). *Techedge*. Obtenido de https://www.techedgegroup.com/es/blog/inteligencia-artificial-mantenimiento-predictivo

Dynatec. (2020). Obtenido de https://dynatec.es/2020/07/13/la-fabricacion-aditiva-clave-en-la-industria-4-0/

EcuRed. (24 de noviembre de 2019). Obtenido de https://www.ecured.cu/M%C3%A1quina_herramienta

EN 692:2005+A1:2009. (2009). *Machine tools - Mechanical presses - Safety.* International Organization for Standardization.

Esneca Business School. (2019). *Esneca*. Obtenido de https://www.esneca.com/blog/robotica-colaborativa-que-es/

Espeso, J., Fernández, F., Espeso, M., & Fernández, B. (2008). *Seguridad en el Trabajo, Manual para la formación del especialista.* Valladolid: Lex Nova.

Factoría del Futuro. (28 de febrero de 2020). Obtenido de https://www.factoriadelfuturo.com/inteligencia-artificial-aprendizaje-automatico-y-mantenimiento-predictivo/

Familia, R. (9 de diciembre de 2014). Obtenido de https://www.unibe.edu.do/docentes/2014/12/09/la-logica-como-paradigma-de-la-programacion-en-inteligencia-artificial/

Gestión del mantenimiento cod 7969. (s.f.). Obtenido de https://sites.google.com/site/gestiondelmantenimientocod7969/mantenimiento-no-planeado

Gomeringer, R. (2014). *Tabellenbuch Metall.* Haan: Europa-Lehrmittel.

Gonzalez, A. A. (24 de marzo de 2020). *YouTube*. Obtenido de https://www.youtube.com/watch?v=bjgZczRUL-s

Goodfellow, I. B. (s.f.). Deep Feedforward Networks". In: "Deep Learning".

Gräf, W. (1997). *Maschinensicherheit.* Heidelberg: Hüthig.

Grupo IGN. (8 de abril de 2019). *Grupo IGN*. Obtenido de https://ignsl.es/fabricacion-aditiva/#:~:text=Qu%C3%A9%20es%20la%20Fabricaci%C3%B3n%20Aditiva,en%20base%20a%20planos%20virtuales.

Grupo Retailgas. (04 de julio de 2020). Obtenido de https://retaintechnologies.com/usos-de-la-realidad-virtual-y-aumentada-en-mantenimiento-de-activos/

Hinton, G. E. (2006). *Reducing the dimensionality of data with neural networks.*

History & Maps. (24 de noviembre de 2019). Obtenido de https://www.lahistoriaconmapas.com/historia/historia2/definicion-de-maquina-herramienta/

Ibérica Multimedia. (25 de abril de 2008). Obtenido de https://ibericamultimedia.com/cloud-computing-computacion-la-nube/

ICEX España Exportación e Inversiones. (2018). *Estudio de Mercado: El mercado de la máquina-herramienta en Alemania.* Resumen Ejecutivo, Berlín.

ICEX España Exportación e Inversiones. (2019). *Informe de Feria: Hannover 2019.* Dusseldorf.

Idea Ingeniería. (s.f.). Obtenido de https://ideaingenieria.es/realidad-aumentada-virtual/realidad-aumentada-mantenimiento-industrial/

IDEA Ingeniería. (s.f.). Obtenido de https://ideaingenieria.es/realidad-aumentada-virtual/realidad-aumentada-mantenimiento-industrial/

InNeurociencia. (8 de octubre de 2018). *Brain Investigations*. Obtenido de https://www.brainvestigations.com/neurociencia/inteligencia-artificial-tandem/

Ipargama. (2019). *16MnCr5-Acero para cementar.* Obtenido de http://www.ipargama.com/pdf/16MnCr5.pdf

ISO 12100:2010. (2010). *Safety of machinery — General principles for design — Risk assessment and risk reduction.*

ISO 13857:2019. (2019). *Safety of machinery — Safety distances to prevent hazard zones being reached by upper and lower limbs.* International Organization for Standardization.

ITCL. (18 de setiembre de 2018). Obtenido de https://itcl.es/blog/aplicaciones-de-big-data-en-la-industria/

Lopez, J. (s.f.). *Factoria del Futuro*. Obtenido de https://www.factoriadelfuturo.com/tecnologias-habilitadoras

Mancuzo, G. (17 de setiembre de 2020). *Compara Software*. Obtenido de https://blog.comparasoftware.com/evolucion-del-mantenimiento/

Maxon motor. (2018). *ESCON Module 50/8 HE Hardware Reference.* Obtenido de https://www.maxongroup.ch/medias/sys_master/root/8834400550942/532872-586137-ESCON-Module-50-8-Hardware-Reference-En.pdf

Maxon motor. (2018). *ESCON Overview Catalog Page.* Obtenido de https://www.maxongroup.ch/medias/sys_master/root/8831198232606/2018EN-442-443-446.pdf

Maxon Motor. (2019). *Micromotores y sistemas de alta precisión.* Obtenido de http://maxon.blaetterkatalog.ch/b9990/catalog/index.html?data=b9990/b999054

McGeough, J. A., & McCarthy, W. J. (23 de noviembre de 2019). *Encyclopedia Britannica*. Obtenido de https://www.britannica.com/technology/machine-tool

McKechnie, G. (25 de noviembre de 2019). *Encyclopaedia Britannica*. Obtenido de https://www.britannica.com/technology/vise

Medina Nuñez, R. J. (3 de setiembre de 2018). *LinkedIn*. Obtenido de https://www.linkedin.com/pulse/mantenimiento-basado-en-condici%C3%B3n-mbc-o-riesgo-mbr-robinson-medina?originalSubdomain=es

Moreno, C. G. (s.f.). *INDRA*. Obtenido de https://www.indracompany.com/es/blogneo/deep-learning-sirve

NetApp. (2021). Obtenido de https://www.netapp.com/es/artificial-intelligence/what-is-deep-learning/

Neugart. (2017). *Precision gearbox catalog.* Obtenido de https://www.neugart.com/fileadmin/user_upload/Downloads/Product_Catalogs/Neugart-Product-Catalog-DE.pdf

Niemann, G. (2005). *Maschinenelemente, Band 1: Konstruktion und Berechnung von Verbindungen, Lagern, Wellen.* Berlín: Springer.

OBS Business School . (27 de setiembre de 2018). Obtenido de https://obsbusiness.school/es/blog-investigacion/operaciones/la-automatizacion-industrial-y-los-sistemas-ciberfisicos

Ocerín, J. M. (s.f.). *Helvia*. Obtenido de https://helvia.uco.es/xmlui/bitstream/handle/10396/6671/braco124_1993_7.pdf?sequence=1&isAllowed=y

Parlamento Europeo; Consejo de la Unión Europea. (s.f.). *Directiva 2006/42/CE.* Diario Oficial de la Unión Europea.

Pingyuan Zhenghao Machinery Co., Ltd. (2019). *Precision Universal Vise*. Obtenido de https://www.pyzhjx.com/Precision-Universal-Vise-6-91-1.html

Pistarelli, A. J. (2010). *Manual de Mantenimiento: Ingeniería, Gestión y Organización.* Buenos Aires.

PJRC. (2019). *Teensy 3.2*. Obtenido de https://www.pjrc.com/teensy/teensy31.html#specs

PodCast Industria 4.0. (26 de Marzo de 2016). Obtenido de https://www.podcastindustria40.com/sistemas-ciber-fisicos/

Ponce-Mostacero, A. (2018). *Optimización del Mantenimiento Planeado en una línea de producción de bebidas carbonatadas.* Piura.

Power Data. (s.f.). Obtenido de https://www.powerdata.es/big-data

Quinde, J. (2018). Dirección de proyectos. *Visión general.* Universidad de Piura.

R, M. A. (16 de abril de 2020). *Express Computer*. Obtenido de https://www.expresscomputer.in/guest-blogs/how-can-blockchain-add-value-in-enterprise-maintenance-services/53196/#:~:text=It%20enables%20organizations%20to%20trace,securely%20in%20a%20distributed%20ledger.

Redacción España. (7 de abril de 2020). *B12 Tech Business*. Obtenido de https://agenciab12.pe/noticia/como-funcionan-algoritmos-inteligencia-artificial

ROEMHELD Gruppe. (2017). *HILMA Turmspannsystem "TS 125" in der Horizontalbearbeitung*. Obtenido de Youtube: https://www.youtube.com/watch?v=bfLIzfoxF2s

Sainath, T. N. (2013). "Deep convolutional neural networks for LVCSR.". *"Deep convolutional neural networks for LVCSR."Acoustics, Speech and Signal Processing*. 2013 IEEE International Conference on. IEEE.

Smart Panel. (s.f.). Obtenido de https://www.smartpanel.com/que-es-deep-learning/

Software y Soluciones de Analítica. (s.f.). Obtenido de https://www.sas.com/es_pe/insights/analytics/deep-learning.html

Statista. (2019). *Ventas nacionales e internacionales de la ingeniería mecánica alemana entre los años 2008 y 2018.* Obtenido de https://de.statista.com/statistik/daten/studie/235563/

Sutskever, I. J. (2011). *"Generating text with recurrent neural networks".Proceedings of the 28th International Conference on Machine Learning.*

Thyssenkrupp. (2019). *Ficha técnica-Aleación de aluminio EN AW-6060.* Obtenido de https://es.materials4me.com/media/pdf/f9/e2/dc/ficha-tecnica_calidad_EN-AW-6060_espanol.pdf

Tikkanen, A. (25 de noviembre de 2019). *Encyclopaedia Britannica*. Obtenido de https://www.britannica.com/technology/vise

TMV. (s.f.). http://www.tmv.com.mx/servicio-de-mantenimiento-proactivo.html.

Ultimaker. (2019). *Ficha de datos técnicos ABS.* Obtenido de https://ultimaker.com/download/67619/TDS%20ABS%20v3.011-spa-ES.pdf

Ultimaker. (2019). *Ficha de datos técnicos PLA.* Obtenido de https://ultimaker.com/download/67583/TDS%20PLA%20v3.011-spa-ES.pdf

Universidad de Granada. (20 de abril de 2020). Obtenido de https://canal.ugr.es/noticia/investigadores-ugr-uja-desarrollan-software-ministerio-defensa-sistemas-de-inteligencia-artificial/

Universidad Internacional de Valencia. (21 de marzo de 2018). Obtenido de https://www.universidadviu.com/int/actualidad/nuestros-expertos/que-es-un-dron-y-como-funciona

UNNE. (s.f.). *Universidad Nacional del Nordeste*. Obtenido de http://ing.unne.edu.ar/pub/informatica/Alg_diag.pdf

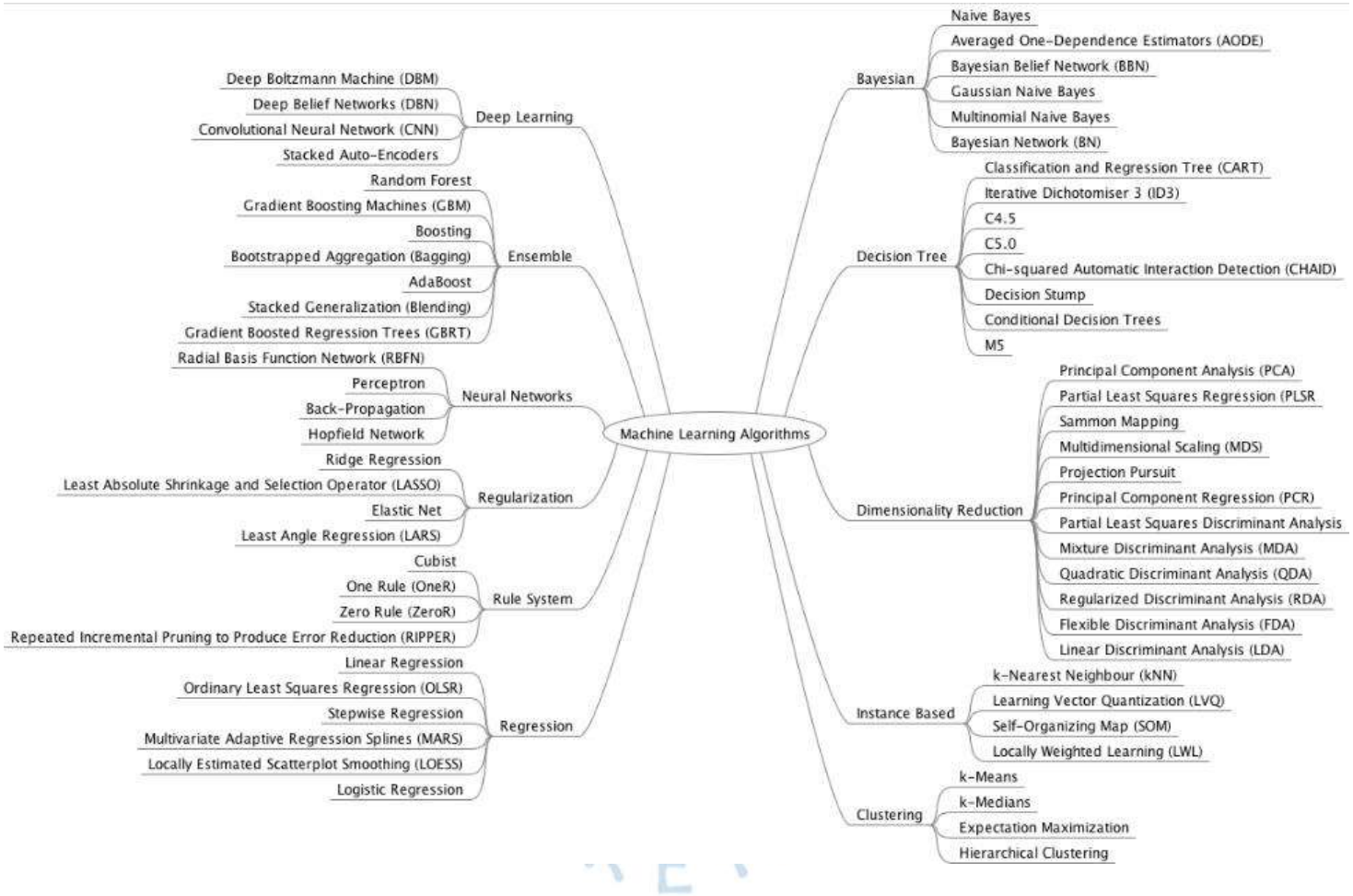
Vasquez, A. (2019). *Desarrollo de un sistema RF para monitorear la fuerza en una mordaza industrial.* Universidad de Piura, Facultad de Ingeniería. Programa Académico de Ingeniería Mecánico-Eléctrica. Piura, Perú.

ZwickRoell. (2019). *Mordazas hidráulicas*. Obtenido de https://www.zwickroell.com/es-es/mordazas/mordazas-hidráulicas

Anexos

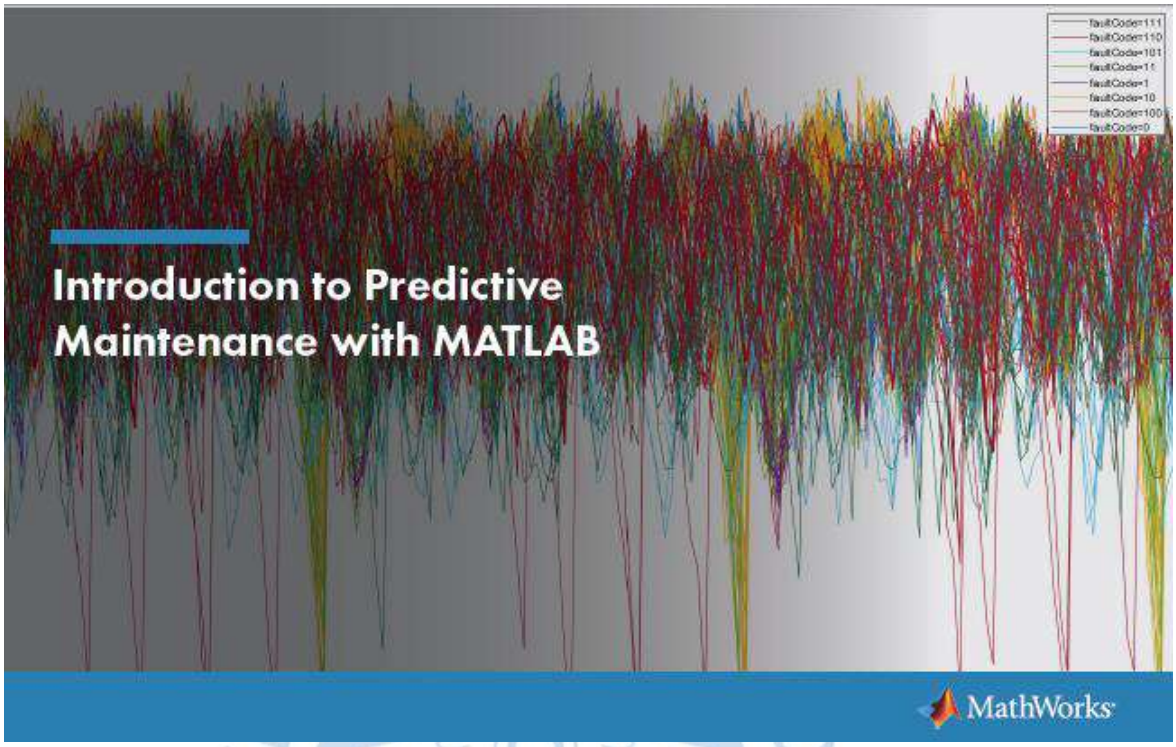
UNIVERSITAS STUDIORUM PIURENSIS

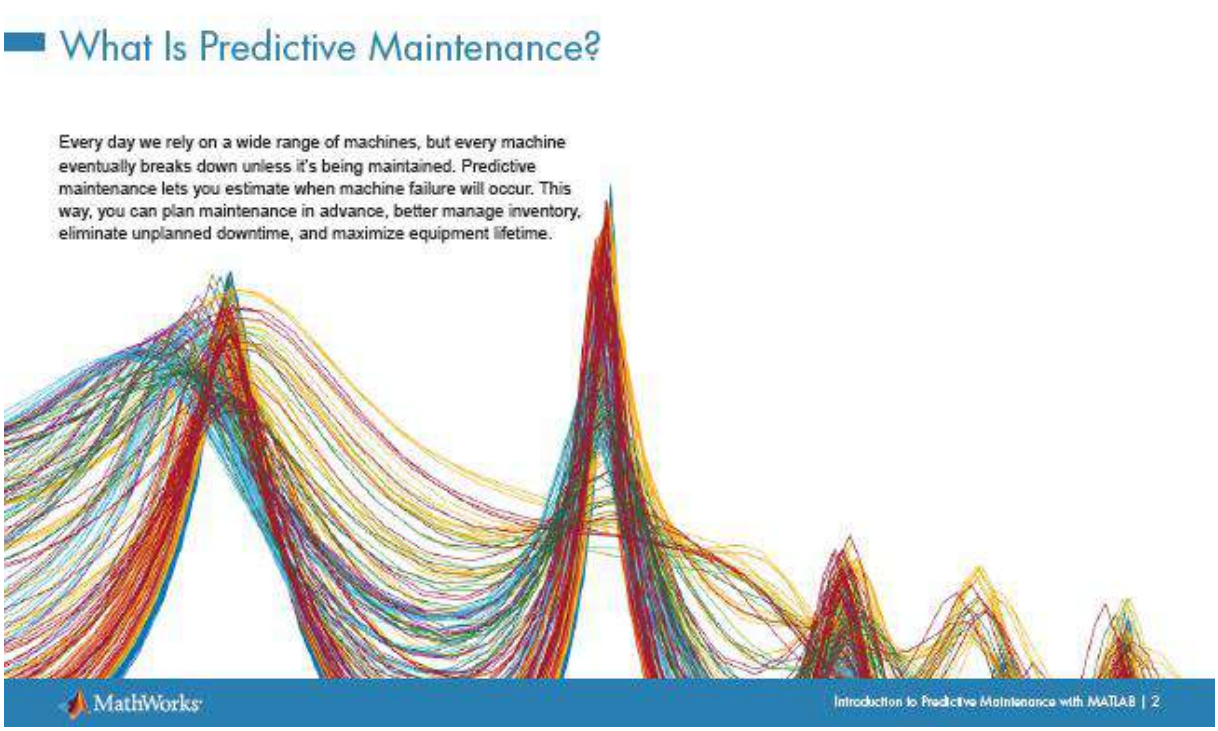
Anexo A. Mapa de todos los tipos de algoritmo de *machine learning* y su respectiva clasificación



Fuente: (Cervan, 2020)

Anexo B. Introducción al mantenimiento predictivo con MATLAB



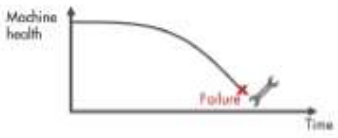


Fuente: (Matlab, s.f.)

Maintenance Strategies

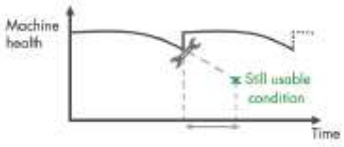
Reactive Maintenance

With reactive maintenance, the machine is used to its limit and repairs are performed only after the machine fails. If you're maintaining an inexpensive system like a light bulb, the reactive approach may make sense. But think of a complex system with some very expensive parts, such as an aircraft engine. You can't risk running it to failure, as it will be extremely costly to repair highly damaged parts. But, more importantly, it's a safety issue.



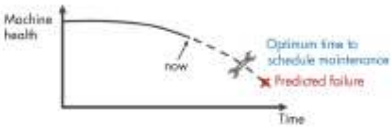
Preventive Maintenance

Many organizations try to prevent failure before it occurs by performing regular checks on their equipment. One big challenge with preventive maintenance is determining when to do maintenance. Since you don't know when failure is likely to occur, you have to be conservative in your planning, especially if you're operating safety-critical equipment. But by scheduling maintenance very early, you're wasting machine life that is still usable, and this adds to your costs.



Predictive Maintenance

Predictive maintenance lets you estimate time-to-failure of a machine. Knowing the predicted failure time helps you find the optimum time to schedule maintenance for your equipment. Predictive maintenance not only predicts a future failure, but also pinpoints problems in your complex machinery and helps you identify what parts need to be fixed.



Getting Started with Predictive Maintenance

Implementing predictive maintenance helps reduce downtime, optimize spare parts inventory, and maximize equipment lifetime. But how do you get started? First you need to develop an algorithm that will predict a time window, typically some number of days, when your machine will fail and you need to perform maintenance. Let's look at the predictive maintenance workflow to get started on algorithm development.

Data

Predictive Maintenance Algorithm

Failure in 20±2 days

Fuente: (Matlab, s.f.)

Predictive Maintenance Workflow at a Glance

Algorithm development starts with **data** that describes your system in a range of healthy and faulty conditions. The raw data is **preprocessed** to bring it to a form from which you can extract **condition indicators**. These are features that help distinguish healthy conditions from faulty. You can then use the extracted features to **train a machine learning model** that can:

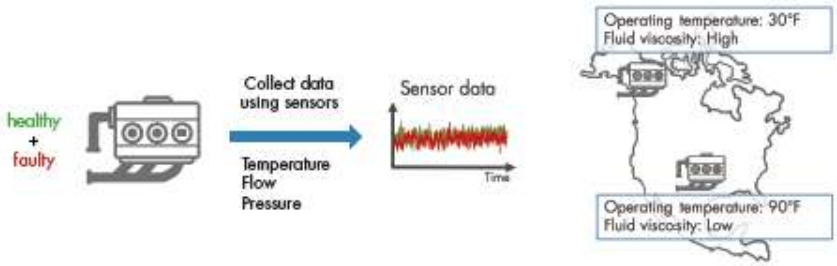
- Detect anomalies
- Classify different types of faults
- Estimate the remaining useful life (RUL) of your machine

Finally, you **deploy** the algorithm and **integrate** it into your systems for machine monitoring and maintenance.

In the next sections, we use a triplex pump example to walk through the workflow steps. Triplex pumps are commonly used in the oil and gas industry.

Acquire Data

The first step is to **collect a large set of sensor data** representing **healthy** and **faulty operation**. It's important to collect this data under varying operating conditions. For example, you may have same type of pumps running in different places, one in Alaska and the other one in Texas. One may be pumping highly viscous fluid whereas the other one operates with low-viscosity fluid. Although you have the same type of pump, one may fail sooner than the other due to these different **operating conditions**. Capturing all this data will help you develop a **robust algorithm** that can better detect faults.



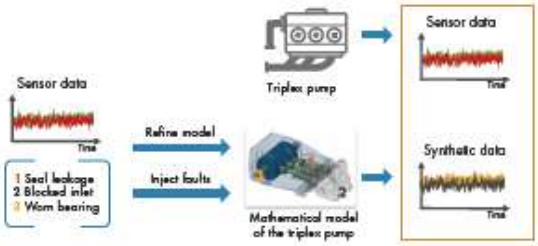
Note: For simplicity in this example, healthy and faulty operations are each represented by a single measurement. In a real-world scenario, there may be hundreds of measurements for both types of conditions.

Fuente: (Matlab, s.f.)

Acquire Data - Continued

In some cases, you may not have enough data representing healthy and faulty operation. As an alternative, you can build a **mathematical model of the pump** and estimate its parameters from sensor data. You can then simulate this model with different fault states under varying operating conditions to **generate fault data.** This data, also referred to as **synthetic data**, now supplements your sensor data. You can use a combination of synthetic and sensor data to develop your predictive maintenance algorithm.

Learn more: *Generate fault data with Simulink*



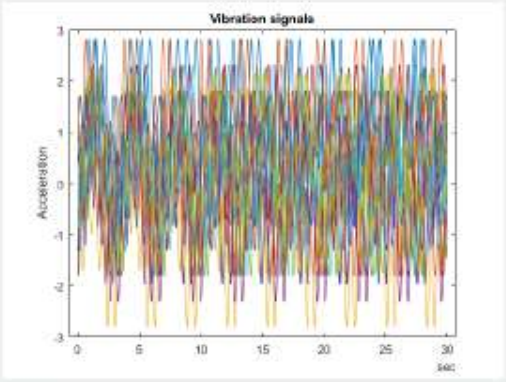
Simulating the model under different fault states to generate fault data.

Acquire Data with MATLAB

Data from equipment can be structured or unstructured, and reside in multiple sources such as local files, the cloud (e.g., AWS® S3, Azure® Blob), databases, and data historians. Wherever your data is, you can get to it with MATLAB®. When you don't have enough failure data, you can generate it from a Simulink® model of your machine equipment by injecting signal faults, and modeling system failure dynamics.

> *MATLAB gave us the ability to convert previously unreadable data into a usable format; automate filtering, spectral analysis, and transform steps for multiple trucks and regions; and ultimately, apply machine learning techniques in real time to predict the ideal time to perform maintenance.*
>
> —Gulshan Singh, Baker Hughes

» *Read user story*

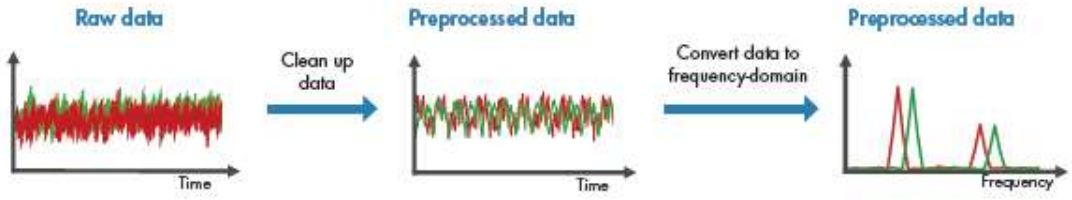


Synthetic fault data for a transmission model generated with Simulink.

Fuente: (Matlab, s.f.)

Preprocess Data

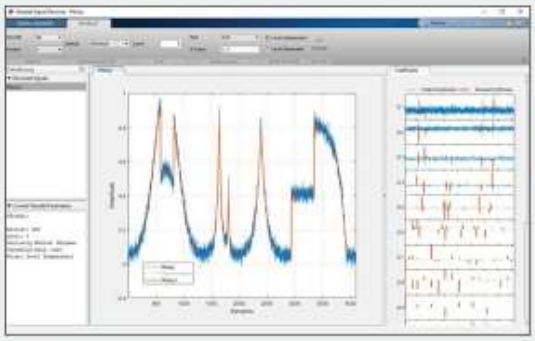
Once you have the data, the next step is to **preprocess the data** to convert it to a form from which condition indicators can be easily extracted. Preprocessing includes techniques such as **noise, outlier,** and **missing value removal**. Sometimes further preprocessing is necessary to reveal additional information that may not be apparent in the original form of the data. For example, this preprocessing may include conversion of time-domain data to **frequency-domain**.



Preprocess Data with MATLAB

Data is messy. With MATLAB, you can preprocess it, reduce its dimensionality, and engineer features:

- Align data that is sampled at different rates, and account for missing values and outliers.
- Remove noise, filter data, and analyze transient or changing signals using advanced signal processing techniques.
- Simplify datasets and reduce overfitting of predictive models using statistical and dynamic methods for feature extraction and selection.



Wavelet Signal Denoiser app for visualizing and denoising signals and comparing results.

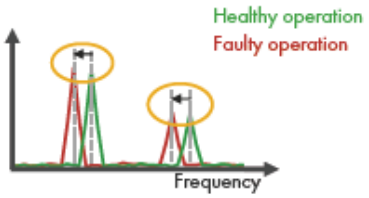
Fuente: (Matlab, s.f.)

Identify Condition Indicators

ACQUIRE DATA → PREPROCESS DATA → IDENTIFY CONDITION INDICATORS → TRAIN MODEL → DEPLOY AND INTEGRATE

The next step is to identify **condition indicators**, features whose behavior changes in a predictable way as the system degrades. These features are used to **discriminate between healthy and faulty operation**.

In the plot on the right, the peaks in the frequency data shift left as the pump degrades; therefore, the peak frequencies can serve as condition indicators.

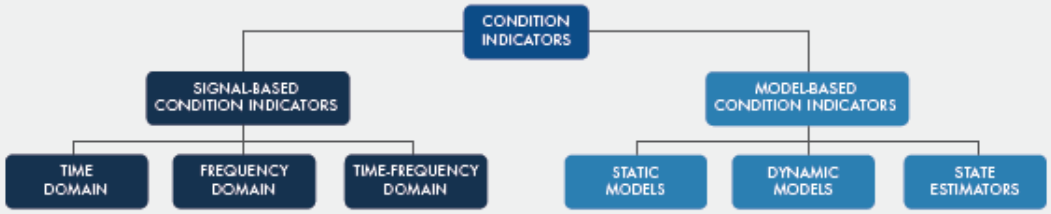


Peak frequencies can serve as **condition indicators.**

Identify Condition Indicators with MATLAB

MATLAB and Predictive Maintenance Toolbox™ let you design condition indicators using both signal-based and model-based approaches. You can calculate time-frequency moments that enable you to capture time-varying dynamics, which are frequently seen when analyzing vibration data. To detect sudden changes in data collected from machines displaying nonlinear behavior or characteristics, you can compute features based on phase-space reconstructions that track changes in your system's state over time.

Learn more: *What Is Predictive Maintenance Toolbox?* (2:00) - Video



Designing condition indicators using signal-based and model-based methods to monitor the health of your machinery.

Fuente: (Matlab, s.f.)

Train the Model

So far, you've extracted some features from your data that help you understand healthy and faulty operation of the pump. But at this stage, it's not clear what part needs repair or how much time there is until failure. In the next step, you can use the extracted features to **train machine learning models** to do several things.

Detect anomalies

You can track changes in your system to determine the presence of anomalies.

Learn more: *Anomaly Detection for Powertrain Production at Mercedes-Benz*

Detect different types of faults through classification

You can gain insight into what part of the pump requires attention.

Predict the transition from healthy state and failure

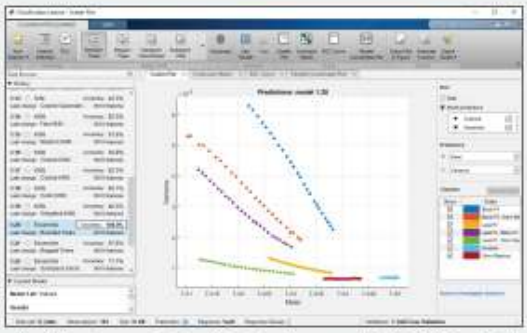
Finding a model that captures the relationship between the extracted features and the degradation path of the pump will help you estimate how much time there is until failure (**remaining useful life**) and when you should schedule maintenance.

Learn more: *Three Ways to Estimate Remaining Useful Life with MATLAB*

Train the Model Using Machine Learning with MATLAB

You can identify the root cause of failures and predict time-to-failure using classification, regression, and time-series modeling techniques in MATLAB:

- Interactively explore and select the most important variables for estimating RUL or classifying failure modes.
- Train, compare, and validate multiple predictive models with built-in functions.
- Calculate and visualize confidence intervals to quantify uncertainty in predictions.



Classification Learner app for trying different classifiers on your dataset. Find the best fit with common models like decision trees and support vector machines.

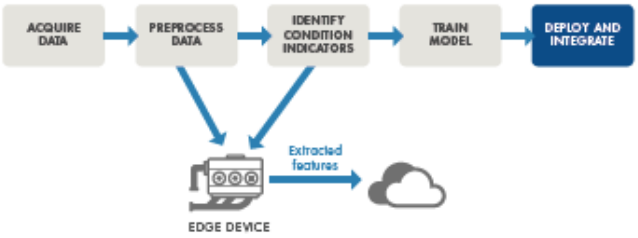
Fuente: (Matlab, s.f.)

Deploy and Integrate

After developing your algorithm, you can get it up and running by **deploying** it on the **cloud** or on your **edge device**. A cloud implementation can be useful when you are also gathering and storing large amounts of data on the cloud.

Alternatively, the algorithm can run on embedded devices that are closer to the actual equipment. This may be the case if an internet connection is not available.

A third option is to use a combination of the two. If you have a large amount of data, and if there are limits on how much data you can transmit, you can perform the preprocessing and feature extraction steps on your edge device and then send only the extracted features to your prediction model that runs on the cloud.



Deploy Algorithms in Production Systems with MATLAB

Data scientists often refer to the ability to share and explain results as *model interpretability*. A model that is easily interpretable has:

- A small number of features that typically are created from some physical understanding of the system
- A transparent decision-making process

Interpretability is important for applications when you need to:

- Prove that your model complies with government or industry standards
- Explain factors that contributed to a diagnosis
- Show the absence of bias in decision-making

We operate our machines nonstop, even on Christmas, and we rely on our MATLAB based monitoring and predictive maintenance software to run continuously and reliably in production.

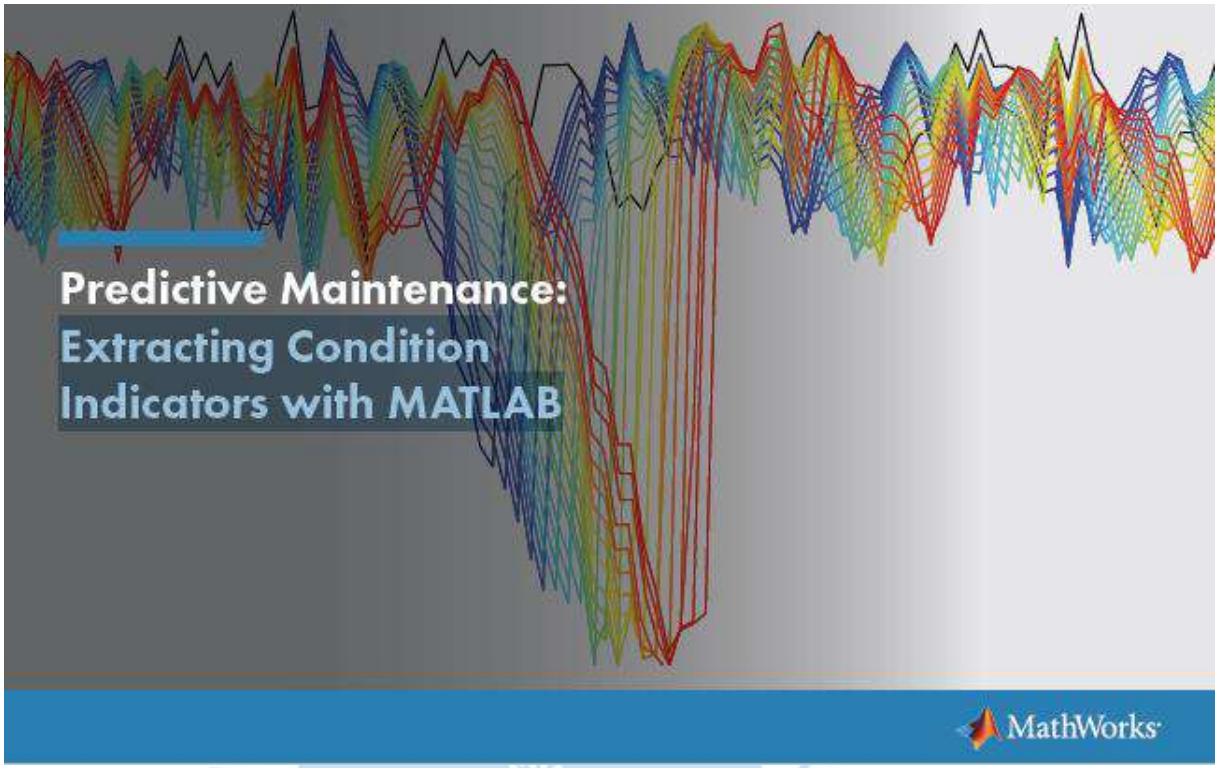
—Dr. Michael Kohlert, Mondi

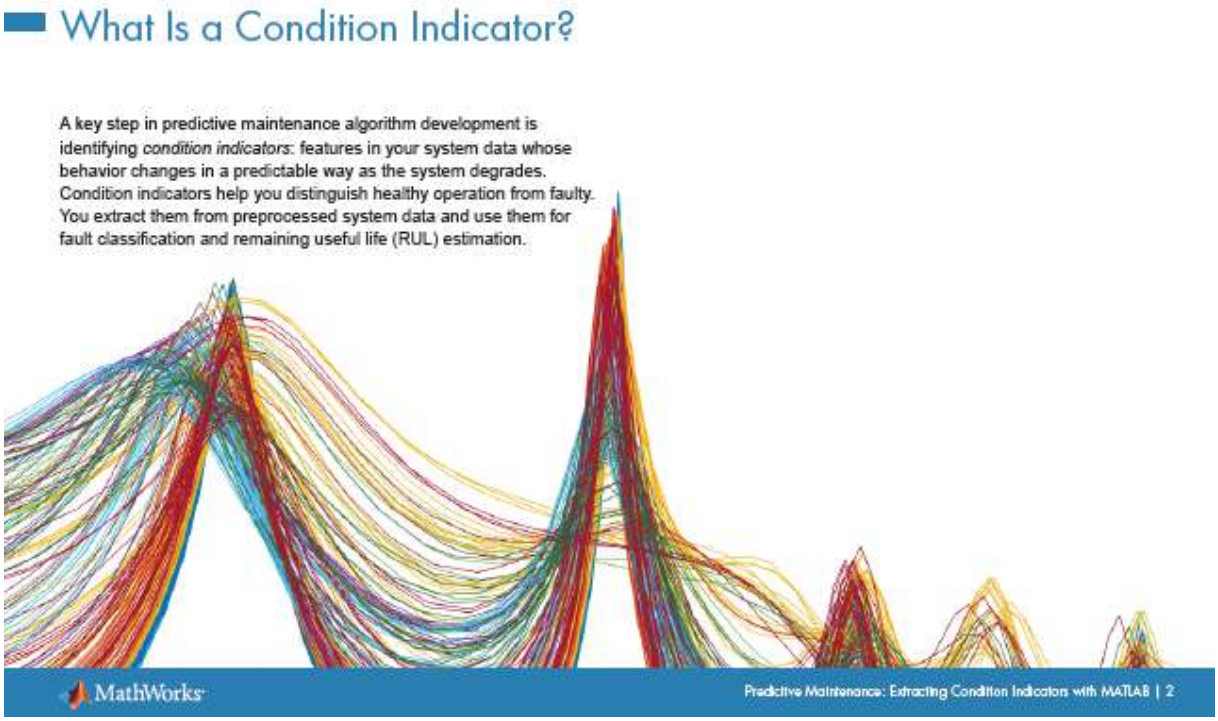
» *Read user story*

Share standalone MATLAB applications or run MATLAB analytics as a part of web, database, desktop, and enterprise applications without having to create custom infrastructure.

Fuente: (Matlab, s.f.)

Anexo C. Mantenimiento Predictivo: Condición de extracción. Indicadores con MATLAB.

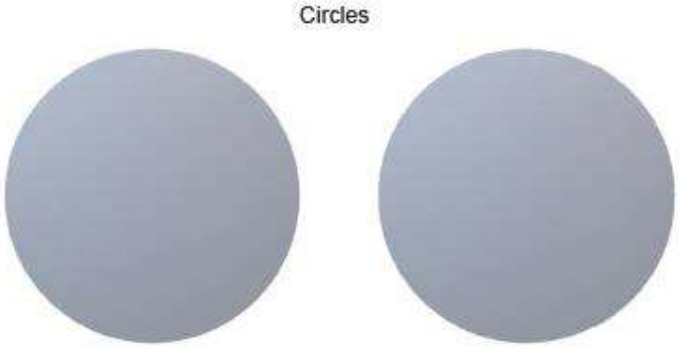




Fuente: (Matlab, s.f.)

A Visual Exercise

Let's start with a visual exercise to understand how condition indicators work. What's the difference between these two shapes?



It looks like there's no significant difference because the two circles look almost the same.

A Visual Exercise - Continued

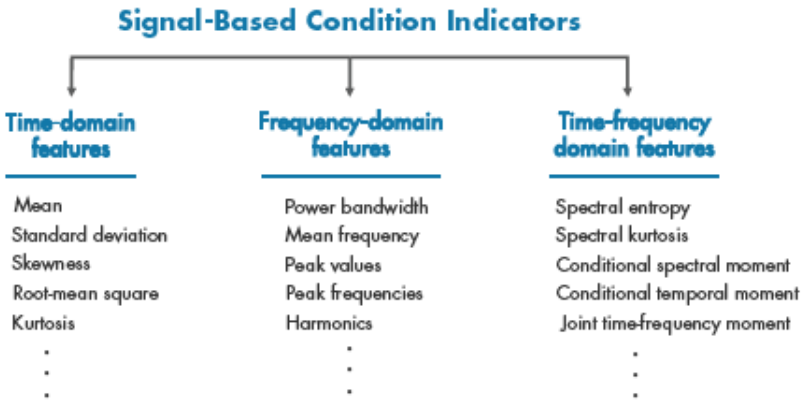
On the previous page, the shapes looked the same because you were looking at them from a certain angle, the top view. However, if you change your perspective, you can clearly see the differences between the two shapes and can identify them as a cone and a cylinder.

Similarly, when you look at raw measurement data from your machine, it's hard to tell healthy operation from faulty. But, using condition indicators, you're able to look at the data from a different perspective that helps you discriminate between healthy and faulty operation.

Fuente (Matlab, s.f.)

Feature Extraction Using Signal-Based Methods

You can derive condition indicators from data by using time, frequency, and time-frequency domain features:

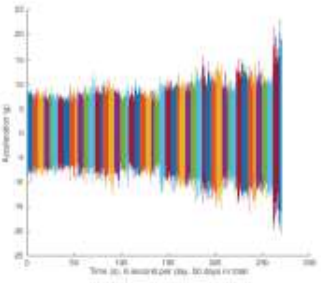


Feature Extraction Using Signal-Based Methods - Continued

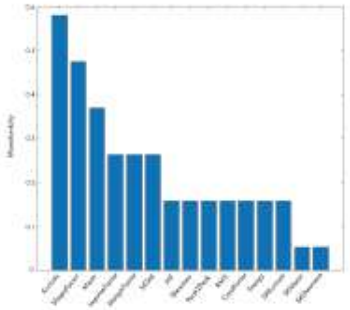
Time-Domain Features

For some systems, simple statistical features of time signals can serve as condition indicators, distinguishing faulty conditions from healthy. For example, the average value of a particular signal or its standard deviation might change as system health degrades. You can also use higher-order moments of the signal such as skewness and kurtosis. With such features, you can try to identify threshold values that distinguish healthy operation from faulty operation, or look for abrupt changes in the value that mark changes in system state.

Predictive Maintenance Toolbox™, an add-on product for MATLAB®, contains additional functions for computing time-domain features, demonstrated in the example *Turbine High-Speed Bearing Prognosis*.



The changing trend in time-domain vibration signals shows the degradation of a wind turbine high-speed shaft over 60 consecutive days.



The monotonicity graph shows the feature importance ranking for different time-domain and frequency-domain features.

Fuente: (Matlab, s.f.)

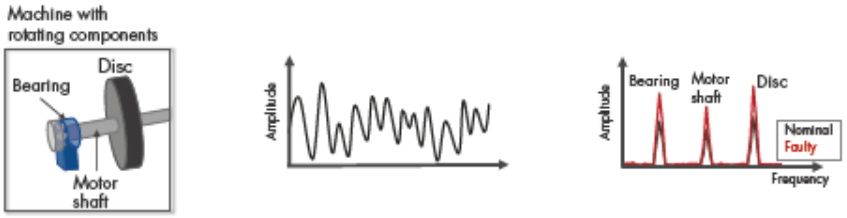
Feature Extraction Using Signal-Based Methods - Continued

Frequency-Domain Features

Sometimes, time-domain features alone are not sufficient to act as condition indicators, so you'll want to look at frequency-domain features as well.

Take a machine with rotational components and three vibration sources: bearing, motor shaft, and disc. If you look at the vibration data from the machine in the time domain, you see the combined effect of all the vibrations from these different rotating components. But by analyzing the data in the frequency domain, you can isolate different sources of vibration, as seen in the second plot. The peak amplitudes, and how much they change from nominal values, can indicate the severity of the faults.

More information on calculating frequency-domain condition indicators can be found in the example *Condition Monitoring and Prognostics Using Vibration Signals*.



When analyzing the vibration data from the machine in the time domain, the bearing, motor shaft, and disc all affect vibration amplitude. Using frequency-domain analysis, you can distinguish between different sources of vibration.

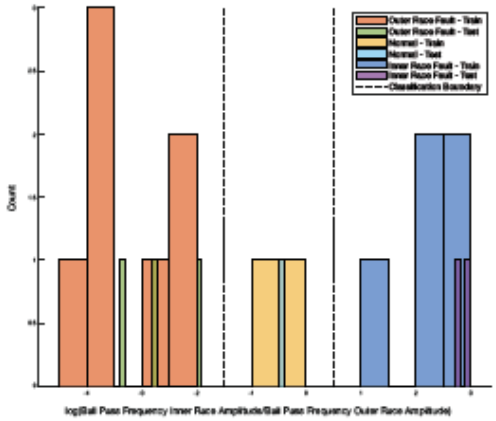
Feature Extraction Using Signal-Based Methods - Continued

Time-Frequency Domain Features

Another way to extract features is to perform time-frequency spectral analysis on the data, which helps characterize changes in the spectral content of a signal over time. Time-frequency domain condition indicators include features such as spectral kurtosis and spectral entropy.

The *Rolling Element Bearing Fault Diagnosis* example shows how to use kurtogram, spectral kurtosis, and envelope spectrum to identify different types of faults in rolling element bearings.

» *Learn more about time, frequency, and time-frequency domain features*



The histogram shows a clear separation between the three bearing conditions. The log ratio between the bandpass frequency inner and outer race amplitudes is a valid feature to classify bearing faults.

Fuente: (Matlab, s.f.)

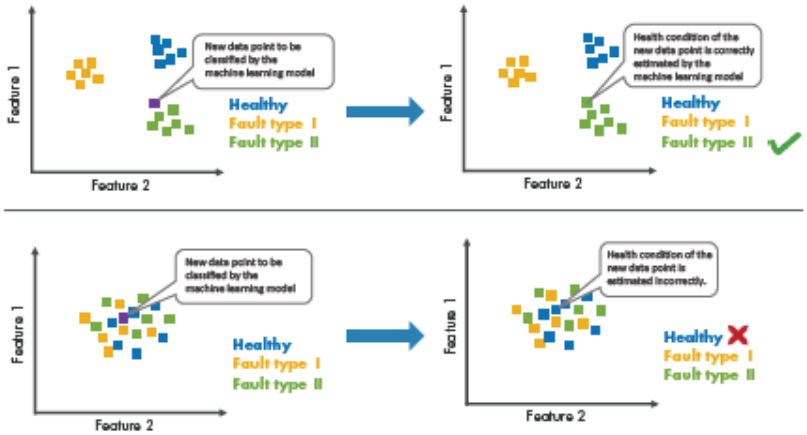
Predictive Maintenance Workflow

Now that you've looked at what condition indicators are, it's time to take a look at the steps involved in extracting the features for identifying condition indicators. Designing a predictive maintenance algorithm starts with collecting data from your machine under different operating conditions and fault states. The raw data is then preprocessed for cleaning it up and bringing it into a form from which condition indicators can be extracted. Feature extraction looks for features that are distinctive, meaning they uniquely define healthy operation and different fault types. These features will be your *condition indicators*.

Using the extracted features, you can train a machine learning model for fault classification and remaining useful life estimation. You can then deploy your algorithm and integrate it into your systems for machine monitoring and maintenance.

What Are Distinctive Features and Why Are They Important?

Once you identify some useful features, you can use them to train a machine learning model. If the selected set of features is distinctive, the model can correctly estimate the machine's current condition when you feed new data from the machine to the model.

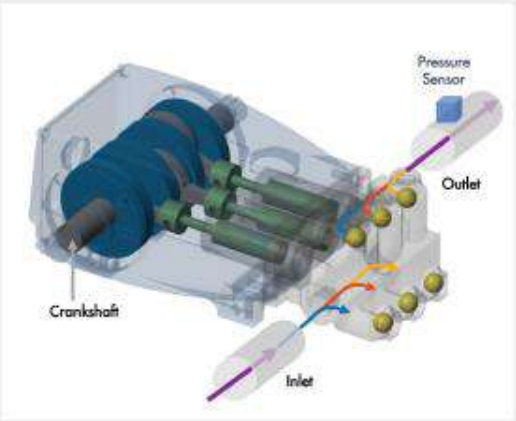


Fuente: (Matlab, s.f.)

Example: The Triplex Pump

This example uses a triplex pump to demonstrate the workflow. The pump has a motor that turns the crankshaft that in turn drives three plungers. The fluid gets sucked into the inlet and discharged through the outlet, where the pressure is measured by a sensor. Faults that can develop in such a pump include:

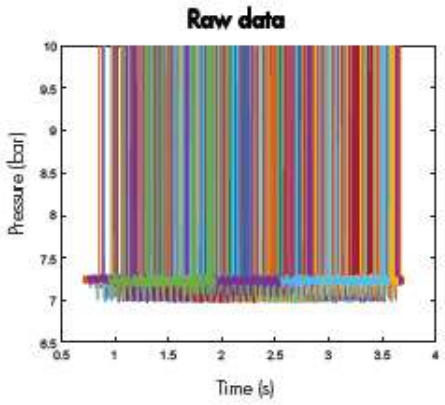
- Seal leakage
- Blocked inlet
- Worn bearing



Acquiring Data

This pressure data includes one-second-long measurements taken at steady state from normal operation, all three fault types, and also their combinations:

- Healthy
- Blocked inlet
- Worn bearing
- Seal leakage
- Blocked inlet, worn bearing
- Seal leakage, worn bearing
- Seal leakage, blocked inlet

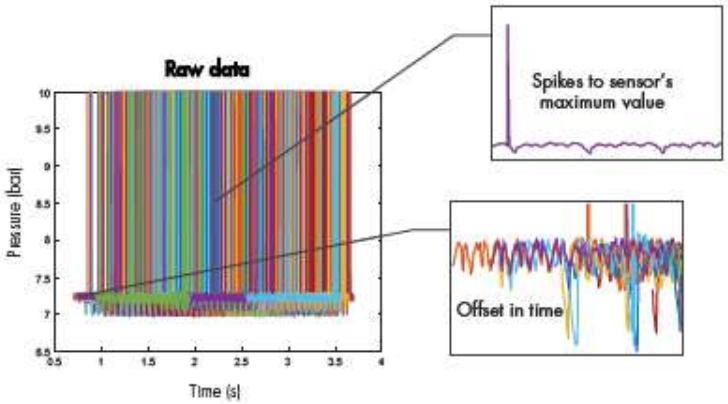


Plot of the pressure data that has been collected from the triplex pump.

Fuente: (Matlab, s.f.)

Preprocessing Data

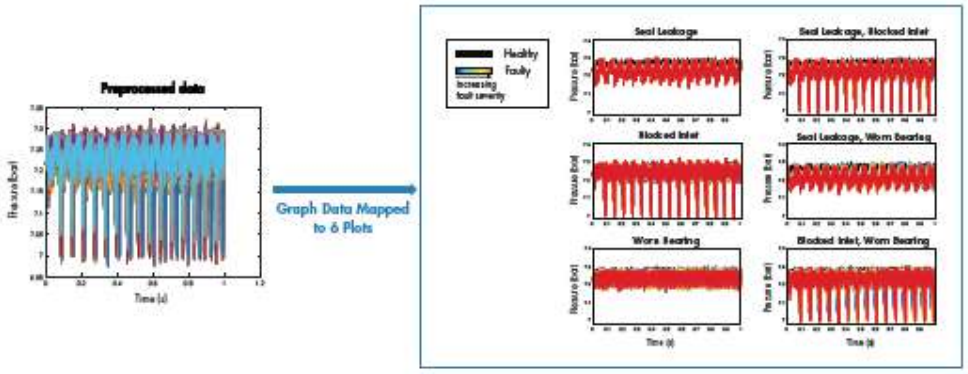
You need to bring the data into a usable form to extract condition indicators. The raw data is noisy and has spikes up to the sensor's maximum value. It's also offset in time even though the durations of the measurements are the same.



Preprocessing Data - Continued

MATLAB has *functions* to help smooth, denoise, and perform other preprocessing techniques on signal data.

The processed data includes all the healthy and faulty conditions. In order to investigate different fault types and their combinations, you can plot them individually. The first thing to notice on these plots is the cyclical behavior of the time-domain pressure signal. Next, take a look at a plot of a shorter time period on the next page to see what's happening in each cycle more clearly.

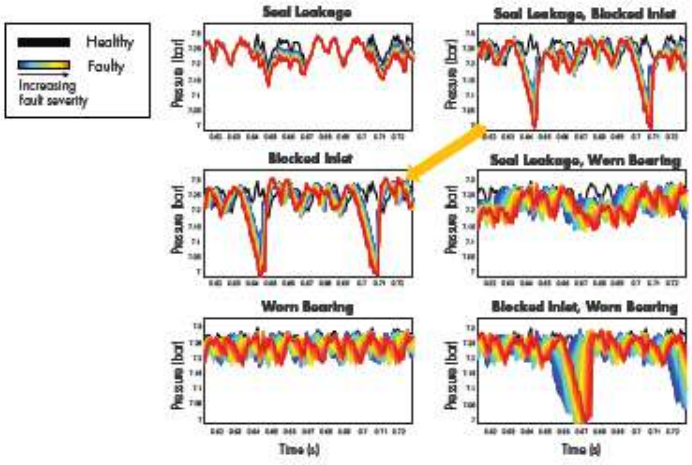


Fuente: (Matlab, s.f.)

Preprocessing Data - Continued

These plots provide a more detailed look at the pressure signal for different types of faults. The change in the pressure data as the pump deteriorates is reflected with changing colors from dark blue to red indicating increasing fault severity. Now the question is: Can you distinguish the black line, the healthy operation, from the rest of the data on each plot? And, can you identify the unique differences between each set of colored lines? Notice how the pressure data looks very similar for the "seal leakage, blocked inlet" and "blocked inlet" faults.

Now let's look at some of the time-domain features to identify condition indicators to help you distinguish between fault types.

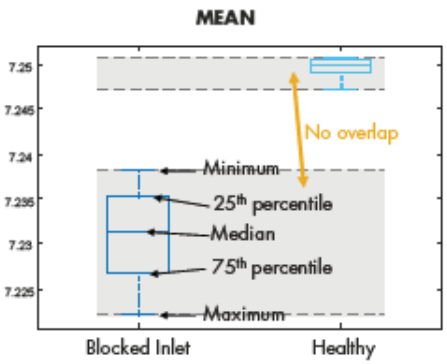


Using Time-Domain Features for Identifying Condition Indicators

Use trial and error to see how each of the following set of common time-domain features performs: *Mean*, *Variance*, *Skewness*, and *Kurtosis*.

One way to understand if these condition indicators can differentiate between types of faults is to investigate them using a boxplot. First, plot a single feature, such as the mean, for the healthy condition and blocked inlet fault.

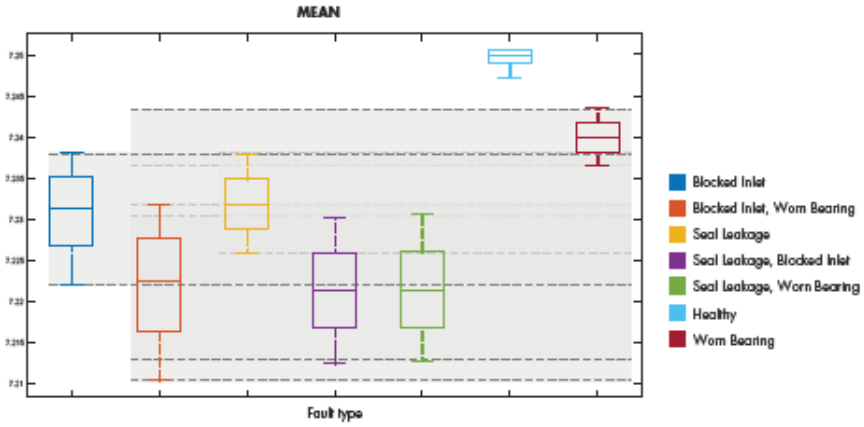
The boxes don't overlap in the plot. This means there's a difference between these data groups. By using the mean of the pressure data, you can easily distinguish the blocked inlet fault from healthy condition.



Fuente: (Matlab, s.f.)

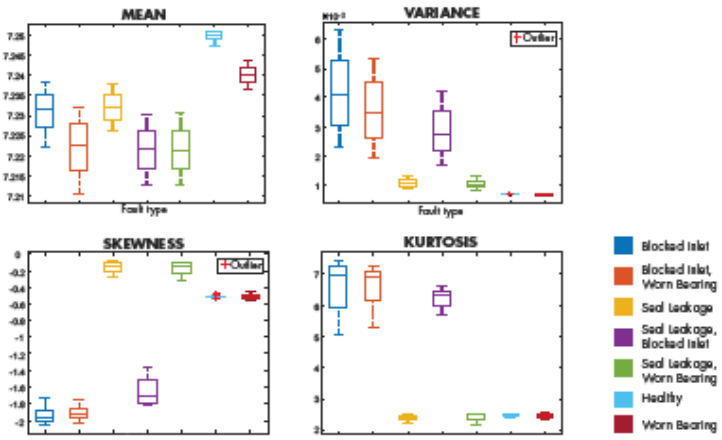
Using Time-Domain Features for Identifying Condition Indicators - Continued

Things change when you add the datasets for other fault types as well. You're not able to distinguish between all the fault types as some of them overlap. Due to this overlapping, the mean on its own is not enough to set fault types apart.



Using Time-Domain Features for Identifying Condition Indicators - Continued

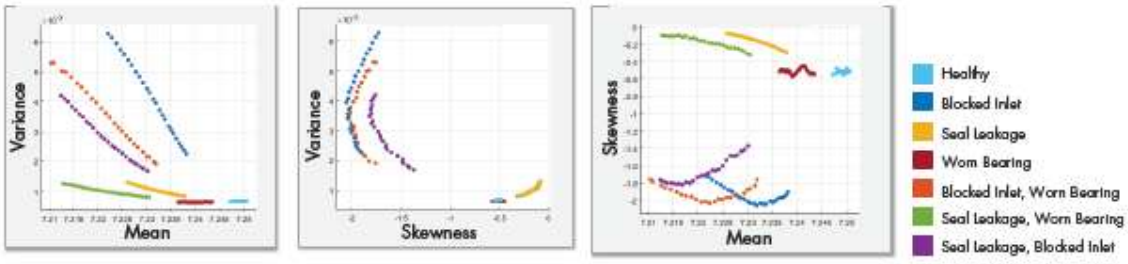
If you try this with other features as well, you end up at the same conclusion: A single condition indicator is not sufficient to classify the faulty behavior, especially when you have multiple faults.



Fuente: (Matlab, s.f.)

Using Time-Domain Features for Identifying Condition Indicators - Continued

Below is a scatter plot of a combination of the features: mean, variance, and skewness. Notice how well the variance versus mean plot distinguishes between different types of faults.



You can immediately see that two condition indicators are better than one for separating different faults. You can try different pairs of features to see which ones are better at classifying faults.

Using Time-Domain Features for Identifying Condition Indicators - Continued

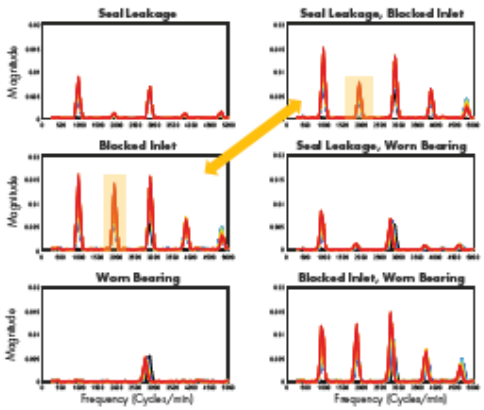
Frequency-domain analysis is important in analyzing periodic data and data acquired from a machine with rotating components; let's see if you can extract additional features by analyzing your data in the frequency domain.

What differentiates these plots from each other are the peaks and the peak frequencies, so they can serve as condition indicators. With time-domain features, it was hard to distinguish between the "seal leakage, blocked inlet" and "blocked inlet" faults because of the similarity of the datasets. By looking at the data in the frequency domain, you can see that the peak values at the highlighted frequency range will help you successfully separate these two faults.

In MATLAB, you can use `fft` function to compute the fast Fourier transform of a signal and analyze it in the frequency domain. You can then use the `findpeaks` function to extract the peaks and peak frequencies from the FFT signal.

In summary, the features you should use to train a machine learning model in this example include:

- **Time-domain features**: Mean, variance, skewness, kurtosis
- **Frequency-domain features**: Peaks and peak frequencies

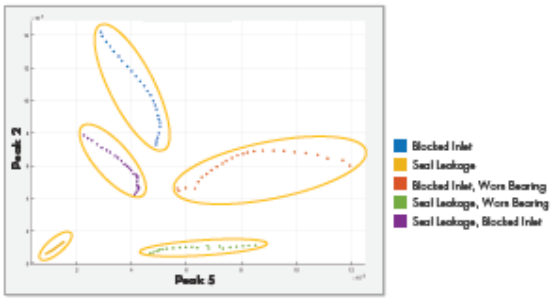


Fuente: (Matlab, s.f.)

Using Time-Domain Features for Identifying Condition Indicators - Continued

After selecting the frequency-domain features, try to perform an analysis like you did with the time-domain features. The below plot shows the second and fifth peaks with respect to each other.

These features effectively separate different groups, which are highlighted with yellow circles. This means that the selected features are distinctive and good candidates for training a machine learning model.



Note that when you're investigating these features, not only are you looking for different clusters, but you also want them to be further away from each other. This makes it easier for the trained models to identify new data points.

The previous page shows plots of the peaks for each of the faults, with the faults in color and the healthy condition in black. Notice that the "worn bearing" plot contains only Peak 3 for the worn bearing and healthy conditions. This is why these conditions don't show up on a plot representing Peaks 2 and 5. This is another reason why we need multiple features to effectively separate the different groups.

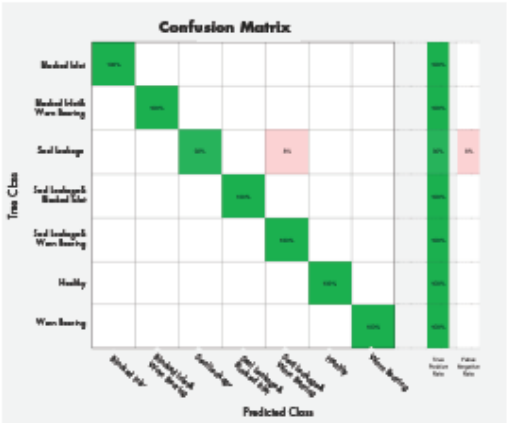
Training the Model

After extracting the condition indicators, you can train a *machine learning model* with the extracted features and check the accuracy of the trained model with a confusion matrix. The confusion matrix at right shows the results of one the best-performing classifiers that has been trained with the extracted features. The *Classification Learner app* in MATLAB helps you find the best classifier for your dataset quickly.

The plot shows the true positive rates in green and false negative rates in red.

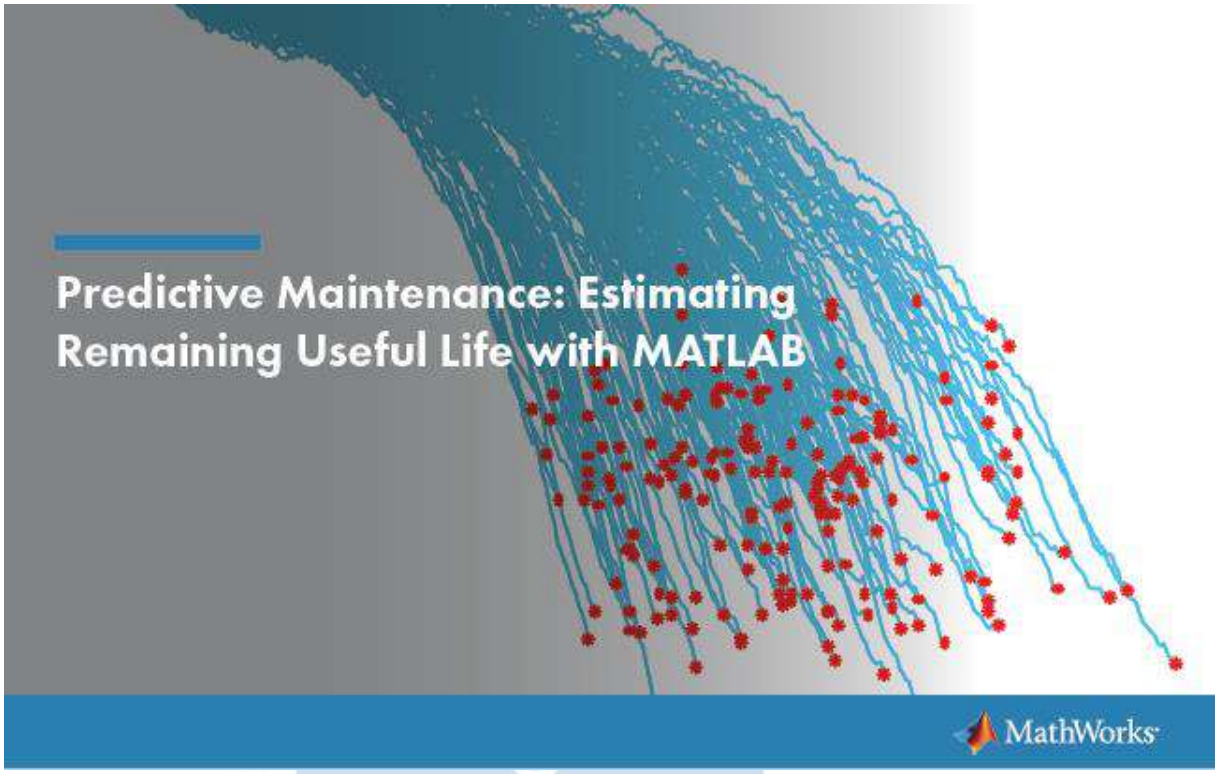
If you're satisfied with the accuracy of your machine learning model, you can continue with deploying your predictive maintenance algorithm and integrating it into your system. Otherwise, you should revisit the feature extraction step of the predictive maintenance workflow and try training machine learning models with different sets of features, as highlighted with the arrows in the workflow chart.

You may be wondering, how many features are enough to train a machine learning model? Unfortunately, there's no magic number. Just remember that machine learning models can benefit from a high-dimensional set of features that are distinctive and can effectively differentiate fault types.



Fuente: (Matlab, s.f.)

Anexo D. Mantenimiento predictivo: estimación de vida útil restante con MATLAB

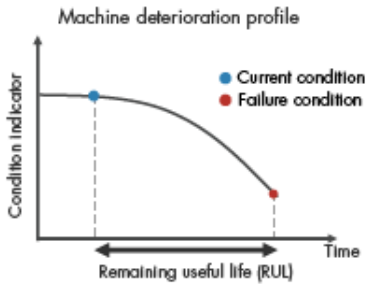


What Is Remaining Useful Life?

One of the goals of predictive maintenance is to estimate the remaining useful life (RUL) of a system. RUL is the time between a system's current condition and failure. Depending on your system, time can be represented in terms of days, flights, cycles, or any other quantity.

On the plot, we see the deterioration path of a machine over time.

This ebook explores three common models used to estimate RUL (similarity, survival, and degradation) and then walks through the RUL workflow with an example using a similarity model.



Fuente: (Matlab, s.f.)

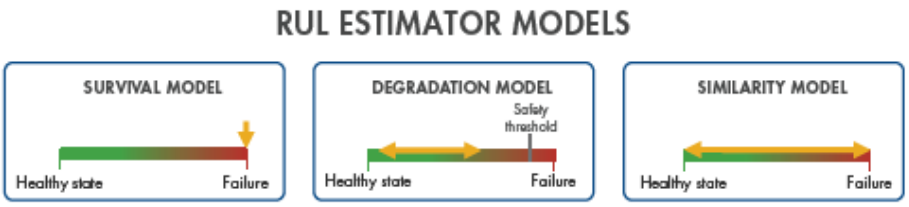
Three Common Ways to Estimate RUL

There are three common models used to estimate RUL: similarity, survival, and degradation. Which model should you use? It depends on how much information you have.

Use a *survival model* if you have data only from time of failure rather than complete run-to-failure histories.

Use a *degradation model* if you have some data between a heathy state and failure, and you know a safety threshold that shouldn't be exceeded.

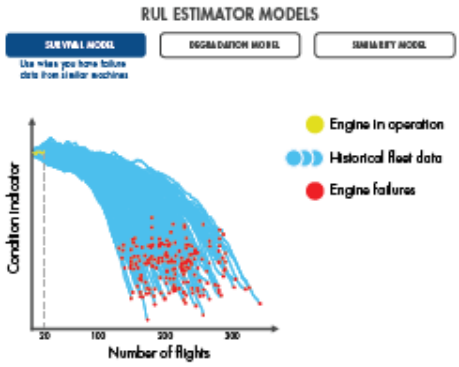
Use a *similarity model* if your data spans the full degradation of a system from a healthy state to failure.



Find more detailed information on *RUL Estimator Models* used in predictive maintenance.

The following aircraft engine example illustrates how estimator models work.

The Working Principle of RUL Estimator Models: The Survival Model

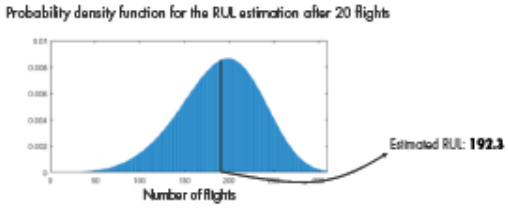


In this example, you want to figure out how many flights the engine can operate until its parts need repair or replacement.

The yellow line on the plot represents your engine, which has been in operation for 20 flights. The blue lines represent the data from a fleet with the same type of engine. The red marks indicate when the engines failed.

If you don't have the complete histories from the fleet, but you do have the failure data, then you can use survival models to estimate RUL.

You can determine how many engines failed after a certain number of cycles (number of flights), and you also know how many flights the engine has been in operation. The survival model uses a probability distribution of this data to estimate the remaining useful life.



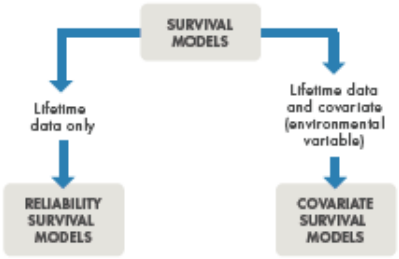
Fuente: (Matlab, s.f.)

The Working Principle of RUL Estimator Models: The Survival Model in MATLAB

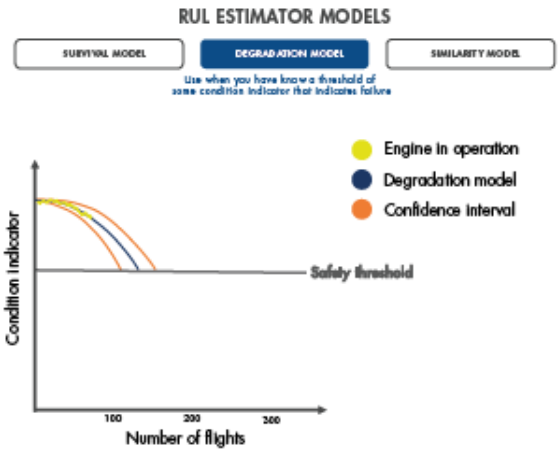
Survival analysis is a statistical method used to model time-to-event data. It is useful when you do not have complete run-to-failure histories, but instead have:

Only data about the life span of similar components. For example, you might know how many miles each engine in your ensemble ran before needing maintenance. In this case, you use the MATLAB® model `reliabilitySurvivalModel`. Given the historical information on failure times of a fleet of similar components, this model estimates the probability distribution of the failure times. The distribution is used to estimate the RUL of the test component.

Both life spans and some other variable data (covariates) that correlates with the RUL. Covariates include information such as the component provider, regimes in which the component was used, or manufacturing batch. In this case, use the MATLAB model `covariateSurvivalModel`. This model is a proportional hazard survival model that uses the life spans and covariates to compute the survival probability of a test component.



The Working Principle of RUL Estimator Models: The Degradation Model



In some cases, no failure data is available from similar machines. But you may have knowledge about a safety threshold that shouldn't be crossed as this may cause failure. You can use this information to fit a degradation model to the condition indicator, which uses the past information from our engine to predict how the condition indicator will change in the future. This way, you can statistically estimate how many cycles there are until the condition indicator crosses the threshold, which helps you estimate the remaining useful life.

Fuente: (Matlab, s.f.)

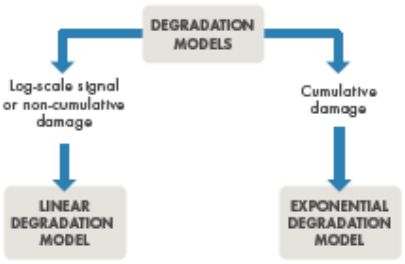
The Working Principle of RUL Estimator Models: The Degradation Model in MATLAB

Degradation models estimate RUL by predicting when the condition indicator will cross a prescribed threshold. These models are most useful when there is a known value of your condition indicator that indicates failure. The two available degradation model types in Predictive Maintenance Toolbox™ are:

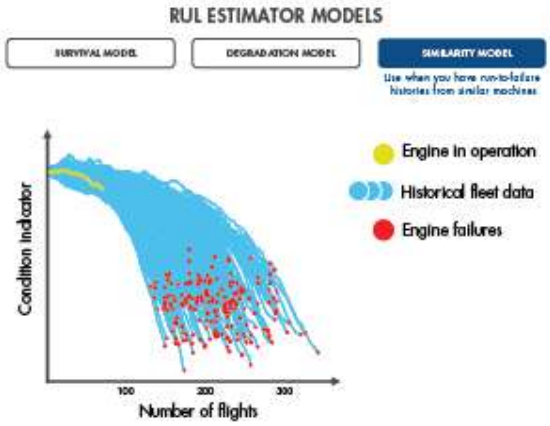
Linear degradation model (`linearDegradationModel`) describes the degradation behavior as a linear stochastic process with an offset term. Linear degradation models are useful when your system does not experience cumulative degradation.

Exponential degradation model (`exponentialDegradationModel`) describes the degradation behavior as an exponential stochastic process with an offset term. Exponential degradation models are useful when the test component experiences cumulative degradation.

Degradation models work with a single condition indicator. However, you can use principal-component analysis or other fusion techniques to generate a fused condition indicator that incorporates information from more than one condition indicator.



The Working Principle of RUL Estimator Models: The Similarity Model



Similarity models are useful when you have run-to-failure data (the complete histories from a fleet with the same type of engine, from healthy state through to degradation and failure).

Fuente: (Matlab, s.f.)

The Working Principle of RUL Estimator Models: The Similarity Model in MATLAB

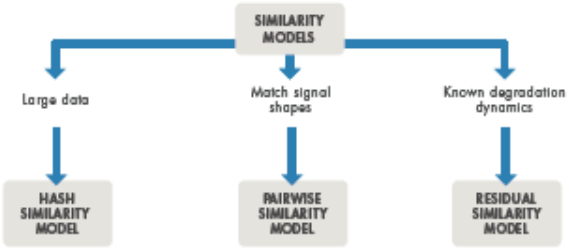
Predictive Maintenance Toolbox includes three types of similarity models:

Hashed-feature similarity model (`hashSimilarityModel`) transforms historical degradation data from each member of your ensemble into fixed-size, condensed information such as the mean, maximum, or minimum values, etc.

Pairwise similarity model (`pairwiseSimilarityModel`) finds the components whose historical degradation paths are most correlated to those of the test component.

Residual similarity model (`residualSimilarityModel`) fits prior data to a model such as an ARMA model or a model that is linear or exponential in usage time. It then computes the residuals between the data predicted from the ensemble models and the data from the test component. See *Similarity-Based Remaining Useful Life Estimation* for more information.

The following section discusses the similarity model in more detail to illustrate how an RUL prediction is performed.

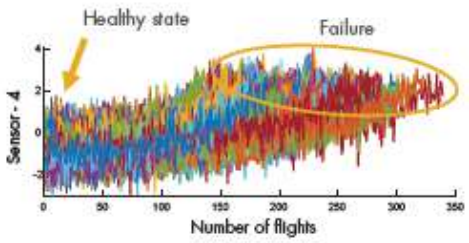


RUL Estimation Workflow Using the Similarity Model

Acquiring Data

The similarity model is a useful RUL estimation technique. See how this model is used in an example to better understand how an RUL prediction is performed.

The first step when developing a predictive maintenance model is to acquire data. This example uses the Prognostics and Health Management challenge dataset publicly available on *NASA's data repository*. This data set includes run-to-failure data from 218 engines, where each engine dataset contains measurements from 21 sensors. Measurements such as fuel flow, temperature, and pressure are gathered through sensors placed in various locations in the engine to provide measurements to the control system and monitor the engine's health. The plot shows what one sensor's measurements look like for all 218 engines.



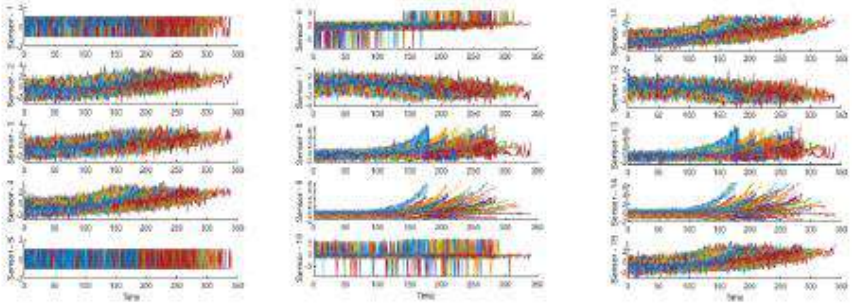
On the plot, the x-axis shows the number of cycles (flights), and the y-values represent the averaged sensor values at each flight. Each engine starts in a healthy state and ends in failure.

Fuente: (Matlab, s.f.)

RUL Estimation Workflow Using the Similarity Model *continued*

Preprocessing Data and Identifying Condition Indicators

The previous page showed data from Sensor 4, but the full dataset also includes data from 20 other sensors. If you take a closer look at some of the other sensor readings, you can see that some of these measurements don't show a significant trend toward failure (such as Sensors 1, 5, 6, and 10). Therefore, they won't contribute to the selection of useful features for training a similarity model.



RUL Estimation Workflow Using the Similarity Model *continued*

Preprocessing Data and Identifying Condition Indicators

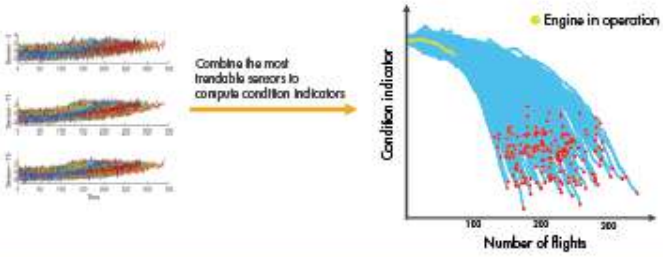
Instead of using all the sensor measurements, identify the three most trendable datasets (the ones that show a significant change in their profile between the healthy state and failure). Sensors 2, 11, and 15 are good candidates to use together to create degradation profiles.

Degradation profiles represent the evolution of one or more condition indicators for each machine in the ensemble (each component), as the machine transitions from a healthy state to a faulty state.

In the preprocessing step, data reduction is performed by selecting only the most trendable sensors (Sensor 2, 11, and 15) and combining them to compute condition indicators.

For more information on how to select and employ condition indicators, see *Predictive Maintenance: Extracting Condition Indicators with MATLAB*.

To estimate the remaining useful life for the current engine (shown in yellow), you would use Sensors 2, 11, and 15 to compute condition indicators that represent the degradation profiles of the fleet. In the graph on the right, you can see that the engine is currently at 60 flights, and the red dots mark where similar engines in the fleet have failed.



Fuente: (Matlab, s.f.)

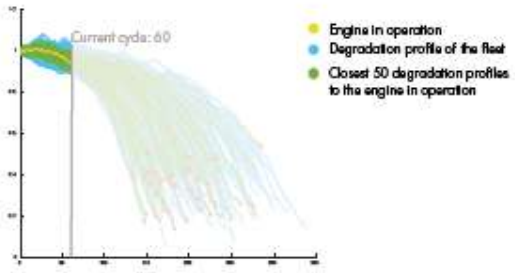
RUL Estimation Workflow Using the Similarity Model *continued*

Training a Similarity Model

You'll want to split the data into two groups, using a larger portion of it to train a similarity model and the rest to test the trained model. You use the known RUL to evaluate the trained model's accuracy.

The similarity model works by finding the closest engine profiles to your engine up to the current cycle. You have the failure times of the closest engines from the historical fleet data, which gives you an idea of the expected failure time of your current engine. You can use this data to fit a probability distribution as seen in the bottom plot.

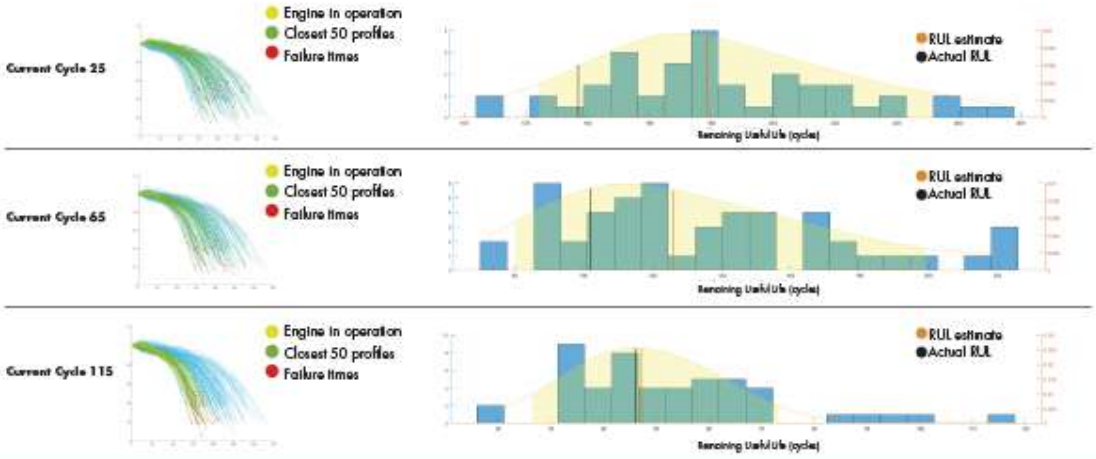
The median of this distribution gives you the remaining useful life estimate of your engine, which you can compare to the actual RUL to measure accuracy.



RUL Estimation Workflow Using the Similarity Model *continued*

Training a Similarity Model

At each iteration of the RUL computation, the similarity model finds the closest engine paths, which are shown in green, and computes the RUL using a probability distribution plot (shown on the right). The plots below show the engine profiles and their probability distribution for three cycles: 25, 65, and 115.



Fuente: (Matlab, s.f.)

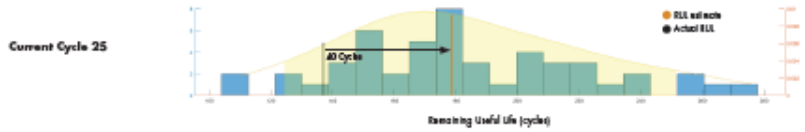
RUL Estimation Workflow Using the Similarity Model *continued*

Training a Similarity Model

On the probability distribution plots, the orange lines represent the predicted RULs and the black lines show the actual RUL.

Notice that the predicted RUL gets closer to the actual RUL as time passes and the model has more data.

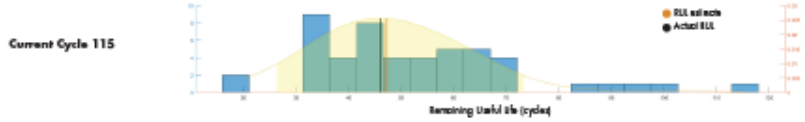
When the engine is at 25 cycles (or flights), the model doesn't have much data to work with yet, so the prediction is 40 cycles off from the true value with a very wide distribution.



As the model gets new data from the engine, the similarity model trains on a larger set of data, as seen at cycle 65 and 115. As a result, the prediction accuracy improves over time.

With more data the RUL predictions become more accurate and the distribution more concentrated.

Using the RUL estimation from cycle 115, you would have a reasonably accurate expectation of your engine's RUL and be able to schedule maintenance at the best time.



Learn More

Ready for a deeper dive? Explore these resources to learn more about predictive maintenance workflow, examples, and tools.

Watch

What Is Predictive Maintenance Toolbox? (2:06) - Video
Predictive Maintenance Tech Talks - Video Series
Predictive Maintenance in MATLAB and Simulink (35:54) - Video
Feature Extraction Using Diagnostic Feature Designer App (4:45) - Video

Read

Overcoming Four Common Obstacles to Predictive Maintenance - White Paper
MATLAB and Simulink for Predictive Maintenance - Overview
MATLAB Predictive Maintenance Examples - Code Examples

Fuente: (Matlab, s.f.)

Anexo E. Big Data y Machine Learning para mantenimiento predictivo. MATLAB EXPO 2017

MathWorks

Agenda

- The Predictive Maintenance Opportunity
- Exploring Big Data
- Machine Learning Approaches
- Deep Learning
- Fault Modelling
- Deploying to the Edge and the Cloud

MATLAB EXPO 2017 2

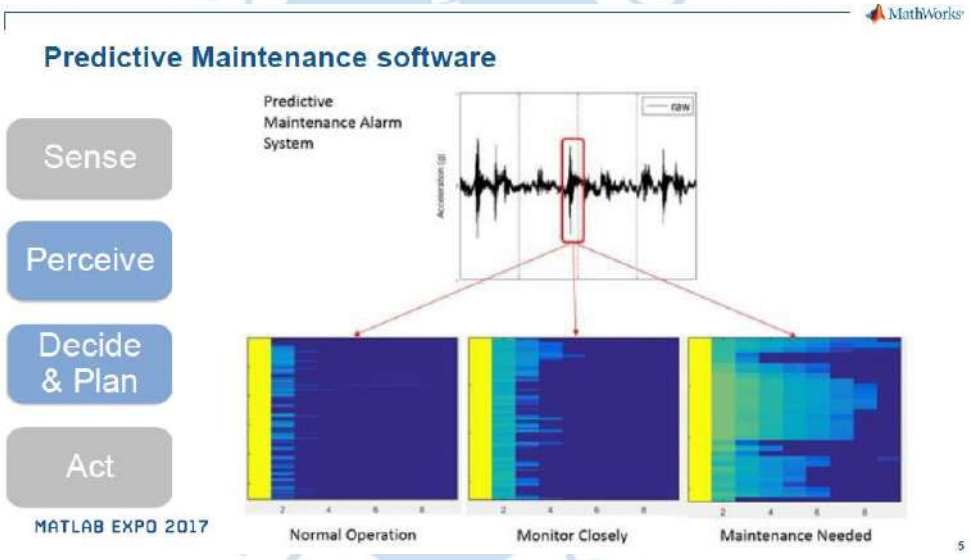
MathWorks

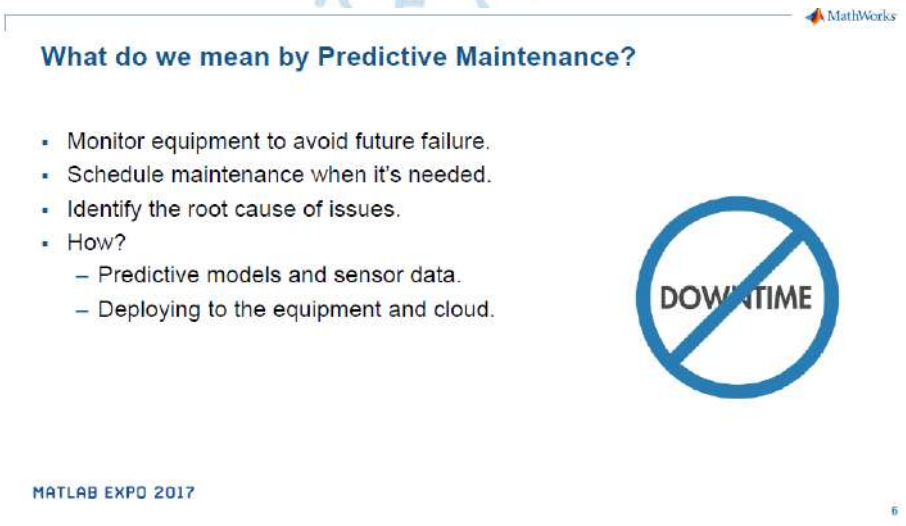
React or Prevent?

Aaron "tango" Tang on Flickr

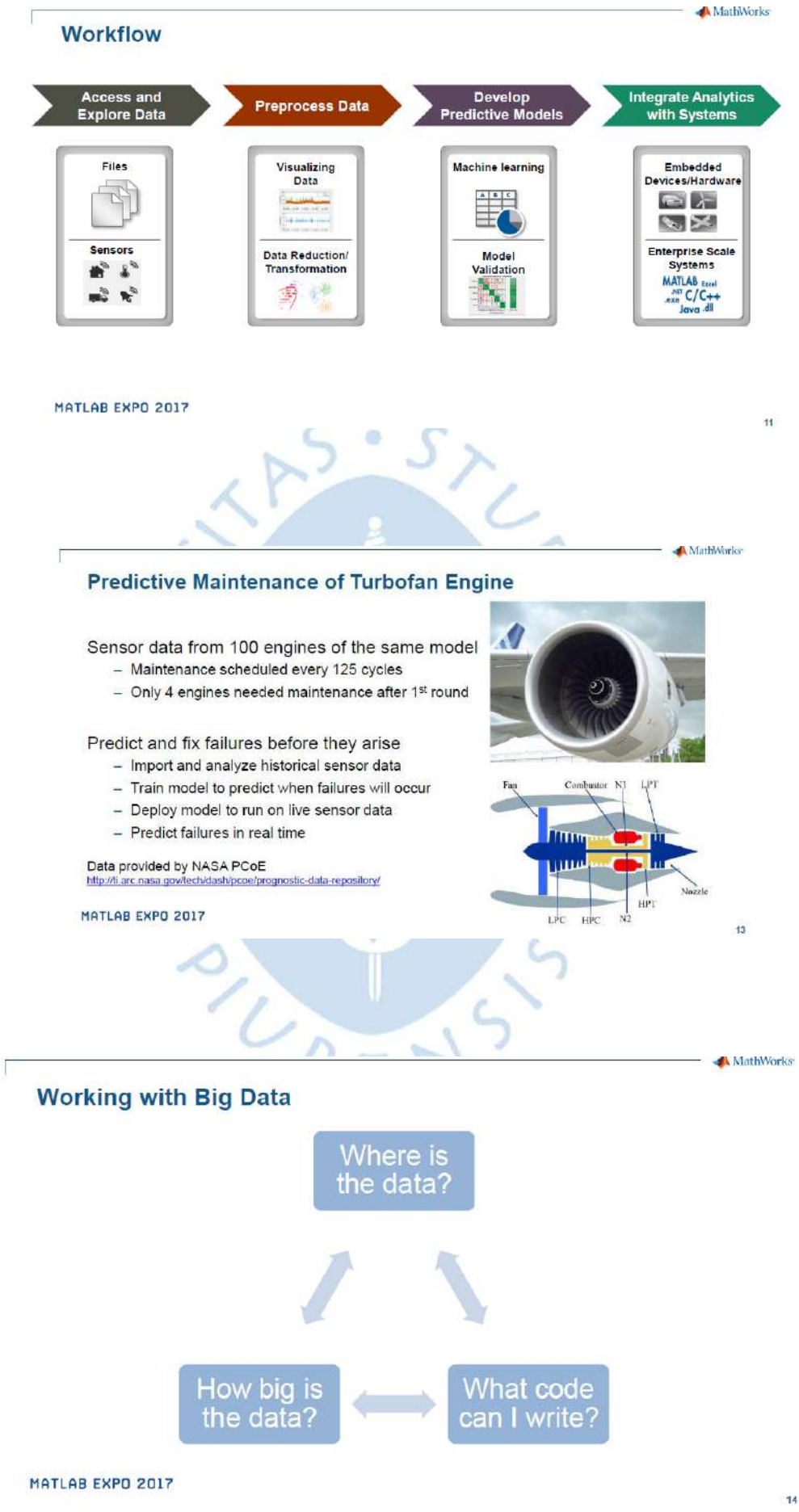
MATLAB EXPO 2017 3

Fuente: (Matlab, s.f.)





Fuente: (Matlab, s.f.)



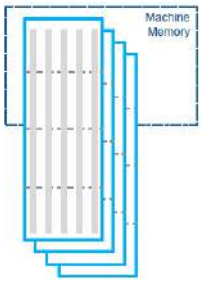
Fuente: (Matlab, s.f.)

MathWorks

Tall Arrays

Scaling your code to big data

- Automatically optimize data access bottlenecks
  - Write code the same way you've always written it
  - MATLAB automatically reorders operations to minimize disk access
- Applicable when:
  - Data is **columnar** – with **many** rows
  - Overall data size is **too big to fit into memory**
  - Operations are mathematical/statistical in nature
- Statistical and machine learning applications
  - Hundreds of functions supported in MATLAB and Statistics and Machine Learning Toolbox



Tall Data

MATLAB EXPO 2017

15

MathWorks

Filtering Data

```
% Point to where the data lives. Could be large text files, large collections
% of small files, or pageable databases.
ds = datastore('.\Data\*.csv');
% Inform MATLAB that we will treat this data as a tall array.
% We could a tall array from a local variable for prototyping.
engineData = tall(ds);
% Assume maintenance is being done regardless of condition after 125 cycles
engineData = engineData(engineData.Time <= 125,:)
```

```
engineData =

  M×16 tall table

    Unit    Time    LPCOutletTemp    HPCOutletTemp    LPTOutletTemp    TotalHPCOutletPres
    ____    ____    _____________    _____________    _____________    __________________

    1.00     5.00       642.21           1587.08          1403.21           554.16
    1.00     6.00       642.26           1585.98          1402.76           554.23
    1.00     7.00       642.33           1586.08          1401.69           554.34
    1.00     8.00       642.37           1585.08          1401.84           554.26
    1.00     9.00       642.33           1586.72          1399.63           554.11
    1.00    10.00       642.19           1588.39          1398.47           554.03
    1.00    11.00       642.22           1587.85          1398.93           554.00
    1.00    12.00       642.15           1586.07          1399.40           554.04
     :        :           :                :                :                 :
     :        :           :                :                :                 :
```

MATLAB EXPO 2017

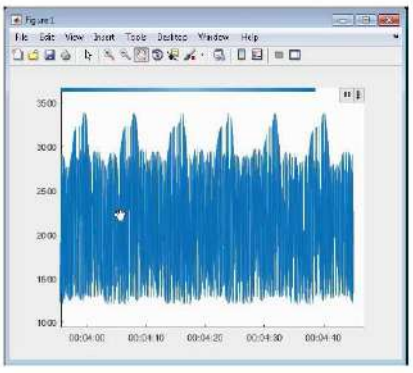
16

MathWorks

Visualizing Big Data Using `tall`

- Support for:
  - `histogram`
  - `histogram2`
  - `ksdensity`
  - `plot`
  - `scatter`
  - `binscatter`

R2017b



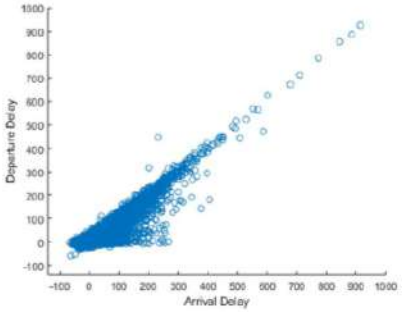
MATLAB EXPO 2017

18

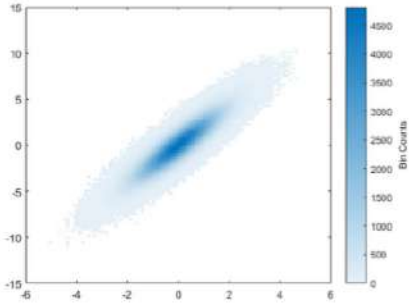
Fuente: (Matlab, s.f.)

Visualizing Big Data Using `tall`

scatter



binscatter



MATLAB EXPO 2017

19

Standardizing Data

```
% Pull out the sensor data, ignoring the unit and timestamp, and
% format it as an array
Xtrain = table2array(engineData(:,3:end));
% Give all sensors mean of zero and standard deviation of one
XtrainMeanTall = mean(Xtrain); % mean of each signal
XtrainStdTall = std(Xtrain); % standard deviation of each signal
% Uses implicit expansion
XtrainStandard = (Xtrain - XtrainMeanTall)./XtrainStdTall
```

```
XtrainStandard =

  Mx14 tall array

    ?    ?    ?   ...
    ?    ?    ?   ...
    ?    ?    ?   ...
    :    :    :
    :    :    :

Preview deferred. Learn more.
```

MATLAB EXPO 2017

20

Deferred evaluation and gathering

```
% read in data, assuming here that one data file can fit into memory
sensorData = gather( engineData( engineData.Unit == 1, : ) )
```

```
Evaluating tall expression using the Local MATLAB Session:
- Pass 1 of 1: Completed in 7 sec
Evaluation completed in 7 sec
  sensorData =  121x16 table
```

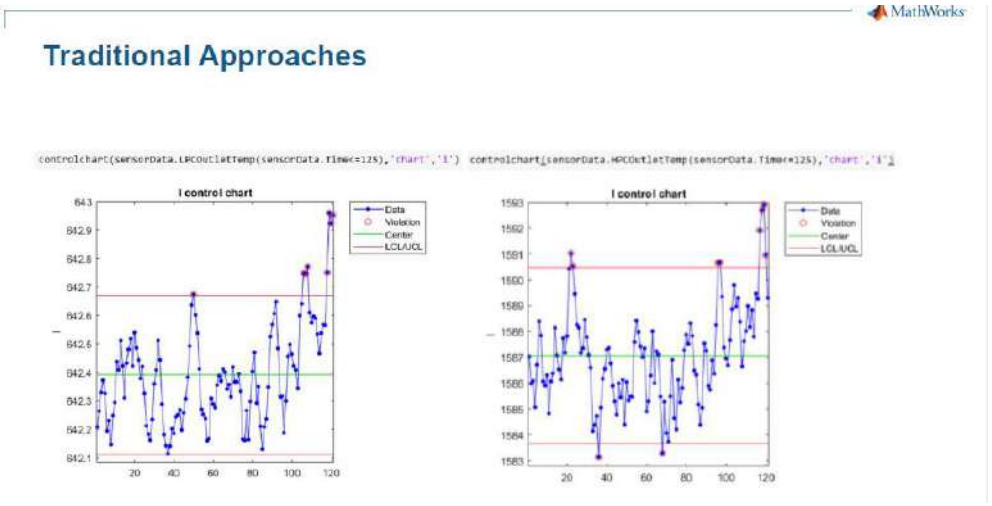
	Unit	Time	LPCOutletTemp	HPCOutletTemp	LPTOutletTemp	TotalHPCOutle...
1	1.00	5.00	642.21	1587.03	1403.21	554.16
2	1.00	6.00	642.26	1585.98	1402.76	554.23
3	1.00	7.00	642.33	1580.08	1401.69	554.34
4	1.00	8.00	642.37	1585.08	1401.04	554.26
5	1.00	9.00	642.33	1588.72	1399.63	554.11
6	1.00	10.00	642.19	1588.39	1398.47	554.03
7	1.00	11.00	642.23	1587.85	1398.93	554.00
8	1.00	12.00	642.15	1588.07	1399.40	554.04
9	1.00	13.00	642.25	1585.91	1399.38	553.96

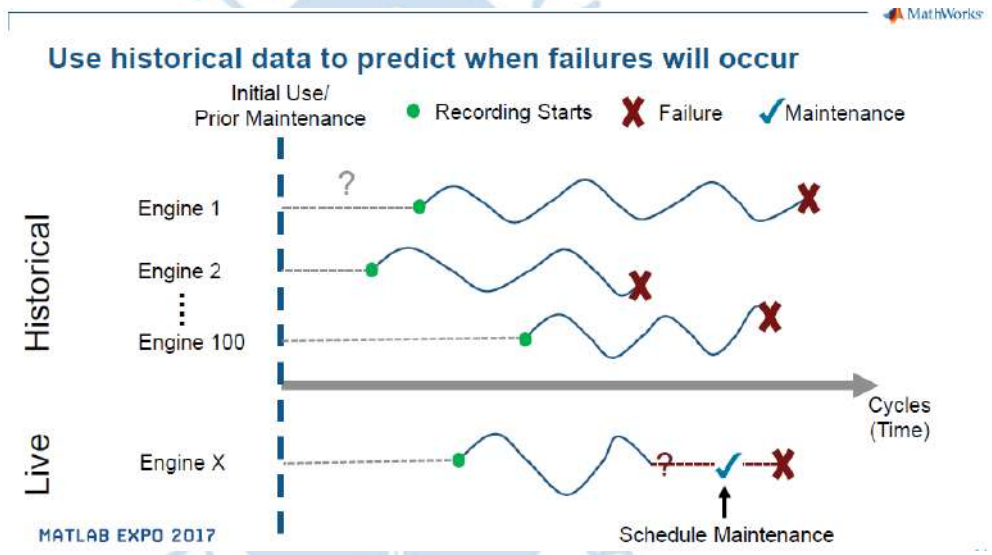
What does "gather" do?

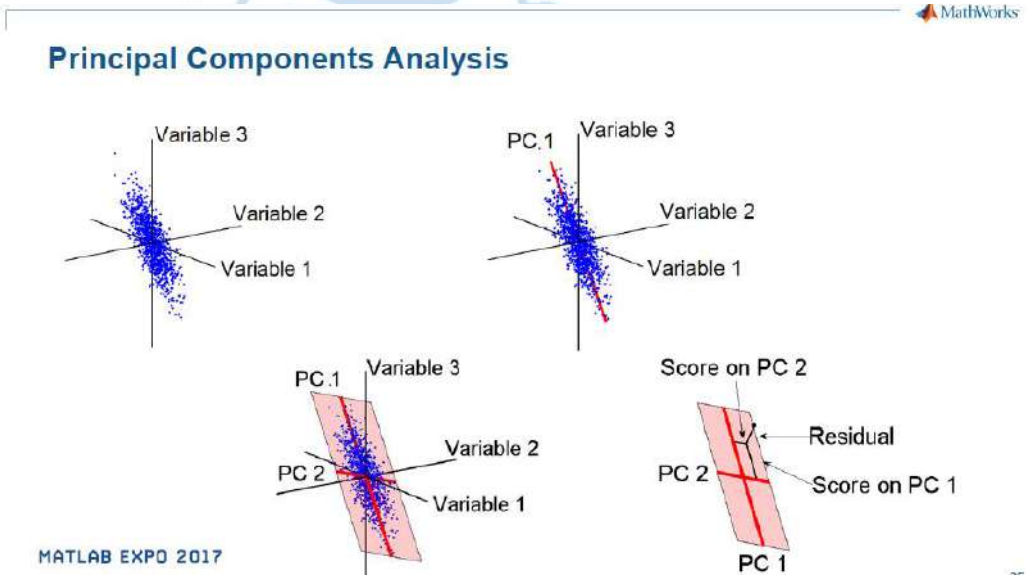
1. **Evaluate** any pending operations
2. **Collect** the partitioned data into MATLAB main memory
3. **Unwrap** the data into an array or table

MATLAB EXPO 2017

Fuente: (Matlab, s.f.)



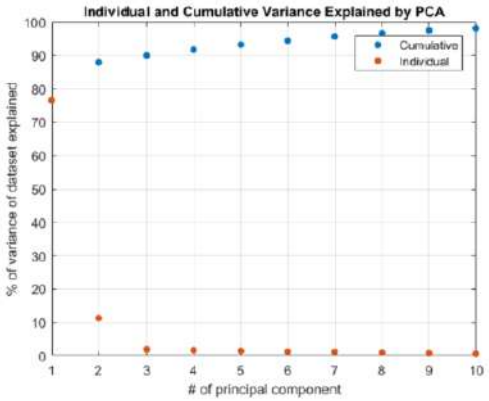




Fuente: (Matlab, s.f.)

Dimensionality Reduction with PCA

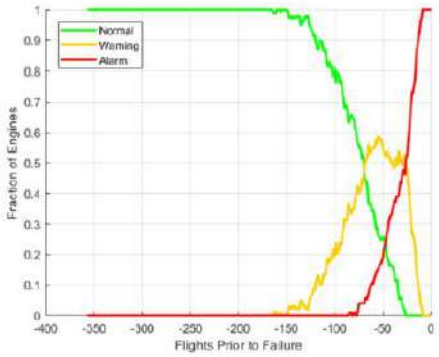
```
[coeff,score,latent] = pca(XtrainStandard);
```

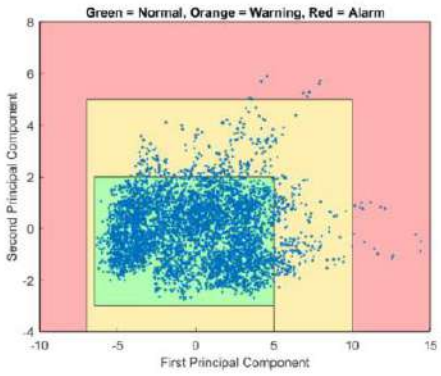


SignalContributions	
LPCOutletTemp	0.27153
HPCOutletTemp	0.25817
LPTOutletTemp	0.28662
TotalHPCOutletPres	-0.2875
PhysFanSpeed	0.29237
PhysCoreSpeed	-0.13822
StaticHPCOutletPres	0.29246
FuelFlowRatio	-0.29173
CorrFanSpeed	0.29192
CorrCoreSpeed	-0.18721
BypassRatio	0.27843
BleedEnthalpy	0.2631
HPTCoolantBleed	-0.27535
LPTCoolantBleed	-0.27706

MATLAB EXPO 2017

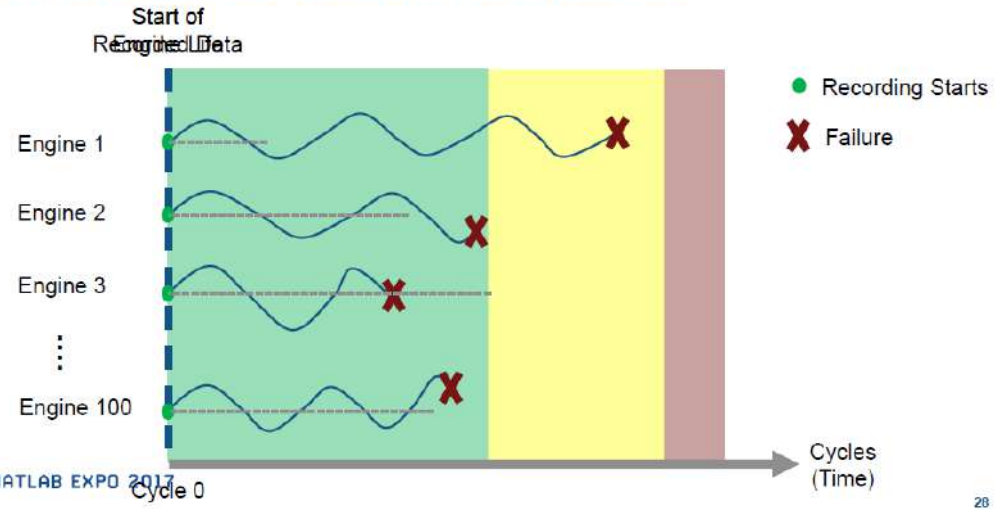
Early Warning System





MATLAB EXPO 2017

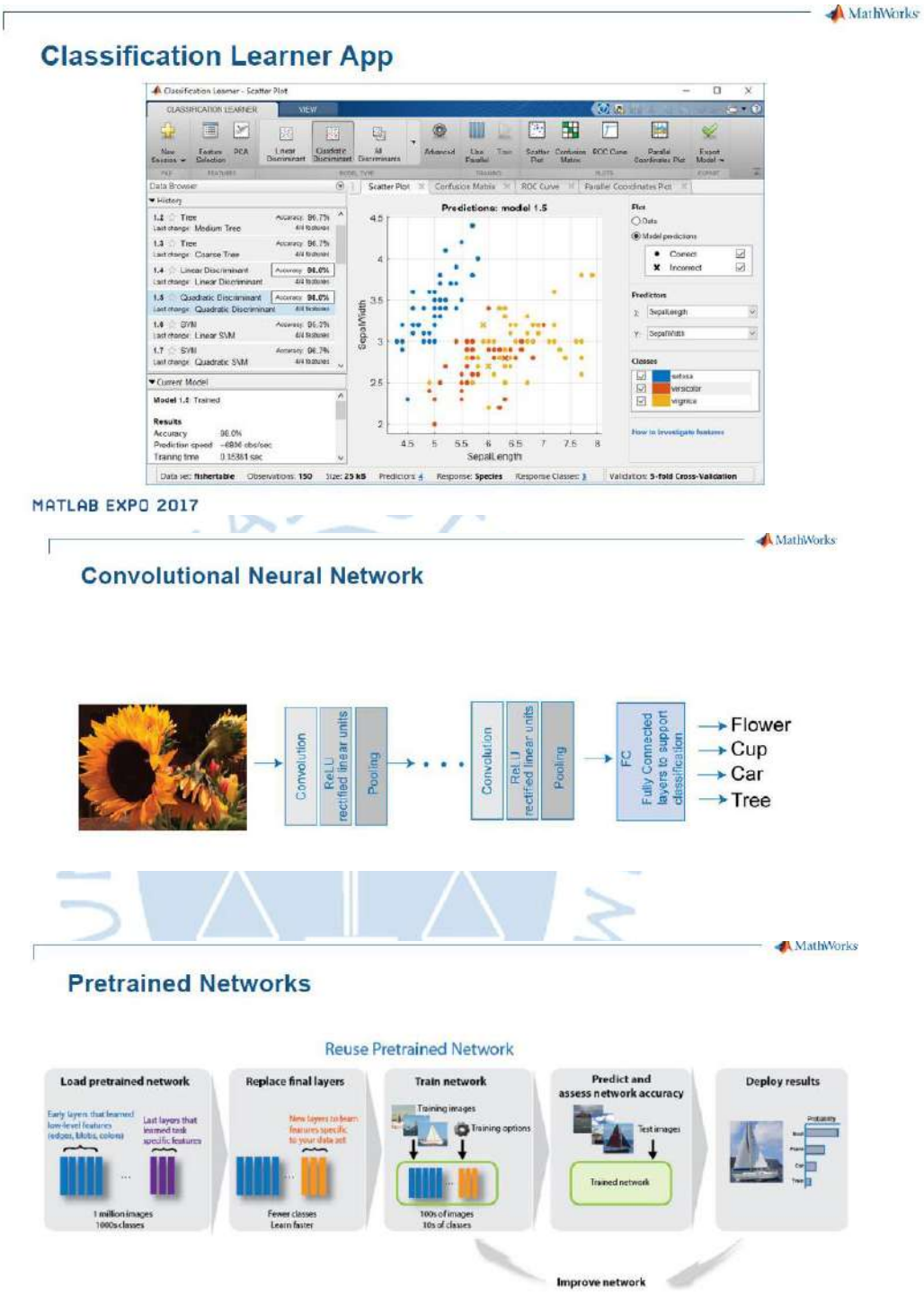
Preprocessing and Classifying our Input Data



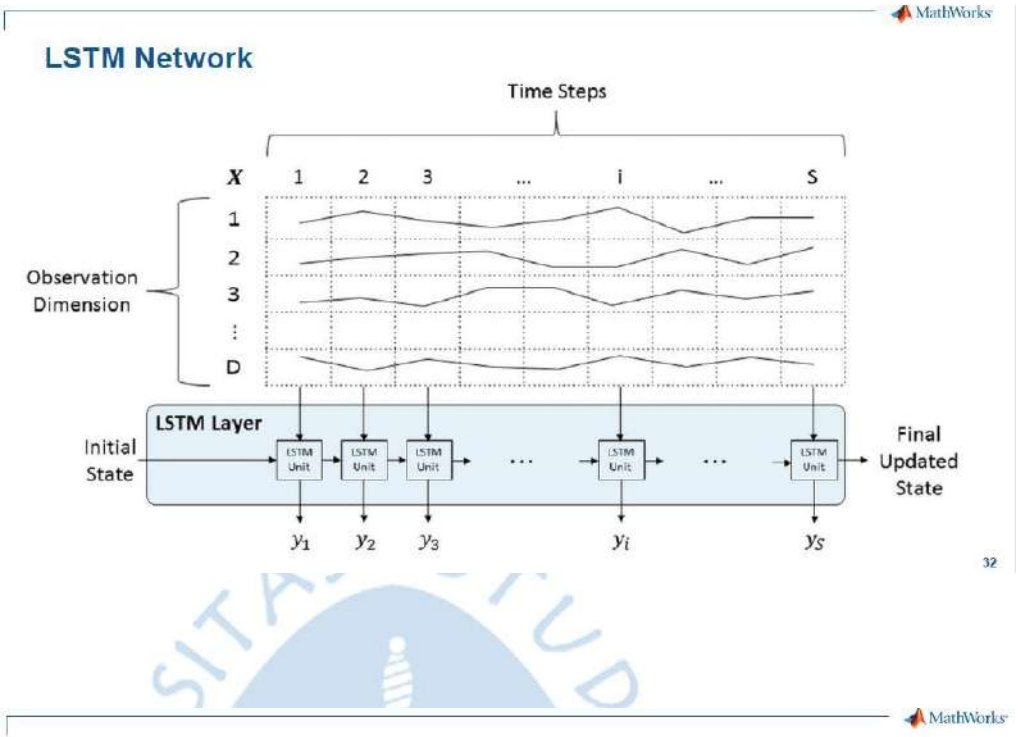
MATLAB EXPO 2017

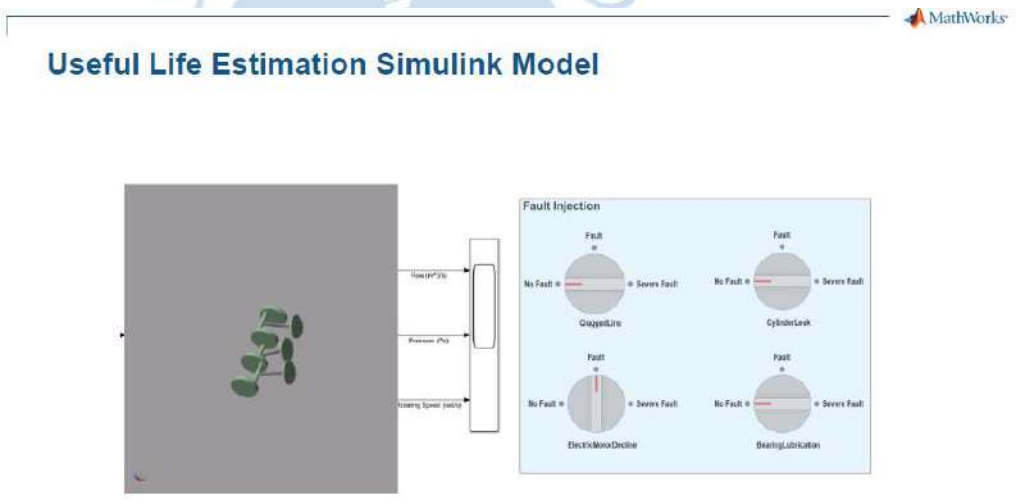
28

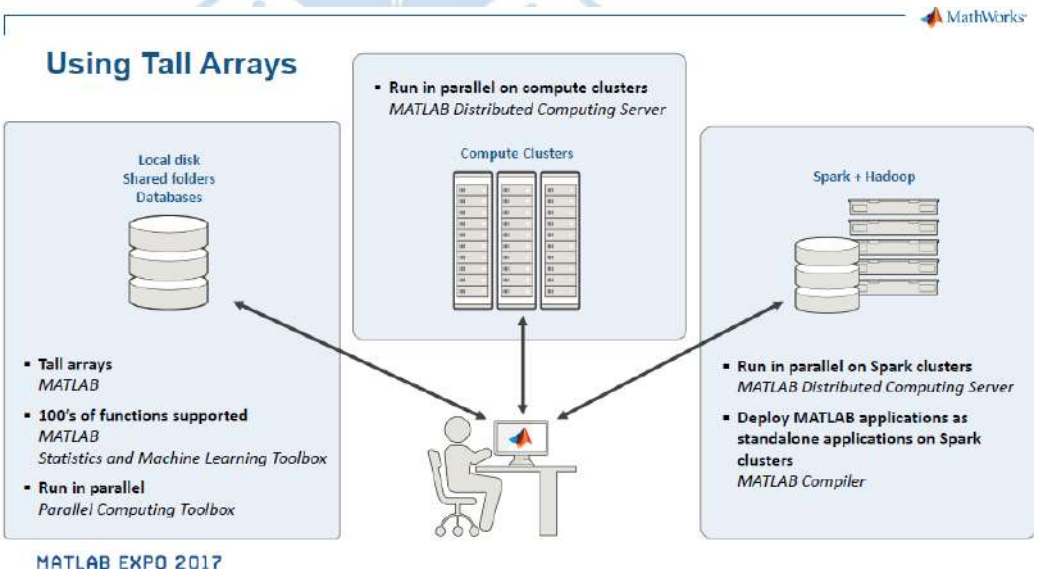
Fuente: (Matlab, s.f.)



Fuente: (Matlab, s.f.)



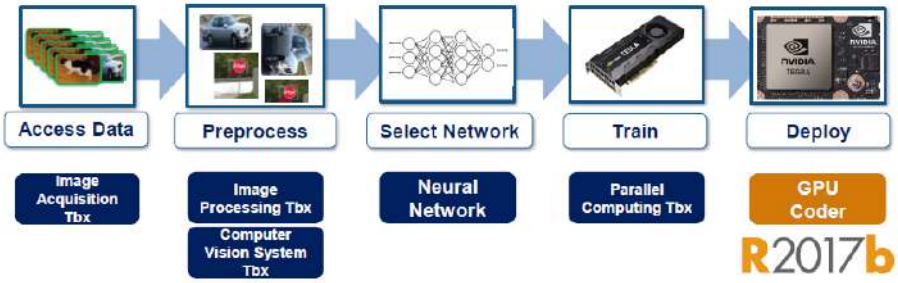




Fuente: (Matlab, s.f.)

MathWorks

Working with GPU Coder: Deep Learning Workflow



MathWorks

Machine Learning on MATLAB Production Server

Shell analyses big data sets to detect events and abnormalities at downstream chemical plants using predictive analytics with MATLAB®. Multivariate statistical models running on MATLAB Production Server™ are used to do real-time batch and process monitoring, enabling real-time interventions when abnormalities are detected.

Big Data and Predictive Analytics at Shell

Anjad Chaudry, Shell

MathWorks

Where Next?

Talks

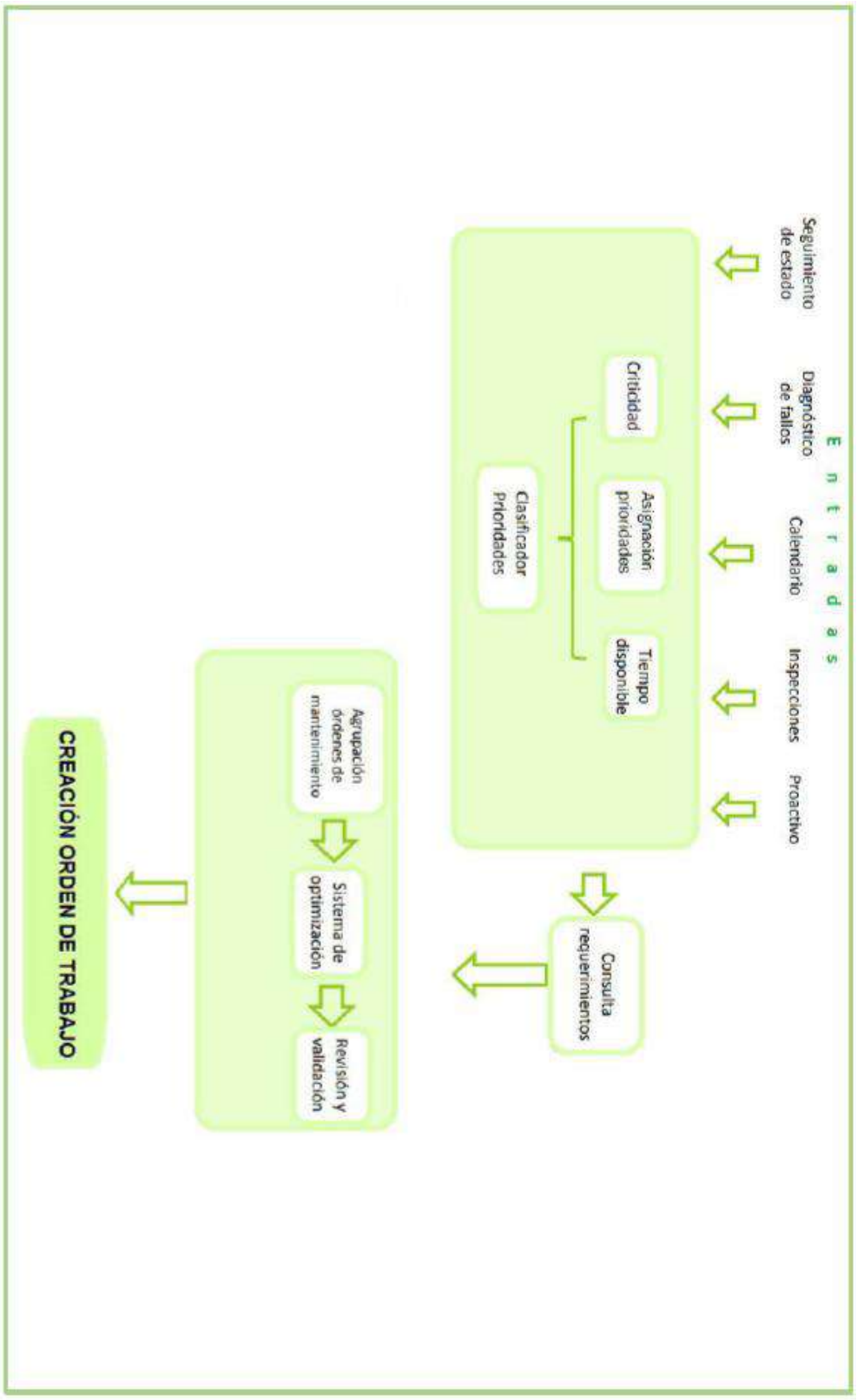
- MatConvNet: Deep Learning Research in MATLAB
- Introduction to Machine & Deep Learning
- Scaling MATLAB for your Organisation and Beyond

Demo Stations

- Big Data with MATLAB
- Deep Learning with MATLAB
- Predictive Maintenance with MATLAB and Simulink
- Deploying Video Processing Algorithms to Hardware
- Using MATLAB and ThingSpeak to Explore the Internet of Things

Fuente: (Matlab, s.f.)

Anexo F. Esquema de creación de orden de trabajo.



Fuente: Elaboración Propia

Anexo G. Código del caso desarrollado con la data del turboventilador de la NASA

```
from google.colab import drive
drive.mount('/content/drive')

# load necessary packages and view available data
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import keras
%matplotlib inline
sns.set()
print(os.listdir("/content/drive/My Drive/data3/CMAPSSData"))
# Setting seed for reproducability
np.random.seed(1234)
PYTHONHASHSEED = 0

# the files did not contain headers. Here we create labels based on documentation
target_var = ['Target_Remaining_Useful_Life']
index_columns_names =  ["UnitNumber","Cycle"]
op_settings_columns = ["Op_Setting_"+str(i) for i in range(1,4)]
sensor_columns =["Sensor_"+str(i) for i in range(1,22)]
column_names = index_columns_names + op_settings_columns + sensor_columns
print(column_names)

# load data
train= pd.read_csv('/content/drive/My Drive/data3/CMAPSSData/train_FD001.txt', sep=" ", header=None)
test = pd.read_csv('/content/drive/My Drive/data3/CMAPSSData/test_FD001.txt', sep=" ", header=None)
print("train shape: ", train.shape, "test shape: ", test.shape)
# drop pesky NULL columns
train.drop(train.columns[[26, 27]], axis=1, inplace=True)
test.drop(test.columns[[26, 27]], axis=1, inplace=True)
# name columns
train.columns = column_names
test.columns = column_names
#train.head(10)
#test.head(10)
train[train['UnitNumber'] == 1].head(5)
test[test['UnitNumber'] == 1].head(5)
```

```
        # this section calculates Remaining Useful Life (RUL) in T-minus notation for the training data
        # find the last cycle per unit number
        max_cycle = train.groupby("UnitNumber")["Cycle"].max().reset_index()
        max_cycle.columns = ['UnitNumber', 'MaxOfCycle']
        # merge the max cycle back into the original frame
        train_merged = train.merge(max_cycle, left_on='UnitNumber', right_on='UnitNumber', how='inner')
        # calculate RUL for each row
        Target_Remaining_Useful_Life = train_merged["MaxOfCycle"] - train_merged["Cycle"]
        train_with_target = train_merged["Target_Remaining_Useful_Life"] = Target_Remaining_Useful_Life
        # remove unnecessary column
        train_with_target = train_merged.drop("MaxOfCycle", axis=1)
        train_with_target[train_with_target['UnitNumber'] == 1].head(5)

        # use seaborn to visualize featuresto target (RUL)
        explore = sns.PairGrid(data=train_with_target.query('UnitNumber < 15') ,
                    x_vars=target_var,
                    y_vars=sensor_columns + op_settings_columns,
                    hue="UnitNumber", size=3, aspect=2.5)
        explore = explore.map(plt.scatter, alpha=0.5)
        explore = explore.set(xlim=(400,0))
        explore = explore.add_legend()

        # operational setting 3 is stable, let's visualize op setting 1 and 2 against some of the most active se
nsors
        g = sns.pairplot(data=train_with_target.query('UnitNumber < 15'),
                    x_vars=["Op_Setting_1","Op_Setting_2"],
                    y_vars=["Sensor_2", "Sensor_3", "Sensor_4", "Sensor_7", "Sensor_8", "Sensor_9", "Sensor_11
", "Sensor_12", "Sensor_13", "Sensor_14", "Sensor_15", "Sensor_17", "Sensor_20", "Sensor_21"],
                    hue="UnitNumber", aspect=1)

        # now it's time to clear out target leakage
        print(train_with_target.shape)
        leakage_to_drop = ['UnitNumber', 'Cycle', 'Op_Setting_1', 'Op_Setting_2', 'Op_Setting_3']
        train_no_leakage = train_with_target.drop(leakage_to_drop, axis = 1)
        print(train_no_leakage.shape)
        # set up features and target variable
        y = train_no_leakage['Target_Remaining_Useful_Life']
        X = train_no_leakage.drop(['Target_Remaining_Useful_Life'], axis = 1)

        # I like to use a simple random forest to determine some of the most important/meaningful feature
s. Can be used as feature selection
        # create an exhuastive random forest (200 trees up to 15 levels deep)
        from sklearn import ensemble
```

```
rf = ensemble.RandomForestRegressor()
single_rf = ensemble.RandomForestRegressor(n_estimators = 200, max_depth = 15)
single_rf.fit(X, y)
y_pred = single_rf.predict(X)
print("complete")

# graph feature importance
import matplotlib.pyplot as plt
importances = single_rf.feature_importances_
indices = np.argsort(importances)[::-1]
feature_names = X.columns
f, ax = plt.subplots(figsize=(11, 9))
plt.title("Feature ranking", fontsize = 20)
plt.bar(range(X.shape[1]), importances[indices], color="b", align="center")
plt.xticks(range(X.shape[1]), indices) #feature_names, rotation='vertical')
plt.xlim([-1, X.shape[1]])
plt.ylabel("importance", fontsize = 18)
plt.xlabel("index of the feature", fontsize = 18)
plt.show()
# list feature importance
important_features = pd.Series(data=single_rf.feature_importances_,index=X.columns)
important_features.sort_values(ascending=False,inplace=True)
print(important_features.head(10))

# based on the graphs as well as random forest feature importance, I will exclude sensors without
much valuable information
print(train_no_leakage.shape)
vars_to_drop = ["Sensor_"+str(i) for i in [5, 15, 9, 17, 4, 18]]
train_final = train_no_leakage.drop(vars_to_drop, axis = 1)
print(train_final.shape)
train_final.head(5)

# identify categorical and numeric fields
from sklearn import preprocessing
categorical = train_final.select_dtypes(include=['object'])
numeric = train_final.select_dtypes(exclude=['object'])
print(categorical.columns.values)
# create dummy variables (if any categorical fields)
for name, values in categorical.items():
    print(name)
    dummies = pd.get_dummies(values.str.strip(), prefix = name, dummy_na=True)
    numeric = pd.concat([numeric, dummies], axis=1)
# imputation (if any NULL values)
```

```
        for name in numeric:
            print(name)
            if pd.isnull(numeric[name]).sum() > 0:
                numeric["%s_mi" % (name)] = pd.isnull(numeric[name])
                median = numeric[name].median()
                numeric[name] = numeric[name].apply(lambda x: median if pd.isnull(x) else x)
        y = numeric['Target_Remaining_Useful_Life']
        X = numeric.drop(['Target_Remaining_Useful_Life'], axis = 1)

        # random forest regression
        # create holdout
        import numpy as np
        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=1234)
        # choose the model
        from sklearn.ensemble import RandomForestRegressor
        rf = ensemble.RandomForestRegressor()
        # set up 5-fold cross-validation
        from sklearn import model_selection
        cv = model_selection.KFold(5)
        # pipeline standardization and model
        from sklearn.pipeline import Pipeline
        pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                                   , ('model', rf) ])
        # tune the model
        my_min_samples_leaf = [2, 10, 25, 50, 100]
        my_max_depth = [7, 8, 9, 10, 11, 12]
        # run the model using gridsearch, select the model with best search
        from sklearn.model_selection import GridSearchCV
        optimized_rf = GridSearchCV(estimator=pipeline
                                    , cv=cv
                                    , param_grid =dict(model__min_samples_leaf = my_min_samples_leaf, model__max_de
pth = my_max_depth)
                                    , scoring = 'neg_mean_squared_error'
                                    , verbose = 1
                                    , n_jobs = -1
                                   )
        optimized_rf.fit(X_train, y_train)
        # show the best model estimators
        print(optimized_rf.best_estimator_)
        # evaluate metrics on holdout
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        y_pred = optimized_rf.predict(X_test)
```

```
print("Random Forest Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("Random Forest Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("Random Forest r-squared: ", r2_score(y_test, y_pred))

# Elastic Net GLM
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn.linear_model import ElasticNet
glm_net = ElasticNet()

# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', glm_net) ])
# tune the model
my_alpha = np.linspace(.01, 1, num=5)
my_l1_ratio = np.linspace(.01, 1, num=3)
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_glm_net = GridSearchCV(estimator=pipeline
                    , cv=cv
                    , param_grid =dict(model__l1_ratio = my_l1_ratio, model__alpha = my_alpha)
                    , scoring = 'neg_mean_squared_error'
                    , verbose = 1
                    , n_jobs = -1
                    )
optimized_glm_net.fit(X_train, y_train)
# show the best model estimators
print(optimized_glm_net.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_glm_net.predict(X_test)
print("GLM Elastic Net Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("GLM Elastic Net Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("GLM Elastic Net r-squared: ", r2_score(y_test, y_pred))

# Support Vector Machines
```

```
# create holdout
import numpy as np
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=1234)
# choose the model
from sklearn import svm
from sklearn.svm import SVR
svm = svm.SVR()
# set up 5-fold cross-validation
from sklearn import model_selection
cv = model_selection.KFold(5)
# pipeline standardization and model
from sklearn.pipeline import Pipeline
pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                    , ('model', svm) ])
# tune the model
my_C = [1]
my_epsilon = [.05, .1, .15]
# run the model using gridsearch, select the model with best search
from sklearn.model_selection import GridSearchCV
optimized_svm = GridSearchCV(estimator=pipeline
                     , cv=cv
                     , param_grid =dict(model__C = my_C, model__epsilon = my_epsilon)
                     , scoring = 'neg_mean_squared_error'
                     , verbose = 1
                     , n_jobs = -1
                    )
optimized_svm.fit(X_train, y_train)
# show the best model estimators
print(optimized_svm.best_estimator_)
# evaluate metrics on holdout
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
y_pred = optimized_svm.predict(X_test)
print("SVM Mean Squared Error: ", mean_squared_error(y_test, y_pred))
print("SVM Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
print("SVM r-squared: ", r2_score(y_test, y_pred))

# Gradient Boosting
# create holdout
import numpy as np
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=1234)
# choose the model
```

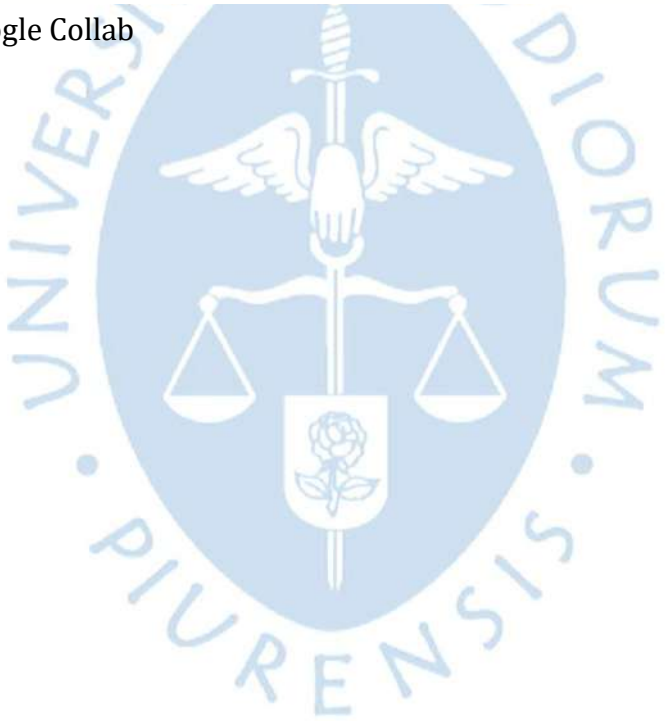
```
        from sklearn.ensemble import GradientBoostingRegressor
        gb = ensemble.GradientBoostingRegressor()
        # set up 5-fold cross-validation
        from sklearn import model_selection
        cv = model_selection.KFold(5)
        # pipeline standardization and model
        from sklearn.pipeline import Pipeline
        pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                                   , ('model', gb) ])
        # tune the model
        my_alpha = [.5, .75, .9]
        my_n_estimators= [500]
        my_learning_rate = [0.005, .01]
        my_max_depth = [4, 5, 6]
        # run the model using gridsearch, select the model with best search
        from sklearn.model_selection import GridSearchCV
        optimized_gb = GridSearchCV(estimator=pipeline
                                    , cv=cv
                                    , param_grid =dict(model__max_depth = my_max_depth, model__n_estimators = my_n
_estimators,
                                                       model__learning_rate = my_learning_rate, model__alpha = my_alpha)
                                    , scoring = 'neg_mean_squared_error'
                                    , verbose = 1
                                    , n_jobs = -1
                                    )
        optimized_gb.fit(X_train, y_train)
        # show the best model estimators
        print(optimized_gb.best_estimator_)
        # evaluate metrics on holdout
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        y_pred = optimized_gb.predict(X_test)
        print("Gradient Boosting Mean Squared Error: ", mean_squared_error(y_test, y_pred))
        print("Gradient Boosting Mean Absolute Error: ", mean_absolute_error(y_test, y_pred))
        print("Gradient Boosting r-squared: ", r2_score(y_test, y_pred))

        # plot actual vs predicted Remaining Useful Life for the best model (GBM)
        fig, ax = plt.subplots()
        ax.scatter(y_test, y_pred, edgecolors=(0, 0, 0))
        ax.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
        ax.set_xlabel('Actual RUL')
        ax.set_ylabel('Predicted RUL')
        ax.set_title('Remaining Useful Life Actual vs. Predicted')
        plt.show()
```

```
        # now let's look at turning this into a classification sol -
> can we confidently identify when an asset within its last 15 cycles?
        # generate label columns for training data
        cycles = 15
        train_no_leakage['Target_15_Cycles'] = np.where(train_no_leakage['Target_Remaining_Useful_Life']
<= cycles, 1, 0 )
        train_no_leakage.tail(5)

        # based on the graphs as well as random forest feature importance, I will exclude sensors without
much valuable information
        print(train_no_leakage.shape)
        vars_to_drop = ["Sensor_"+str(i) for i in [5, 15, 9, 17, 4, 18]]
        target_to_drop = ['Target_Remaining_Useful_Life']
        train_final = train_no_leakage.drop(vars_to_drop, axis = 1)
        train_final = train_no_leakage.drop(target_to_drop, axis = 1)
        train_final.tail()

        # identify categorical and numeric fields
        from sklearn import preprocessing
        categorical = train_final.select_dtypes(include=['object'])
        numeric = train_final.select_dtypes(exclude=['object'])
        print(categorical.columns.values)
        # create dummy variables (if any categorical fields)
        for name, values in categorical.items():
            print(name)
            dummies = pd.get_dummies(values.str.strip(), prefix = name, dummy_na=True)
            numeric = pd.concat([numeric, dummies], axis=1)
        # imputation (if any NULL values)
        for name in numeric:
            print(name)
            if pd.isnull(numeric[name]).sum() > 0:
                numeric["%s_mi" % (name)] = pd.isnull(numeric[name])
                median = numeric[name].median()
                numeric[name] = numeric[name].apply(lambda x: median if pd.isnull(x) else x)
        y = numeric['Target_15_Cycles']
        X = numeric.drop(['Target_15_Cycles'], axis = 1)

        # random forest regression
        # create holdout
        import numpy as np
        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=1234)
        # choose the model
```

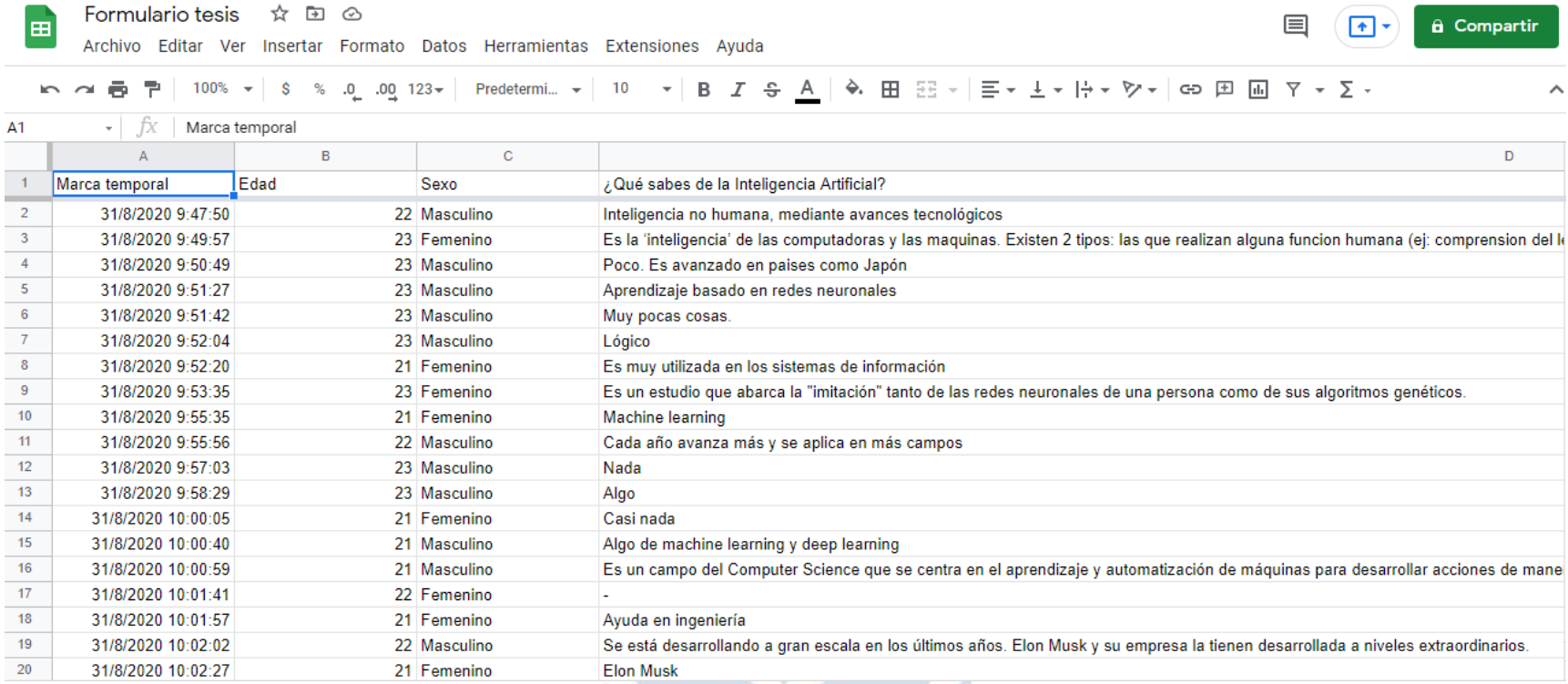
```
        from sklearn import ensemble
        from sklearn.ensemble import RandomForestClassifier
        rf = ensemble.RandomForestClassifier()
        # set up 5-fold cross-validation
        from sklearn import model_selection
        cv = model_selection.KFold(5)
        # pipeline standardization and model
        from sklearn.pipeline import Pipeline
        pipeline = Pipeline(steps=[('standardize', preprocessing.StandardScaler())
                        , ('model', rf) ])
        # tune the model
        my_min_samples_leaf = [2, 25, 50]
        my_max_depth = [8, 9, 10, 12]
        # run the model using gridsearch, select the model with best search
        from sklearn.model_selection import GridSearchCV
        optimized_rf = GridSearchCV(estimator=pipeline
                        , cv=cv
                        , param_grid =dict(model__min_samples_leaf = my_min_samples_leaf, model__max_de
pth = my_max_depth)
                        , scoring = 'roc_auc'
                        , verbose = 1
                        , n_jobs = -1
                        )
        optimized_rf.fit(X_train, y_train)
        # show the best model estimators
        y_pred_proba = optimized_rf.predict_proba(X_test)[:, 1]
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        y_pred = optimized_rf.predict(X_test)
        print(optimized_rf.best_estimator_)
        print("Random Forest Mean Squared Error 2: ", mean_squared_error(y_test, y_pred))
        print("Random Forest Mean Absolute Error 2: ", mean_absolute_error(y_test, y_pred))
        print("Random Forest r-squared 2: ", r2_score(y_test, y_pred))

        from sklearn.metrics import roc_auc_score, accuracy_score, precision_score, recall_score, confusion
_matrix
        print("Confusion Matrix:")
        print(confusion_matrix(y_test, y_pred))
        from sklearn.metrics import classification_report
        print("Random Forest Accuracy: "+"{:.1%}".format(accuracy_score(y_test, y_pred)));
        print("Random Forest Precision: "+"{:.1%}".format(precision_score(y_test, y_pred)));
        print("Random Forest Recall: "+"{:.1%}".format(recall_score(y_test, y_pred)));
        print("Classification Report:")
        print(classification_report(y_test, y_pred))
```

```
from sklearn import metrics
fpr, tpr, threshold = metrics.roc_curve(y_test, y_pred_proba)
roc_auc = metrics.auc(fpr, tpr)
import matplotlib.pyplot as plt
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

Fuente: Google Collab



Anexo H. Encuesta realizada sobre el conocimiento de la IA



Formulario tesis

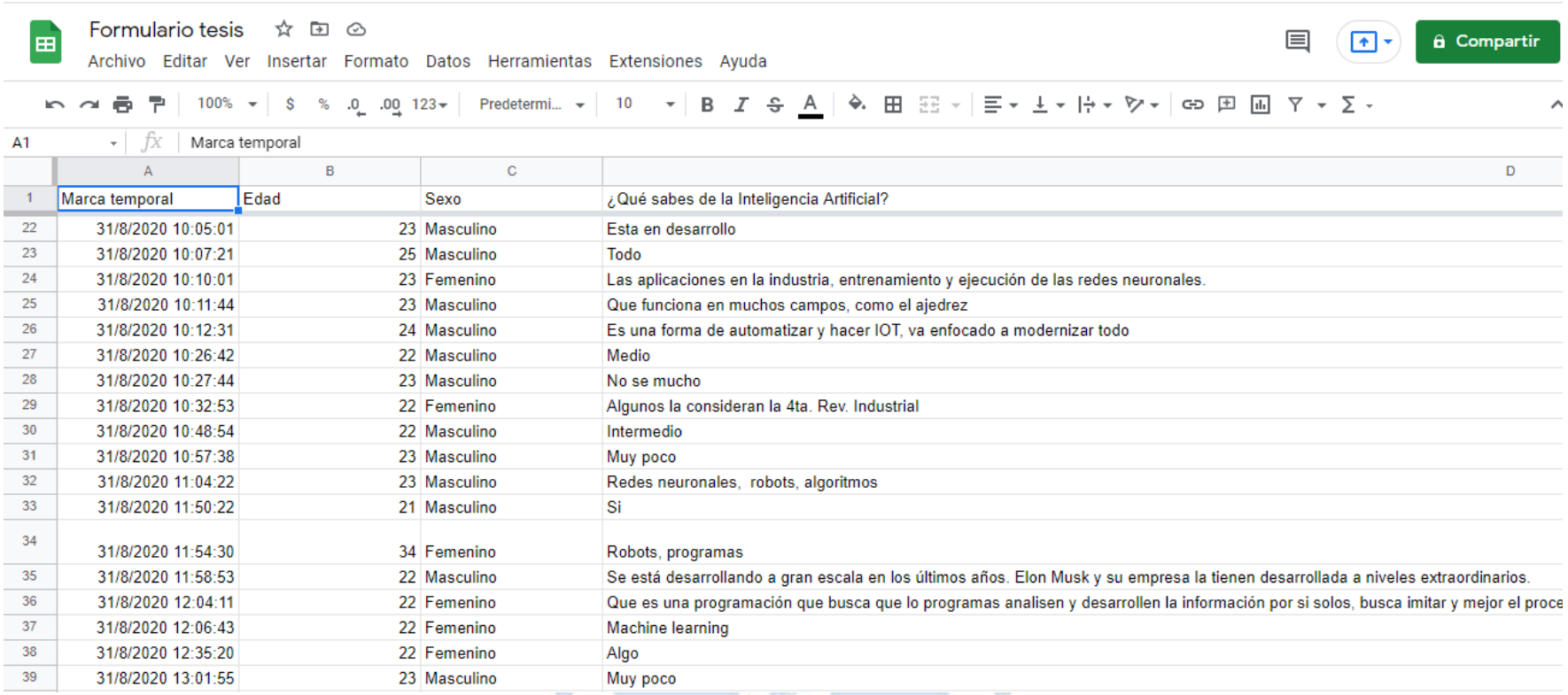
Archivo Editar Ver Insertar Formato Datos Herramientas Extensiones Ayuda

Compartir

A1 | Marca temporal

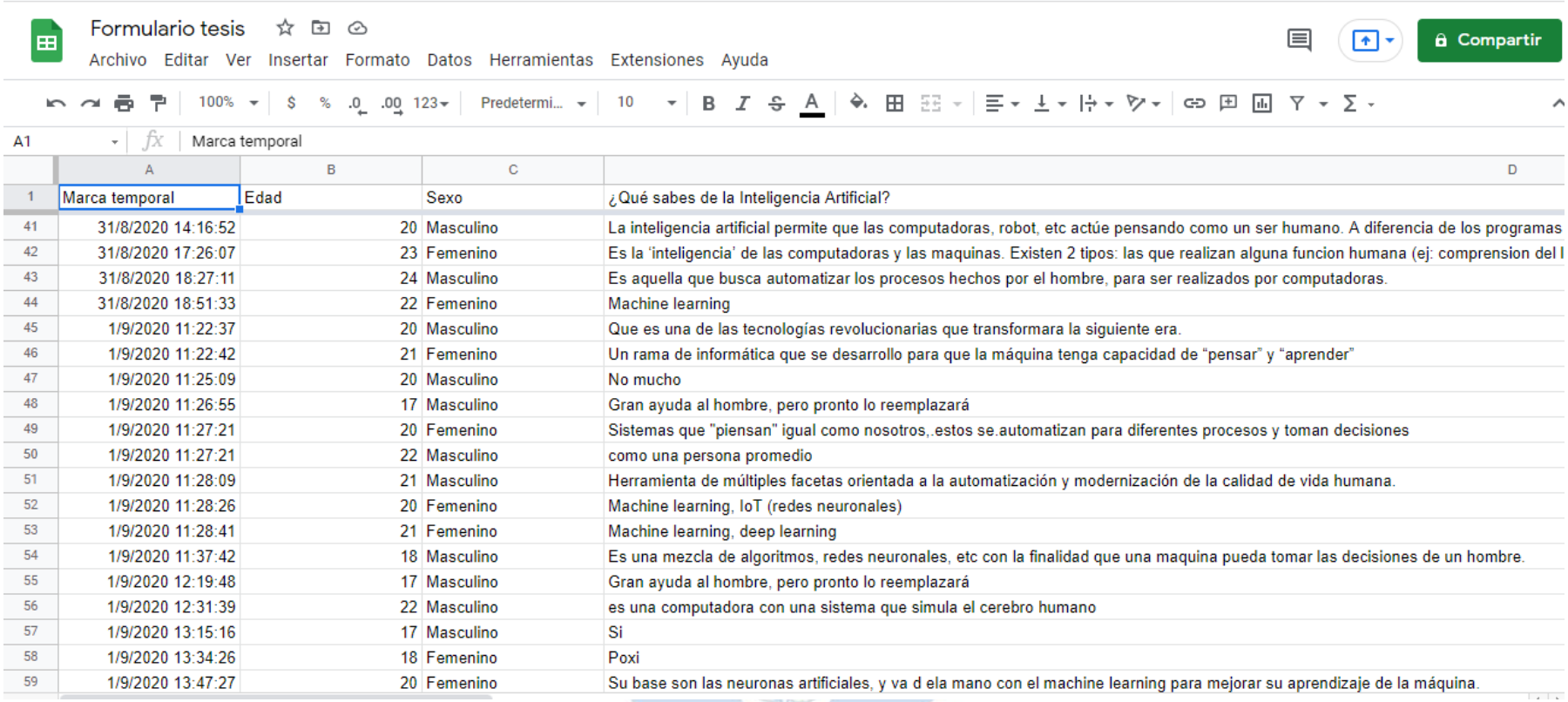
	A	B	C	D
1	Marca temporal	Edad	Sexo	¿Qué sabes de la Inteligencia Artificial?
2	31/8/2020 9:47:50	22	Masculino	Inteligencia no humana, mediante avances tecnológicos
3	31/8/2020 9:49:57	23	Femenino	Es la 'inteligencia' de las computadoras y las maquinas. Existen 2 tipos: las que realizan alguna funcion humana (ej: comprension del l
4	31/8/2020 9:50:49	23	Masculino	Poco. Es avanzado en paises como Japón
5	31/8/2020 9:51:27	23	Masculino	Aprendizaje basado en redes neuronales
6	31/8/2020 9:51:42	23	Masculino	Muy pocas cosas.
7	31/8/2020 9:52:04	23	Masculino	Lógico
8	31/8/2020 9:52:20	21	Femenino	Es muy utilizada en los sistemas de información
9	31/8/2020 9:53:35	23	Femenino	Es un estudio que abarca la "imitación" tanto de las redes neuronales de una persona como de sus algoritmos genéticos.
10	31/8/2020 9:55:35	21	Femenino	Machine learning
11	31/8/2020 9:55:56	22	Masculino	Cada año avanza más y se aplica en más campos
12	31/8/2020 9:57:03	23	Masculino	Nada
13	31/8/2020 9:58:29	23	Masculino	Algo
14	31/8/2020 10:00:05	21	Femenino	Casi nada
15	31/8/2020 10:00:40	21	Masculino	Algo de machine learning y deep learning
16	31/8/2020 10:00:59	21	Masculino	Es un campo del Computer Science que se centra en el aprendizaje y automatización de máquinas para desarrollar acciones de mane
17	31/8/2020 10:01:41	22	Femenino	-
18	31/8/2020 10:01:57	21	Femenino	Ayuda en ingeniería
19	31/8/2020 10:02:02	22	Masculino	Se está desarrollando a gran escala en los últimos años. Elon Musk y su empresa la tienen desarrollada a niveles extraordinarios.
20	31/8/2020 10:02:27	21	Femenino	Elon Musk

Fuente: Encuesta realizada en 2019 a estudiantes de la facultad de ingeniería



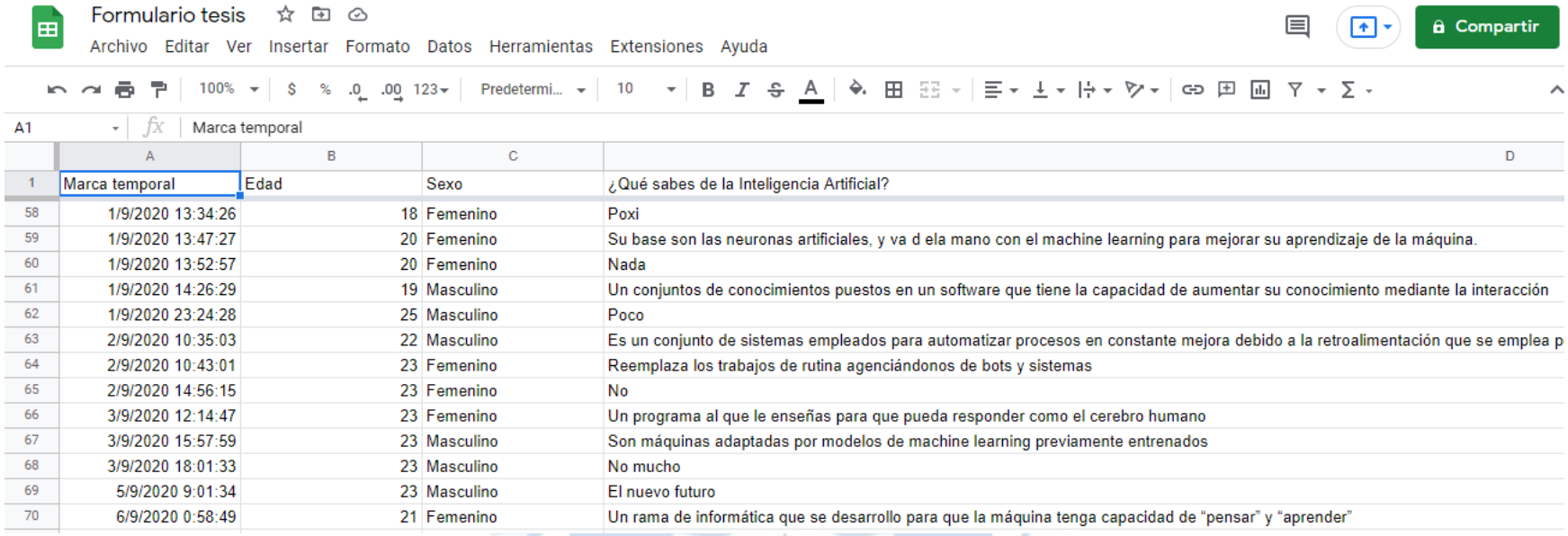
	A	B	C	D
1	Marca temporal	Edad	Sexo	¿Qué sabes de la Inteligencia Artificial?
22	31/8/2020 10:05:01	23	Masculino	Esta en desarrollo
23	31/8/2020 10:07:21	25	Masculino	Todo
24	31/8/2020 10:10:01	23	Femenino	Las aplicaciones en la industria, entrenamiento y ejecución de las redes neuronales.
25	31/8/2020 10:11:44	23	Masculino	Que funciona en muchos campos, como el ajedrez
26	31/8/2020 10:12:31	24	Masculino	Es una forma de automatizar y hacer IOT, va enfocado a modernizar todo
27	31/8/2020 10:26:42	22	Masculino	Medio
28	31/8/2020 10:27:44	23	Masculino	No se mucho
29	31/8/2020 10:32:53	22	Femenino	Algunos la consideran la 4ta. Rev. Industrial
30	31/8/2020 10:48:54	22	Masculino	Intermedio
31	31/8/2020 10:57:38	23	Masculino	Muy poco
32	31/8/2020 11:04:22	23	Masculino	Redes neuronales, robots, algoritmos
33	31/8/2020 11:50:22	21	Masculino	Si
34	31/8/2020 11:54:30	34	Femenino	Robots, programas
35	31/8/2020 11:58:53	22	Masculino	Se está desarrollando a gran escala en los últimos años. Elon Musk y su empresa la tienen desarrollada a niveles extraordinarios.
36	31/8/2020 12:04:11	22	Femenino	Que es una programación que busca que lo programas analisen y desarrollen la información por si solos, busca imitar y mejor el proce
37	31/8/2020 12:06:43	22	Femenino	Machine learning
38	31/8/2020 12:35:20	22	Femenino	Algo
39	31/8/2020 13:01:55	23	Masculino	Muy poco

Fuente: Encuesta realizada en 2019 a estudiantes de la facultad de ingeniería.



	A	B	C	D
1	Marca temporal	Edad	Sexo	¿Qué sabes de la Inteligencia Artificial?
41	31/8/2020 14:16:52	20	Masculino	La inteligencia artificial permite que las computadoras, robot, etc actúe pensando como un ser humano. A diferencia de los programas
42	31/8/2020 17:26:07	23	Femenino	Es la 'inteligencia' de las computadoras y las maquinas. Existen 2 tipos: las que realizan alguna funcion humana (ej: comprension del l
43	31/8/2020 18:27:11	24	Masculino	Es aquella que busca automatizar los procesos hechos por el hombre, para ser realizados por computadoras.
44	31/8/2020 18:51:33	22	Femenino	Machine learning
45	1/9/2020 11:22:37	20	Masculino	Que es una de las tecnologías revolucionarias que transformara la siguiente era.
46	1/9/2020 11:22:42	21	Femenino	Un rama de informática que se desarrollo para que la máquina tenga capacidad de "pensar" y "aprender"
47	1/9/2020 11:25:09	20	Masculino	No mucho
48	1/9/2020 11:26:55	17	Masculino	Gran ayuda al hombre, pero pronto lo reemplazará
49	1/9/2020 11:27:21	20	Femenino	Sistemas que "piensan" igual como nosotros,.estos se.automatizan para diferentes procesos y toman decisiones
50	1/9/2020 11:27:21	22	Masculino	como una persona promedio
51	1/9/2020 11:28:09	21	Masculino	Herramienta de múltiples facetas orientada a la automatización y modernización de la calidad de vida humana.
52	1/9/2020 11:28:26	20	Femenino	Machine learning, IoT (redes neuronales)
53	1/9/2020 11:28:41	21	Femenino	Machine learning, deep learning
54	1/9/2020 11:37:42	18	Masculino	Es una mezcla de algoritmos, redes neuronales, etc con la finalidad que una maquina pueda tomar las decisiones de un hombre.
55	1/9/2020 12:19:48	17	Masculino	Gran ayuda al hombre, pero pronto lo reemplazará
56	1/9/2020 12:31:39	22	Masculino	es una computadora con una sistema que simula el cerebro humano
57	1/9/2020 13:15:16	17	Masculino	Si
58	1/9/2020 13:34:26	18	Femenino	Poxi
59	1/9/2020 13:47:27	20	Femenino	Su base son las neuronas artificiales, y va d ela mano con el machine learning para mejorar su aprendizaje de la máquina.

Fuente: Encuesta realizada en 2019 a estudiantes de la facultad de ingeniería.



	A	B	C	D
1	Marca temporal	Edad	Sexo	¿Qué sabes de la Inteligencia Artificial?
58	1/9/2020 13:34:26	18	Femenino	Poxi
59	1/9/2020 13:47:27	20	Femenino	Su base son las neuronas artificiales, y va d ela mano con el machine learning para mejorar su aprendizaje de la máquina.
60	1/9/2020 13:52:57	20	Femenino	Nada
61	1/9/2020 14:26:29	19	Masculino	Un conjuntos de conocimientos puestos en un software que tiene la capacidad de aumentar su conocimiento mediante la interacción
62	1/9/2020 23:24:28	25	Masculino	Poco
63	2/9/2020 10:35:03	22	Masculino	Es un conjunto de sistemas empleados para automatizar procesos en constante mejora debido a la retroalimentación que se emplea p
64	2/9/2020 10:43:01	23	Femenino	Reemplaza los trabajos de rutina agenciándonos de bots y sistemas
65	2/9/2020 14:56:15	23	Femenino	No
66	3/9/2020 12:14:47	23	Femenino	Un programa al que le enseñas para que pueda responder como el cerebro humano
67	3/9/2020 15:57:59	23	Masculino	Son máquinas adaptadas por modelos de machine learning previamente entrenados
68	3/9/2020 18:01:33	23	Masculino	No mucho
69	5/9/2020 9:01:34	23	Masculino	El nuevo futuro
70	6/9/2020 0:58:49	21	Femenino	Un rama de informática que se desarrollo para que la máquina tenga capacidad de "pensar" y "aprender"

Fuente: Encuesta realizada en 2019 a estudiantes de la facultad de ingeniería.