UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MANTENIMIENTO AUTOMOTRIZ

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MANTENIMIENTO AUTOMOTRIZ

TEMA: GESTIÓN DEL MANTENIMIENTO VEHICULAR A BASE DEL APRENDIZAJE AUTÓNOMO EN MOTORES DE TRACTORES AGRÍCOLAS.

AUTOR: HÉCTOR ENRIQUE CANGÁS ORTEGA

DIRECTOR: ING. MAFLA YÉPEZ CARLOS NOLASCO MSC.

Ibarra, 2021

CERTIFICADO

ACEPTACIÓN DEL DIRECTOR

En mi calidad de Director del plan de trabajo de grado, previo a la obtención del título de Ingeniero en Mantenimiento Automotriz, nombrado por el Honorable Consejo Directivo de la Facultad de Ingeniería en Ciencias Aplicadas.

CERTIFICO:

Que una vez analizado el plan de grado cuyo título es “GESTIÓN DEL MANTENIMIENTO VEHICULAR A BASE DEL APRENDIZAJE AUTÓNOMO EN MOTORES DE TRACTORES AGRÍCOLAS” presentado por el señor: Cangás Ortega Héctor Enrique con número de cedula 040166704-3, doy fe que dicho trabajo tiene los requisitos y méritos suficientes para ser sometido a presentación pública y evaluación por parte de los señores integrantes del jurado examinador que se designe.

En la ciudad de Ibarra, a los 21 días del mes de octubre del 2021

Atentamente

Firmado electrónicamente por:
CARLOS NOLASCO MAFLA YEPEZ

Ing. Carlos Mafla Yépez MSc.
DIRECTOR DEL TRABAJO DE GRADO.

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	040166704-3		
APELLIDOS Y NOMBRES:	CANGÁS ORTEGA HÉCTOR ENRIQUE		
DIRECCIÓN:	SAN GABRIEL – CARCHI		
EMAIL:	hecangaso@utn.edu.ec		
TELEFONO FIJO:	—	**TELEFONO MÓVIL:**	0997536751

DATOS DE LA OBRA	
TÍTULO:	GESTIÓN DEL MANTENIMIENTO VEHICULAR A BASE DEL APRENDIZAJE AUTÓNOMO EN MOTORES DE TRACTORES AGRÍCOLAS.
AUTOR:	CANGÁS ORTEGA HÉCTOR ENRIQUE
FECHA:	21/10/2021
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	[X] PREGRADO [ ] POSGRADO
TITULO POR EL QUE OPTA:	INGENIERIA EN MANTENIMIENTO AUTOMOTRÍZ
ASESOR/DIRECTOR	ING. CARLOS NOLASCO MAFLA YEPEZ MSC.

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por partes de terceros.

Ibarra, a los 21 días del mes de octubre de 2021

EL AUTOR:

.....
Héctor Enrique Cangás Ortega

DEDICATORIA

Dedico este trabajo especialmente a mis padres, quienes han sido un apoyo fundamental en mi vida y gracias a su amor, trabajo y sacrificio me han podido ayudar a cumplir esta anhelada meta.

Héctor Enrique Cangás Ortega

AGRADECIMIENTO

Dentro de estas líneas quiero agradecer especialmente a mi padre por guiarme hacia esta carrera y a mi madre por apoyarme incondicionalmente hasta el final.

Agradezco a la carrera de Ingeniería Automotriz y a sus docentes por haberme impartido sus conocimientos, y en especial a mi director de tesis Ing. Carlos Mafla quien me acompaño con su tutoría dentro de esta investigación.

Héctor Enrique Cangás Ortega

ÍNDICE.

ÍNDICE DE FIGURAS.

ÍNDICE DE TABLAS.

RESUMEN.

El presente trabajo ha sido desarrollado con el objetivo de usar el Machine Learning como una herramienta de ayuda en la gestión del mantenimiento vehicular, por medio de las vibraciones captadas del motor de un tractor agrícola es posible conocer el estado real en el que este se encuentra, usando un algoritmo de clasificación proporcionado por MATLAB® esto se hace posible. El tipo de aprendizaje autónomo empleado es del tipo supervisado, es decir la intervención humana aquí es necesaria dentro de todo el proceso, inicialmente se caracterizó y etiqueto las muestras de datos para crear una tabla necesaria para la ejecución del entrenamiento del algoritmo de clasificación.

En cuanto a la toma de datos, se optó por simular fallas dentro del sistema de alimentación de combustible del motor, al alterar dicho sistema hay anomalías dentro de la combustión de cada cilindro lo que ocasiona una falla fácilmente detectable por medio de las vibraciones captadas por un sensor. Para efectuar el entrenamiento del algoritmo de clasificación se ha usado 4 estados del motor, la etiqueta de BE pertenece a un estado óptimo de funcionamiento donde no se ha simulado ninguna falla, las etiquetas de ME, MEF1, MEF2, MEF3, pertenecen a las fallas simuladas las cuales tendrán valores distintos cada una. Los entrenamientos realizados dentro del algoritmo de clasificación poseen en todas las pruebas una eficiencia superior al 90%, esto quiere decir que esta alternativa tomada en cuenta es una opción viable capaz de ser usada dentro de la gestión del mantenimiento vehicular, siempre y cuando se contenga una biblioteca amplia de datos es posible predecir fallas en cuestión de minutos.

La evolución dentro de la industria automotriz ha sido notoria dentro de los últimos años, de igual forma el mantenimiento vehicular necesita subir de nivel y cambiar los mantenimientos convencionales a uno basado en el estado real del automotor, en esta investigación se propone eso usando vibraciones como fuente de información, que por medio del Machine Learning sea posible predecir fallas y usar de mejor forma los recursos dentro del mantenimiento vehicular.

ABSTRACT.

This work has been developed with the objective of using Machine Learning as a tool to help in the management of vehicle maintenance, by means of the vibrations captured from the engine of an agricultural tractor it is possible to know the real state in which it is, using a classification algorithm provided by MATLAB® this is made possible. The type of autonomous learning used is of the supervised type, i.e. human intervention is necessary throughout the process, initially the data samples were characterized and labeled to create a table necessary for the execution of the training of the classification algorithm.

As for the data collection, it was chosen to simulate failures within the engine fuel supply system, by altering this system there are anomalies within the combustion of each cylinder which causes a failure easily detectable by means of the vibrations captured by a sensor. To perform the training of the classification algorithm 4 engine states have been used, the BE label belongs to an optimal operating state where no failure has been simulated, the labels of ME, MEF1, MEF2, MEF3, belong to the simulated failures which will have different values each one. The trainings performed within the classification algorithm have in all tests an efficiency higher than 90%, this means that this alternative taken into account is a viable option capable of being used within the vehicle maintenance management, as long as it contains a large library of data, it is possible to predict failures in a matter of minutes.

The evolution within the automotive industry has been notorious in recent years, likewise vehicle maintenance needs to raise the level and change the conventional maintenance to one based on the actual state of the vehicle, this research proposes that using vibrations as a source of information, that through Machine Learning is possible to predict failures and better use of resources within the vehicle maintenance.

INTRODUCCIÓN.

Desde su invención, los automotores han necesitado una constante intervención humana que garantice su correcto funcionamiento, ha esto se lo conoce como mantenimiento vehicular el cual es principalmente de forma preventiva y correctiva, aunque actualmente una mejor opción y más eficiente se ha venido desarrollando y es el mantenimiento predictivo. Dentro de las alternativas usadas en cuanto a la implementación del mantenimiento predictivo se ha planteado iniciar por el mantenimiento según el estado del automotor, para esto el Machine Learning ha sido de vital importancia dentro del análisis y procesamiento de datos. Los algoritmos de Aprendizaje Autónomo se han implementado con diferentes principios y funcionalidades, principalmente se puede nombrar la clasificación, regresión y agrupación, usados correctamente es posible diagnosticar y predecir fallas usando únicamente datos característicos del motor, como lo pueden ser las vibraciones.

El Machine Learning es una herramienta de aprendizaje útil dentro de muchos campos, usarla dentro del mantenimiento vehicular es una opción viable y de gran ayuda en cuanto al diagnóstico y predicción de fallas, no obstante, es necesario tener la información suficiente para ejecutar un entrenamiento realmente útil y capaz de ser usado. Dentro del aprendizaje autónomo se tiene dos alternativas en cuanto al entrenamiento, puede ser supervisado o no supervisado, el uno del otro varia únicamente por el nivel de intervención humana que poseen, siendo el primer caso la opción tomada dentro del mantenimiento predictivo. El aprendizaje autónomo supervisado necesita datos previamente procesados por un entrenador, dichos datos deben estar correctamente caracterizados y etiquetados para que así sirvan como referencia inicial para el algoritmo de Machine Learning, además es importante que sean datos reales.

MATLAB® es un software que permite el desarrollo del Machine Learning, contiene múltiples aplicaciones dentro de su interfaz basadas en distintos algoritmos de entrenamiento. La mejor opción dentro del mantenimiento preventivo es emplear el algoritmo de clasificación, usando una tabla de datos que contiene muestras de vibraciones del motor en buen estado y simulando fallas es posible enseñar a la aplicación a identificar cada caso, permitiendo a futuro usar esta herramienta para conocer el estado real del motor o predecir una falla y programar así una intervención que evite daños mayores.

CAPÍTULO I

1. REVISIÓN BIBLIOGRÁFICA.

1.1 ANTECEDENTES.

El mantenimiento siempre ha sido un tema de importancia en maquinaria, equipos y automóviles, pues con él se busca alargar la vida útil de las piezas y otros componentes que tienden a sufrir desgastes con el paso del tiempo y su uso. Un nuevo paso en el mantenimiento ofrece gestionarlo con la implementación del aprendizaje autónomo, el cual se basa en la condición, esta alternativa mediante la recopilación y análisis de datos, analiza el estado del vehículo y prevé acciones a realizar a futuro si son netamente necesarias, esto reduce costos y en muchos casos mantenimientos innecesarios (Jardine et al., 2006).

Para el desarrollo del modelo del mantenimiento predictivo, el internet de las cosas puede ser de gran ayuda monitoreando en tiempo real el comportamiento de las partes mecánicas del vehículo, (Kanawaday & Sane, 2018) sugieren usar la media integrada autorregresiva para pronosticar los parámetros del maquinaria en estados futuros, usando datos recopilados en el presente. El Machine Learning como herramienta en el mantenimiento tomó impuso a partir del año 2013, pues desde esta fecha se ha apreciado mayor interés por esta área de investigación (Carvalho et al., 2019), las máquinas pueden aprender de los datos obtenidos y reaccionar ante lo aprendido anteriormente y para el área de mantenimiento predictivo se desarrollado 2 tipos de aprendizaje autónomo, el supervisado y el no supervisado (Mathew et al., 2018).

Dentro del Machine Learning (Paolanti et al., 2018) en su investigación han implementado un algoritmo de clasificación vinaria con el cual pueden estimar la probabilidad que el sistema falle en cierto periodo del futuro o de igual forma para calcular la vida útil de ciertos elementos mecánicos. El desarrollo de múltiples algoritmos de aprendizaje se ha usado para la investigación del Machine Learning entre los cuales existen las redes neuronales quienes están interconectadas en una topología de red multicapa conformada por 3 tipos de capas, una de entrada, una o más capas ocultas y una capa de salida (Abbas et al., 2019). Entre los

lenguajes de programación se puede nombrar a python™, MATLAB®, Go y C/C++ como los más usados en aprendizaje autónomo pero en MATLAB® y python™ al ser lenguajes de programación "Plug-and-Play" tienen los algoritmos pre-empaquetados por lo que en su mayoría no requieren procesos de aprendizaje (Gao et al., 2020).

Los datos obtenidos para el aprendizaje autónomo pueden ser generalmente por vibraciones, temperatura o sonido, (Praveenkumar et al., 2014) mediante las vibraciones implemento un algoritmo de aprendizaje el cual permita predecir fallas en cajas de cambios, al realizar su investigación tomó en cuanta algunas variables tanto para el estado de los engranajes como las condiciones en las que las ponía en funcionamiento, usando SVM (Support Vector Machine) consiguió resultados cercanos al 100% de certeza. Cualquier sistema que use el movimiento rotativo para desempeñar su trabajo es posible analizarlo y someterlo al aprendizaje autónomo mediante vibraciones, (Amihai et al., 2018) evaluó el estado de bombas hidráulicas, usando un algoritmo de clasificación encontró variaciones en los datos que pertenecían a turbulencias, cavitaciones, desequilibrios y desalineaciones las cuales el algoritmo pudo identificar sin problema generando un aprendizaje de información que puede ser usada a futuro.

1.2 PLANTEAMIENTO DEL PROBLEMA

El mantenimiento vehicular se lo ha venido realizando únicamente de forma correctiva y preventiva, en el primer caso los vehículos tienden a sufrir fallas mecánicas que impiden su normal funcionamiento e incluso a paradas por largos lapsos de tiempo hasta que se cambie la o las piezas defectuosas, para automotores que realizan actividades comerciales se tendrá también pérdidas financieras considerables. En el caso del mantenimiento preventivo el cual lo recomienda directamente el fabricante, se lo realizará en determinados lapsos de tiempo ya establecidos, por kilometraje u horas de funcionamiento, es decir, se realizan acciones al vehículo sin conocer el estado real de sus componentes, únicamente se toman en cuenta estimaciones de la vida útil de estos (Carvalho et al., 2019).

No todos los vehículos trabajan bajo las mismas condiciones por lo que en algunos habrá desgaste de sus piezas más acelerado que en otros, tomar periodos de funcionamiento para

realizar los mantenimientos es una práctica útil, pero en algunos casos ineficiente ya que existe la posibilidad de remplazar elementos que aun podrían seguir en buen estado, por lo que dichas actividades serian innecesarias para el vehículo en ese momento. Otra posibilidad es que se realice el mantenimiento preventivo cuando los componentes ya no se encuentran en un estado óptimo, por lo que ya no desempeñarían su función correctamente, en este caso las consecuencias podrían ser peores al generar algún daño a los sistemas motrices del vehículo, obteniendo las paradas no programadas. En cualquiera de los dos escenarios se aprecia que el costo final para mantener el vehículo trabajando es más alto, a esto se le adjunta el tiempo y mano de obra invertidos en mantenimientos que aún no eran necesarios.

El mantenimiento dependiendo del estado del vehículo requiere una constante toma de datos para conocer la condición en que se encuentran sus piezas y componentes, dicho gran flujo de datos necesitan ser analizados minuciosamente para determinar patrones o variaciones que indiquen el inicio de alguna falla que se deberá corregir, pero esto sería demasiado trabajo para realizarlo manualmente, además que sería necesario un técnico para cada vehículo generando un costo de mantenimiento excesivamente mayor.

1.3 FORMULACIÓN DEL PROBLEMA

Aplicar aprendizaje autónomo como herramienta en la gestión del mantenimiento vehicular desarrollado con tomas de datos de vibraciones de un tractor agrícola.

1.4 DELIMITACIÓN TEMPORAL Y ESPACIAL

La delimitación temporal permite conocer en qué periodo de tiempo se dio el desarrollo de la investigación, mientras que en la delimitación espacial el enfoque va a los datos y aplicaciones que permite el uso del Machine Learning.

1.4.1 DELIMITACIÓN TEMPORAL

El presente trabajo de investigación se desarrolló en el periodo de tiempo comprendido entre los meses de agosto de 2020 y julio de 2021.

1.4.2 DELIMITACION ESPACIAL

El presente estudio se realizará dentro de la ciudad de Ibarra dentro de los talleres de la Universidad Técnica del Norte.

1.5 OBJETIVOS

1.5.1 OBJETIVO GENREAL

Implementar el aprendizaje autónomo como herramienta en la gestión del mantenimiento predictivo vehicular, para monitorear en tiempo real las condiciones de funcionamiento de un motor diésel en un tractor agrícola.

1.5.2 OBJETIVOS ESPESIFICOS.

- Investigar las aplicaciones y bases de funcionamiento del aprendizaje autónomo en el mantenimiento predictivo vehicular.
- Valorar un software que permita el desarrollo del aprendizaje autónomo enfocado al mantenimiento vehicular.
- Programación del algoritmo de aprendizaje autónomo.
- Evaluar el algoritmo de aprendizaje autónomo haciendo el uso de datos ya obtenidos para valorar su eficacia en la predicción de fallas.

1.6 JUSTIFICACIÓN

Los vehículos automotores constantemente se deben someter a mantenimientos que garanticen su correcto funcionamiento, planificar dichos mantenimientos conlleva la realización de ciertas acciones las cuales pueden ser el remplazo o reajuste de piezas en específico. Para un mantenimiento más eficiente es necesario conocer el estado real del vehículo y sus componentes, principalmente de su motor que es el corazón del vehículo, recopilar datos técnicos puede permitir conocer desde el inicio de una falla hasta predecir cuándo esta ya presentara un problema grave, por lo que la implementación del mantenimiento predictivo es la clave para asegurar una larga vida útil hacia los automotores y de igual forma evitar paradas no programadas las cuales únicamente se reflejan como perdidas productivas y financieras (Siles, 2012).

Un mantenimiento predictivo basado en aprendizaje autónomo se convierte en la mejor alternativa para monitorear y diagnosticar el estado de funcionamiento de un vehículo, de un motor convencional se pude tomar un sinnúmero de datos ya sea por vibraciones, termografía, ultrasonido, entre otros. Pero se vuelven completamente útiles cuando se los somete a un análisis o procesamiento mediante un software especializado, que interprete dichos datos y además sea capaz de predecir futuras anormalidades que se traducen en fallas mecánicas o de funcionamiento (Amihai et al., 2018). El aprendizaje autónomo por medio de un algoritmo de aprendizaje puede realizar este trabajo, por lo que si se lo implementa en el área automotriz se tendrá un mayor control en el mantenimiento vehicular en donde se planificará únicamente acciones necesarias acorde al estado real del vehículo, garantizando un funcionamiento continuo y evitando paradas no programadas o de emergencia (Carvalho et al., 2019).

En el segundo eje de los objetivos nacionales de desarrollo del Plan Nacional del Buen Vivir 2017-2021 el objetivo número 5 dice“ impulsar la productividad y competitividad para el crecimiento sostenible de manera redistributiva y solidaria” acorde a esta investigación se relaciona directamente con la política 5.6 la cual dice “Promover la investigación, la formación, la capacitación, el desarrollo y la transferencia tecnológica, la innovación y el emprendimiento, la protección de la propiedad intelectual, para impulsar el cambio de la matriz productiva mediante la vinculación entre el sector público, productivo y las

universidades", el tema respecto a esta investigación es de carácter innovador en el área automotriz haciendo el uso de la tecnología enfocada al mejoramiento del mantenimiento vehicular (Secretaría Nacional de Planificación y Desarrollo, 2017).

1.7 MARCO TEÓRICO.

1.8 MANTENIMIENTO, GENERAL.

Para garantizar el funcionamiento óptimo de un automotor es necesario darle el mantenimiento adecuado a lo largo de su vida útil, dichos mantenimientos no son más que acciones que garantizan la disponibilidad del vehículo para el uso que este proporcione. El mantenimiento de un automotor se lo realiza normalmente por periodos de tiempo de funcionamiento, en maquinaria se lo realiza tomando en cuenta sus horas de trabajo, mientras que en vehículos convencionales el kilometraje de recorrido.

Un mantenimiento vehicular se enfoca generalmente a su tren motriz, es decir, motor, caja de cambios y transmisión. Entre las principales acciones que contiene un mantenimiento están el remplazo de fluidos, lubricantes, filtros, calibración de piezas y remplazo de elementos de desgaste. Para vehículos con funcionalidades específicas de trabajo como los son la maquinaria pesada o agrícola se realizan acciones extras dependiendo de los mecanismos que estas contengan.

Debido al desgaste o deterioro de elementos en un automotor los mantenimientos son inevitables en la vida útil de un vehículo. Un mantenimiento bien realizado garantiza una inmediata disponibilidad de la maquinaria en cualquier momento, pero un mantenimiento correctamente programado y realizado únicamente cuando sea necesario permite ahorra también recursos económicos.

1.9 DESGASTE DE PIEZAS.

Un motor de combustión interna está conformado por un sinnúmero de piezas móviles las cuales están sometidas constantemente a contacto con otras piezas y por ende a fricción entre ellas, siendo esta la principal causa del desgaste en elementos móviles. El desgaste se define como una progresión de eventos donde interviene deslizamiento y desgaste abrasivo (Santos et al., 2018), de esta forma se produce un desprendimiento de material inevitable conocido como desgaste de piezas.

Las piezas mecánicas de un motor de combustión interna están diseñadas para durar un largo periodo de tiempo trabajando, aun cuando se someten a desgastes controlados por lubricantes la fricción siempre estará presente, por lo que tarde o temprano perderán sus medidas originales, haciendo que su funcionamiento ya no sea el correcto. La fricción y el desgaste de superficies son la principal causa de disipación de energía en motores de automóviles, siendo esta energía desperdiciada en el desprendimiento de material de elementos mecánicos (Ali et al., 2018). Como resultado del desgaste se obtiene partículas por deslizamiento, por desgaste de corte, por desgaste de fatiga, partículas laminares e incluso óxidos rojos y negros, efecto de adhesiones, fatiga, abrasión y corrosión (Zhao et al., 2020).

El desgaste de piezas dentro de un motor es un evento que no se puede evitar, pero si controlar, cuando un elemento mecánico ha culminado su vida útil esta ya no estará en las condiciones adecuadas de funcionamiento. Una pieza desgastada tendrá dimensiones inferiores a las iniciales, teniendo como resultado un mayor juego entre piezas, fácilmente detectable por medio de vibraciones.

1.10 MANTENIMIENTO PREDICTIVO.

Generalmente se realizan 2 tipos de mantenimientos en la mayoría de los automotores, el mantenimiento preventivo el cual se lo realiza bajo recomendaciones de fabricantes por periodos de tiempo ya establecidos, y el mantenimiento correctivo, el cual se lo realiza cuando la falla ya ha pasado y el automotor necesita una reparación inmediata para continuar con su funcionamiento normal. Ambos mantenimientos permiten a la maquinaria continuar

con su funcionamiento, pero cada uno tiene un déficit importante que no lo hace 100 por ciento eficiente, el mantenimiento preventivo se lo realiza sin tomar en cuenta el estado real del automotor, mientras que el mantenimiento correctivo se lo realiza cuando ya el vehículo sufrió una falla y no puede seguir trabajando. El mantenimiento preventivo intenta solucionar los déficits de los 2 mantenimientos anteriormente nombrados, adelantándose a las fallas y ahorrando recursos al programar mantenimientos únicamente cuando la maquinaria lo necesite.

El mantenimiento predictivo es una alternativa viable, el cual se considera como una estrategia que permite optimizar de mejor forma los recursos invertidos en el funcionamiento del automotor. Determinar el momento exacto en el que la maquinaria necesita acciones referentes a mantenimiento es primordial para un mantenimiento preventivo bien ejecutado, una intervención temprana representa desperdicio de vida útil de componentes que se remplazan aun teniendo una funcionalidad aceptable para el vehículo, y una intervención tardía dará como resultado posibles fallas catastróficas que inmovilicen por completo el automotor (Montero Jimenez et al., 2020).

Un mantenimiento predictivo se lo realiza tomando en cuenta datos recopilados del automotor, por lo general de piezas móviles donde mayor desgaste e intervención se necesita. Los datos se toman por sensores colocados estratégicamente y posteriormente se analizan para obtener un pronóstico sobre cuándo se debe remplazar los componentes del vehículo, optimizando así costos de mantenimiento y el tiempo de inactividad (Kimera & Nangolo, 2020).

1.11 VIBRACIONES.

Las vibraciones son oscilaciones de un cuerpo respecto a una línea base, de la cual se toma en referencia cada aspecto de una onda producida. Las vibraciones son productos de alteraciones en el estado de reposo de un cuerpo, en máquinas rotatorias es común que existan desequilibrios en su masa por ende producción de vibraciones.

Existen distintos tipos de vibraciones clasificadas por su frecuencia, de muy baja frecuencia si son inferiores a 1 Hz, de baja frecuencia si se encuentran entre un rango de 1 a 20 Hz y de alta frecuencia si superan los 20 Hz hasta los 1000 Hz, por lo general las de baja frecuencia son las producidas por automotores, desde automóviles hasta maquinaria pesada.

Un motor de combustión interna por su mecanismo de funcionamiento va a producir vibraciones apreciables y fáciles de detectar, el ciclo de un motor ya se otto o diésel tendrá un tiempo de trabajo, siendo en esta etapa la responsable de crear una perturbación en la masa del motor conocida como vibración. La variación de presión en la combustión y golpes son la principal fuente de ruidos y vibración en un motor, en motores otto el frente de llama producido por una chispa libera energía rápidamente, mientras que en motores diésel la ignición empieza con la inyección de combustible la cual se enciende con un retardo, dichos eventos provocan un rápido e irregular aumento de presión del gas, lo que genera golpes en cada cilindro del motor (Taghizadeh-Alisaraei & Mahdavian, 2019).

Por el tipo de funcionamiento las vibraciones son inevitables dentro de un motor de combustión interna, se producen en todos las direcciones variando según el número de pistones, configuración de estos y la masa de equilibrio, además de él orden de encendido y las condiciones en las que se encuentre, como régimen, carga y sincronización del motor (Farag et al., 2017).

1.11.1 UNIDADES DE MEDIDA.

Las vibraciones son de diferentes tipos y formas por lo que generalizarlas mediante medidas estandarizadas es la forma más adecuada para clasificarlas según su tipo. Las medidas generales que se usan dentro de las vibraciones se enfocan a la forma de onda que estas produzcan.

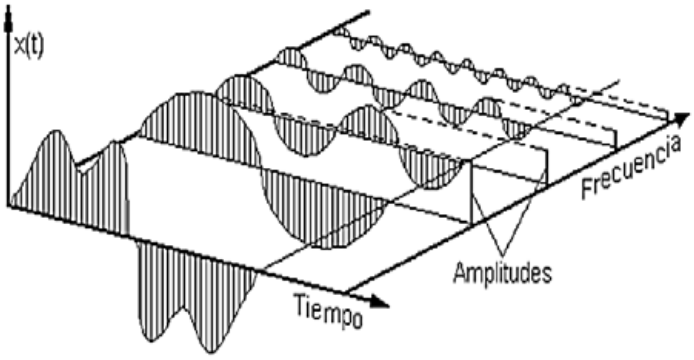


Figura 1.1. Partes de una vibración, tiempo, amplitud y frecuencia.
(Batista et al. 2011, pág. 23)

Magnitud.

La magnitud de una vibración se mide normalmente como una aceleración constante que tiene el objeto al acelerar hacia un lado y luego hacia otro, entonces la magnitud se puede expresar en función al desplazamiento en ambos sentidos (de pico a pico). La magnitud para fines más prácticos se expresa como la distancia que se tiene entre el pico más alto contra el pico más bajo de la oscilación.

Frecuencia.

Conociendo a la magnitud como el desplazamiento de una oscilación entre un punto máximo y mínimo, la frecuencia es la cantidad de veces que se produce una oscilación completa dentro de un segundo, es decir, una oscilación por segundo equivale a 1Hz.

Dirección.

La dirección de una vibración se produce respectivamente a la fuerza que ocasionó el movimiento inicial, se puede producir dentro de los 3 ejes conocidos o también en dirección rotacional.

Duración.

La duración de una vibración depende directamente de la magnitud de la fuerza inicial que ocasionó el movimiento vibracional, la duración de una vibración se mide fácilmente en segundos y se prolongará si el aporte de movimiento continúa.

1.11.2 TIPOS DE VIBRACIONES.

Existen diferentes tipos de vibraciones las cuales se caracterizan por su naturaleza única al producirse y por las características individuales que posee. Las vibraciones cambian al tener diferentes tipos de oscilaciones, o al moverse dentro de un punto de equilibrio, también se toma en cuenta donde se mueven y los grados de libertad que estas tienen.

Vibración libre.

Una vibración libre se produce cuando una masa u objeto se desplaza cierta distancia y sin restricciones se mueve libremente sobre su punto de equilibrio.

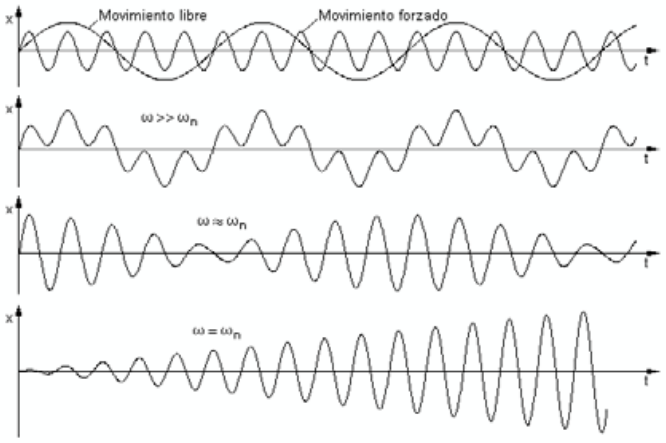


Figura 1.2. Vibraciones forzadas amortiguadas y no amortiguadas.
(Batista et al. 2011, pág. 45)

Vibración libre no amortiguada.

Las vibraciones libres no amortiguadas se caracterizan principalmente porque las ondas que producen se propagan de forma que la amplitud no se disminuye con el paso del tiempo, al contrario, se mantiene constante a lo largo de toda la vibración.

Vibración libre amortiguada.

El caso contrario al tipo de vibración anterior, ya que aquí la onda oscilatoria si disminuye progresivamente debido a la disipación de energía, la cual se produce dentro de un punto de equilibrio del cuerpo. El amortiguamiento se conoce como una fuerza negativa a la fuerza inicial que produjo el primer movimiento, dicha fuerza es la responsable de la velocidad de amortiguamiento de la vibración.

Vibración Forzada.

Una vibración forzada se produce siempre que se aplique una fuerza periódica a una masa en estado de equilibrio, la velocidad angular de la fuerza aplicada en la frecuencia y la amplitud de la oscilación depende directamente de la magnitud.

Vibración forzada no amortiguada.

Una vibración forzada no amortiguada se produce si no existe una disipación de energía posteriormente a la aplicación de la fuerza, por este motivo no se espera una cambio dentro de la vibración producida.

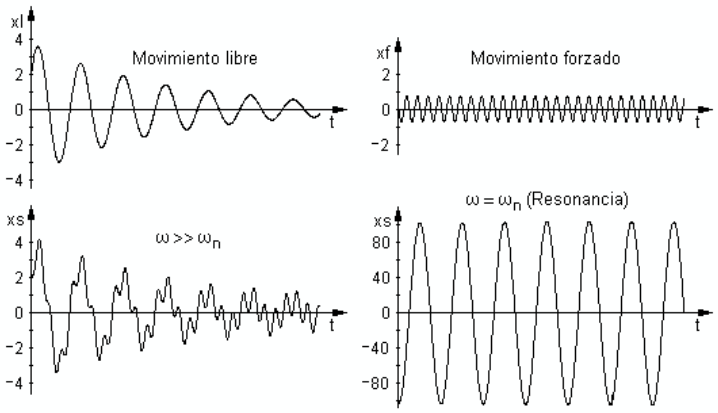


Figura 1.3. Vibraciones libres y forzadas, amortiguadas y no amortiguadas.
(Batista et al. 2011, pág. 50)

Vibración forzada amortiguada.

Es el producto de una vibración forzada pero aplicada una disipación de energía que progresivamente ira disminuyendo la oscilación producida por la vibración, tras un periodo de tiempo la vibración se habrá detenido por completo.

Vibración aleatoria.

Una vibración aleatoria se produce cuando la fuerza que mueve la masa dentro de su punto de equilibrio no es constante, produce una oscilación con picos diferentes en cada oscilación, teniendo una onda asimétrica.

1.12 APRENDIZAJE AUTÓNOMO.

El Machine Learning o aprendizaje autónomo es una rama de la inteligencia artificial, en la década de 1950 el matemático británico Alan Turing cuestionó "¿pueden pensar las máquinas?" (Turing, 1950), para que una máquina sea inteligente necesita aprender por experiencia, experiencias a las que la máquina sea sometida (Anastasi et al., 2021).

El Machine Learning actualmente se ha desarrollado de tal forma que no necesita una programación predeterminada para generar el aprendizaje, se han empleado algoritmos capaces de generar conocimiento usando una gran base de datos como fuente de alimentación inicial (Sandoval, 2018). Es posible emplear el aprendizaje autónomo en indistintos campos, ya que juega un papel importante en el procesamiento de grandes cantidades de datos permite ahorrar tiempo y maximizar los recursos informáticos (Xu et al., 2021).

El aprendizaje autónomo es un método que permite usar dispositivos con capacidad computacional los cuales puedan extraer relaciones y patrones en un conjunto de datos, dichos patrones y relaciones se puede usar posteriormente en toma de decisiones o en la predicción de comportamientos (Ramiro & Alvarado, 2020).

Existen distintos tipos de algoritmos que permiten desarrollar el aprendizaje autónomo, pero inicialmente 2 únicos tipos de aprendizajes, el supervisado y el no supervisado, variando el uno del otro únicamente por la intervención humana en cada proceso.

1.12.1 APRENDIZAJE AUTÓNOMO SUPERVISADO.

El aprendizaje autónomo supervisado no es más que la intervención humana en todo el proceso de entrenamiento del algoritmo. En este método se usa datos con características y etiquetas ya establecidos por el entrenador para el desarrollo del aprendizaje por el algoritmo y de esta forma sepa cómo actuar al entregarle datos en el futuro (Sandoval, 2018).

El proceso de entrenamiento de un algoritmo de aprendizaje autónomo supervisado es básicamente la recopilación de datos, darles características y etiquetarlos ya sea de forma manual o semiautomática, posteriormente se introduce los datos a un algoritmo de aprendizaje autónomo donde este sea capaz de encontrar algún patrón o similitud entre el conjunto de datos, finalmente se introduce nuevos datos para que el algoritmo los procese y presente resultados que puedan ser tomados en cuenta al realizar predicciones o en la toma de decisiones (Brusco, 2017).

El aprendizaje autónomo supervisado permite desarrollar 2 diferentes algoritmos, el de clasificación y el de regresión.

El algoritmo de clasificación usa como referencia las características que poseen el grupo de datos y por medio de sus etiquetas es capaz de clasificarlos. El nuevo grupo de datos no tiene que poseer características exactamente similares al usado en el periodo de aprendizaje, el algoritmo es capaz de encontrar similitudes en los nuevos datos y clasificarlos al grupo que más se asemejen. Las variables otorgadas por clasificación pueden ser binarias (únicamente 2 opciones de clasificación), múltiples (múltiples opciones de clasificación) y ordenadas (por nivel o numéricamente)(Sandoval, 2018).

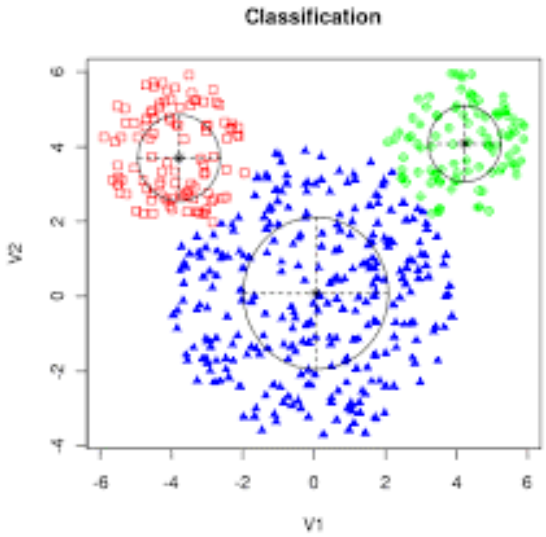


Figura 1.4. Diagrama de dispersión perteneciente a un algoritmo de clasificación.
(Sandoval Serrano, 2018, pág. 37)

El método de regresión usa datos que de igual forma necesitan poseer características y etiquetas para el aprendizaje inicial, posteriormente el algoritmo será capaz de ubicar al nuevo grupo de datos en un lugar en específico en la gráfica, el resultado en la predicción de este método es un valor numérico dentro de una línea recta.

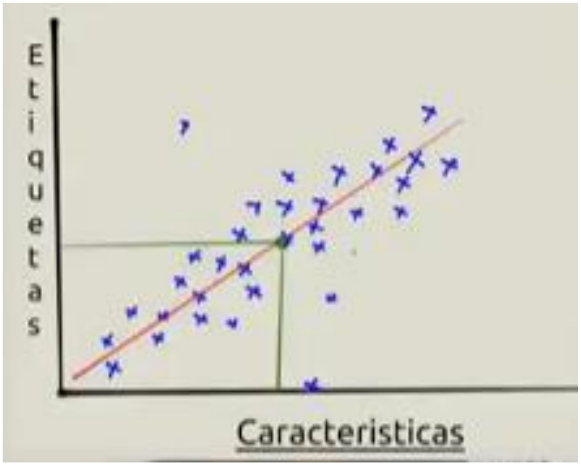


Figura 1.5. Diagrama de dispersión perteneciente a un algoritmo de regresión lineal. (Sandoval Serrano, 2018, pág. 37)

1.12.2 APRENDIZAJE AUTÓNOMO NO SUPERVISADO.

El aprendizaje autónomo no supervisado requiere la intervención humana únicamente en la proporción de datos iniciales, solo con características, más no etiquetas. El algoritmo en este caso intenta relacionar los datos proporcionados buscando patrones y características que comparta entre sí y de esta forma agruparlos.

1.13 ALGORITMOS DE APRENDIZAJE.

Desde el inicio del Machine Learning se han creado diferentes algoritmos que permiten la realización del aprendizaje autónomo, dependiendo de su uso existen algoritmos con distintas características funcionales.

1.13.1 ALGORITMOS DE REGRESIÓN.

Los algoritmos basados en regresión, ya sea lineal o logística, usan conjuntos de datos como puntos dentro de un mismo plano, se ajusta una línea recta a través del conjunto de datos la cual sirve como referencia en el aprendizaje inicial o entrenamiento del algoritmo. El algoritmo de regresión determina una variable dependiente respecto a sus variables

independientes, es decir según sus características el algoritmo le otorgará un lugar dentro de la recta. Es un algoritmo muy útil en predicción y pronostico.

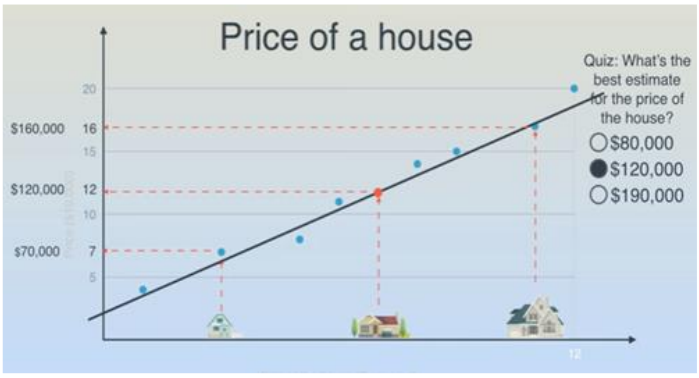


Figura 1.6. Ejemplo de uso de algoritmo de regresión.

(Sandoval Serrano, 2018, pág. 38)

1.13.2 ALGORITMOS DE ÁRBOL DE DECISIÓN.

Un árbol de decisión es un algoritmo de aprendizaje muy útil con estructura similar a un diagrama de flujo. Mediante métodos de bifurcación el algoritmo usa los atributos o características de los datos proporcionados y modela posibles caminos a tomar y los eventos que podrían ocurrir. Este modelo permite llegar a conclusiones lógicas ilustrando cada resultado posible de una decisión, cada nodo que conforma el árbol representa una variable específica, mientras que cada rama se representa como efecto de esa variable.

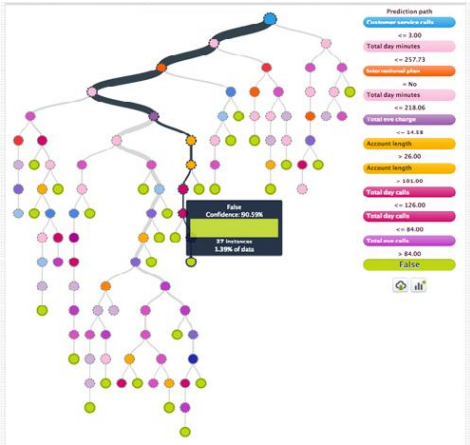


Figura 1.7. Gráfico modelo de árbol.

(Sandoval Serrano, 2018, pág. 38)

1.13.3 ALGORITMOS BAYESIANOS.

Es un algoritmo basado en el teorema de Bayes, útil tanto para regresión como clasificación. El aprendizaje se realiza de tal forma que la clasificación de datos se hace con cada valor independiente de cualquier otro, esto permite una predicción en función de las características de los datos usando modelos de probabilidad.

1.13.4 ALGORITMOS DE CLUSTERING (AGRUPACIÓN).

El algoritmo de agrupación es un tipo de aprendizaje autónomo no supervisado, básicamente aquí se proporcionan los datos sin etiquetas que los identifiquen, únicamente características. Los algoritmos de agrupación permiten ordenar datos en grupos indefinidos, se realiza una búsqueda dentro del conjunto de datos proporcionado donde el algoritmo establece una variable de representación a cada grupo encontrado, posteriormente según las características de cada dato se le asigna dicha variable.

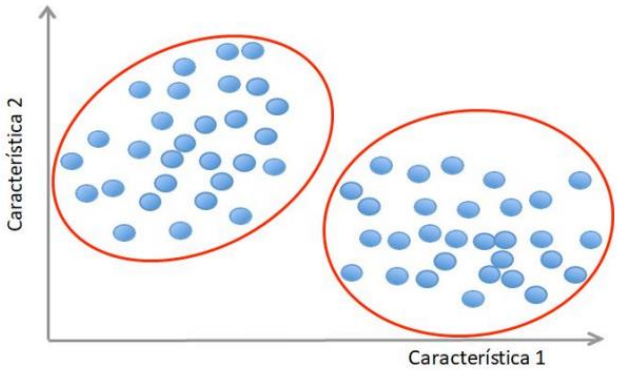


Figura 1.8. Diagrama de dispersión de un algoritmo de clasificación.
(Ramiro & Alvarado, 2020, pág. 33)

1.13.5 ALGORITMOS DE REDES NEURONALES.

Se han querido implementar desde ya hace muchos años, tratando de simular una red neuronal biológica las computadoras no eran lo suficientemente aptas para su uso, pero desde hace pocos años se ha retomado la idea. Su función es similar a una red neuronal real, posee

capas encargadas de procesar la información y dichas capas están interconectadas entre sí. Son requeridas en aprendizajes profundos ya que aquí la información a procesar es difícil de comprender, es un algoritmo muy bueno detectando patrones dentro de conjunto de datos.

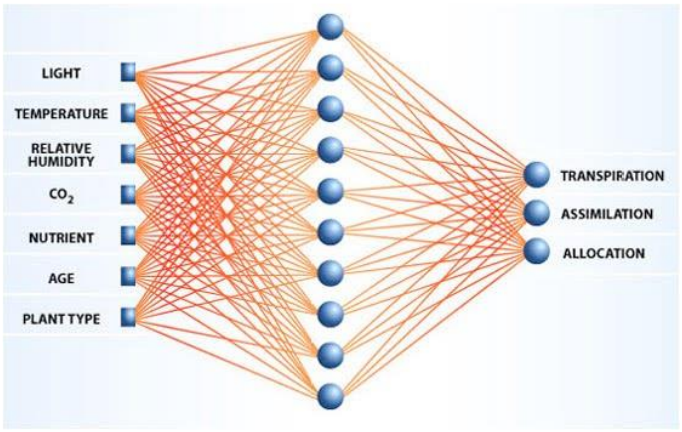


Figura 1.9. Grafica de redes neuronales.
(Sandoval Serrano, 2018, pág. 38)

1.13.6 ALGORITMOS DE APRENDIZAJE PROFUNDO.

Son la evolución de las redes neuronales, ya que aprovechan al máximo este algoritmo, son capaces de procesar información capa tras capa simplificándola de tal forma que sea completamente comprensible para el algoritmo de aprendizaje. En este caso el tipo de aprendizaje es progresivo, teniendo niveles bajos de aprendizaje en las capas primarias y tras cada capa subir el nivel.

Los datos procesados mediante el aprendizaje profundo son generalmente de gran complejidad o con cientos de características, un claro ejemplo son las imágenes, las cuales en un algoritmo convencional no podrían ser entendidas en su totalidad, por lo que el aprendizaje no sería el correcto. El algoritmo de aprendizaje profundo permite el procesamiento de datos de forma progresiva y es capaz de adquirir conocimiento mediante la experiencia.

1.13.7 ALGORITMOS DE REDUCCIÓN DIMENSIONAL.

Es un algoritmo de aprendizaje no supervisado, tiene la cualidad de simplificar las características de los datos otorgados, reduciendo así las variables que se debe considerar para la obtención de resultados. Este algoritmo es usado usualmente para mejorar el proceso de entrenamiento en algoritmos de aprendizaje autónomo supervisado, ya que depura los datos iniciales otorgando una muestra de mejor calidad.

1.14 SOFTWARE QUE PERMITE EL DESARROLLO DEL APRENDIZAJE AUTÓNOMO.

Existen múltiples softwares que permiten el uso de Machine Learning y sus aplicaciones, por lo que no es necesario crear algoritmos de aprendizaje desde cero. Por lo general se emplean programas que permiten el uso de aprendizaje autónomo dentro de la clasificación, regresión y agrupación, dentro de todos estos casos es necesario tener una base de datos y únicamente proporcionarla al software para que este genere el aprendizaje automáticamente. Entre los softwares más usados para la ejecución de Machine Learning están:

1.14.1 MACHINE LEARNING EN python™.

El lenguaje de programación en python™ es uno de los más populares gracias a su facilidad en ser legible y al ser un lenguaje de programación multiparadigma permite soportar diferentes orientaciones, lo que favorece usar diferentes estilos.

Para el desarrollo de Machine Learning existen diferentes librerías, cada una presenta características diferentes tanto para el aprendizaje autónomo como la IA (Inteligencia Artificial). Librerías como Scikit permite desarrollar clasificación, regresión y agrupación mediante un software libre y de origen en programación de python™ y opciones más avanzadas como Open CV con funciones más avanzadas en el reconocimiento de objetos.

1.14.2 MACHINE LEARNING EN MATLAB®.

MATLAB® es un software de programación el cual permite por una de sus extensiones generar aprendizaje autónomo con algoritmos ya pre empaquetados a los cuales se los puede usar directamente. Permite el desarrollo de aprendizaje supervisado en clasificación y regresión, y no supervisado en la agrupación de datos.

MATLAB® no solo es capaz de generar aprendizaje autónomo para predicciones futuras, sino también generar funciones de entrenamiento para usar nuevos datos.

1.15 MATRIZ DE DECISIÓN.

Una matriz de decisión es una herramienta gráfica útil usada para de una forma racional y lógica tomar las mejores decisiones respecto a un tema a analizar. El correcto funcionamiento de una matriz de decisión o árbol de decisión consiste en tomar en cuenta opciones posibles al realizar cierta acción, analizarlas y determinar cuál es más beneficiosa para quienes toman la decisión.

1.16 MATLAB® Y HERRAMIENTAS DE MACHINE LEARNING.

MATLAB® es un software multifuncional con una extensión dedicada únicamente al desarrollo de aprendizaje autónomo, aquí existen opciones útiles que se puede usar en el desarrollo de Machine Learning.

En el apartado de opciones de Machine Learning:

1.16.1 CLASSIFICATION LEARNER.

Esta aplicación de Machine Learning permite la clasificación de datos por medio de un aprendizaje autónomo supervisado. El entrenamiento de datos se lo puede realizar usando diferentes modelos de clasificación, árboles de decisión, análisis discriminante, regresión

logística, vecinos más cercanos, Bayes ingenuo, clasificación de conjuntos y redes neuronales (MathWorks & MATLAB, 2020g).

El conjunto de datos proporcionado para el entrenamiento debe constar de características y etiquetas para que el modelo los pueda clasificar según la información entrante de cada grupo.

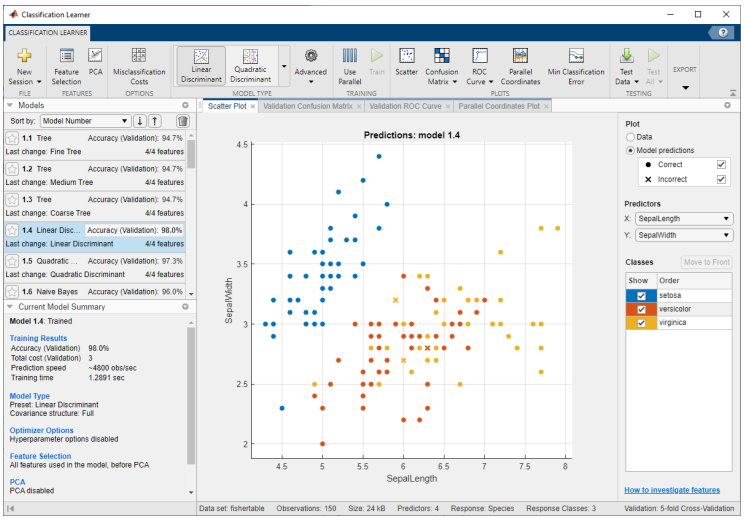


Figura 1.10. Ventana de Classification Learner, aplicación de MATLAB ® (MathWorks & MATLAB, 2020g).

1.16.2 DEEP NETWORK DESIGNER.

Es un método de entrenamiento usado generalmente para el aprendizaje de imágenes, trabaja por un método de aprendizaje conocido como aprendizaje de transferencia, por lo que es un algoritmo que se puede entrenar más rápido y fácil que una red neuronal (MathWorks & MATLAB, 2020d).

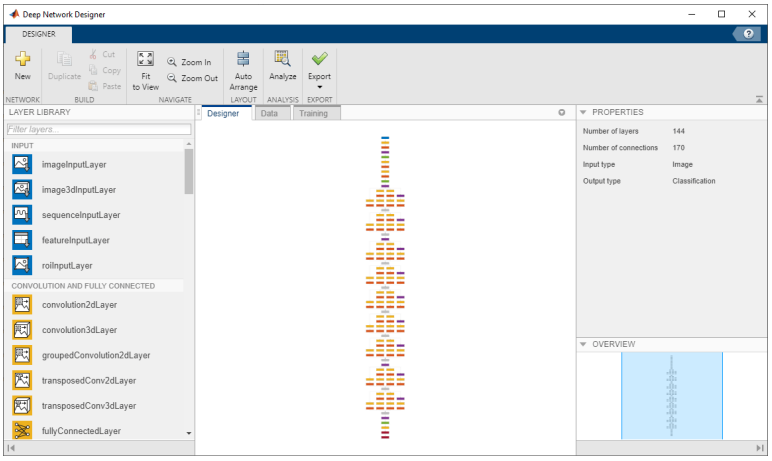


Figura 1.11. Ventana de Deep Network Designer, aplicación de Machine Learning de MATLAB®

(MathWorks & MATLAB, 2020d).

1.16.3 EXPERIMENT MANAGER.

Permite crear experimentos de aprendizaje profundo y así entrenar diversas redes neuronales, se puede comenzar rápidamente usando plantillas ya pre configuradas que admiten flujos de datos en clasificación y regresión de imágenes, entrenamientos personalizados, clasificación de algunas secuencias y segmentación semántica (MathWorks & MATLAB, 2020b).

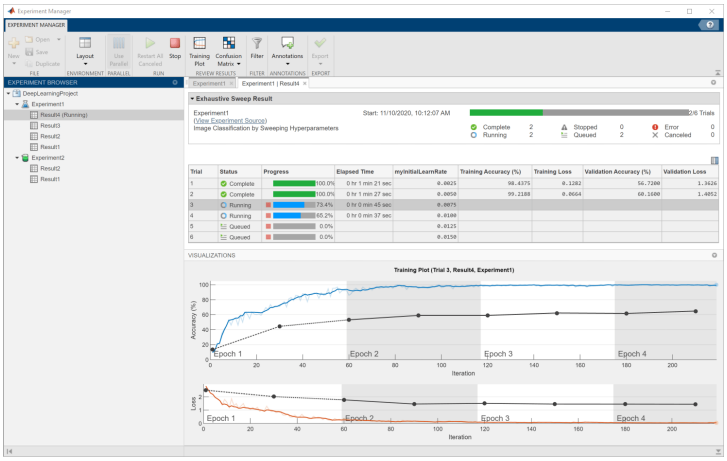


Figura 1.12. Ventana de Experiment Manager, aplicación de Machine Learning de MATLAB®

(MathWorks & MATLAB, 2020b).

1.16.4 NEURAL NET CLUSTERING.

Es una aplicación que permite trabajar con problemas de agrupación al utilizar mapas autoorganizados, mediante este método MATLAB® ayuda tanto en la arquitectura como en el entrenamiento de la red y al finalizar el proceso es posible evaluar los resultados por medio de herramientas visuales, rendimiento de la red en equipos de prueba u ofrecer varias opciones en la red entrenada (MathWorks & MATLAB, 2020a).

1.16.5 NEURAL NET FITTING.

Es una aplicación que permite la resolución de problemas de ajustes de datos usando principalmente una red directa de 2 capas como red de alimentación, aquí MATLAB® ayuda en el proceso total de entrenamiento empezando por la selección de datos, conjuntos de entrenamientos, eficiencia y pruebas del entrenamiento. La evaluación de resultados se la puede apreciar observando su rendimiento, usando error cuadrático y análisis de regresión (MathWorks & MATLAB, 2020c).

1.16.6 NEURAL NET PATTERN RECOGNITION.

Permite la resolución de problemas de clasificación de reconocimiento de patrones usando una red de alimentación directa de 2 capas con neuronas de salida sigmoidea.

1.16.7 NEURAL NET TIME SERIES.

La aplicación permite la resolución de problemas de 3 diferentes tipos de series no lineales, al igual que otras aplicaciones permite la asistencia de MATLAB® en la arquitectura de la red y el desarrollo tanto del aprendizaje como el de evaluación de resultados (MathWorks & MATLAB, 2020f).

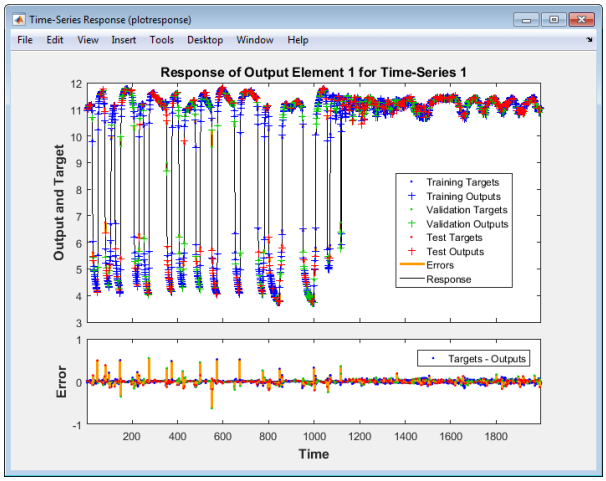


Figura 1.13. Ventana de Neural Net Time Series, aplicación de Machine Learning de MATLAB®

(MathWorks & MATLAB, 2020f).

1.16.8 REGRESSION LEARNER.

Esta aplicación permite entrenar varios modelos de regresión y validar sus resultados comparando errores de validación en paralelo para así elegir la mejor opción con menor error (MathWorks & MATLAB, 2020e).

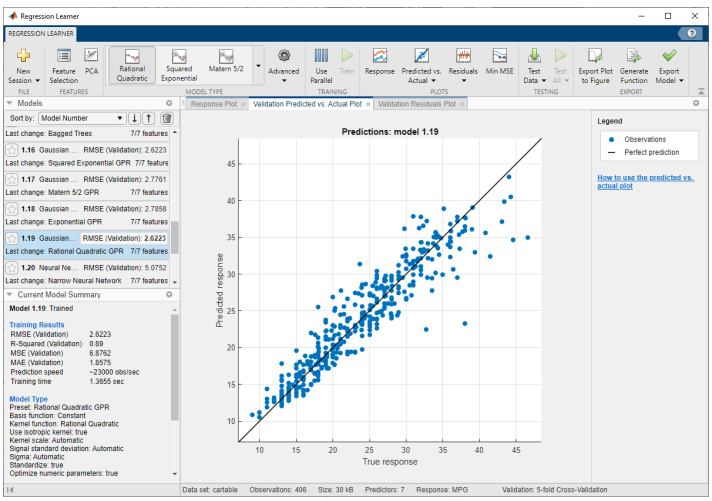


Figura 1.14. Ventana de Regression Learner, aplicación de Machine Learning de MATLAB®

(MathWorks & MATLAB, 2020e).

1.17 ADECUACIÓN DEL ALGORITMO DE APRENDIZAJE AUTÓNOMO ENFOCADO AL MANTENIMIENTO PREDICTIVO.

En MATLAB® la app de Machine Learning es capaz de generar aprendizaje autónomo tanto supervisado como no supervisado, pues al ofrecer esta opción pre empaquetada es posible usar directamente el algoritmo de aprendizaje siempre y cuando se tenga datos suficientes y de una buena calidad.

El algoritmo de aprendizaje necesita ser proporcionado por una buena cantidad de datos para el aprendizaje inicial y posteriormente con el mismo tipo de datos realizar predicciones o clasificaciones. Orientado al mantenimiento predictivo el algoritmo se debe adecuar de tal forma que los datos ingresados correspondan característicamente al estado del automotor ya sea general o de piezas específicas, procesar estos datos mediante programación de MATLAB® al mismo tiempo que realiza predicciones de estado del automotor.

El mantenimiento predictivo es capaz de generarse gracias a algoritmos de aprendizaje, en este caso MATLAB®, ya que permite mediante datos del automotor conocer su estado real y realizar la programación adecuada en cuanto a mantenimientos del vehículo.

1.18 EJECUCIÓN DEL ALGORITMO DE APRENDIZAJE AUTÓNOMO PARA EVALUAR LA CAPACIDAD DE PREDICCIÓN DE FALLAS.

Al generar el aprendizaje en la extensión de Machine Learning de MATLAB® se genera una función de predicción la cual se puede ejecutar con nuevos datos similares a los usados para el entrenamiento, al realizar dicho entrenamiento MATLAB® proporciona información de la eficiencia del aprendizaje, al igual que una matriz de confusión.

La eficiencia del algoritmo de aprendizaje indica la cantidad de muestras de datos que tuvo problemas en clasificar y se encontraban en un punto medio de las 2 variables. Es decir, un 90 por ciento de eficiencia dice que 1 de cada 10 grupos de datos no fue clasificado correctamente lo cual se verificará al realizar pruebas progresivamente con nuevos datos.

CAPÍTULO II

2.MATERIALES Y MÉTODOS.

INTRODUCCIÓN

El mantenimiento predictivo actualmente se está convirtiendo en una alternativa mejorada respecto al mantenimiento preventivo, no solo permite el diagnóstico según el estado del vehículo sino también predecir fallas y tomar acciones antes de que sucedan. Existen múltiples formas de gestionar el mantenimiento predictivo, esta investigación se enfoca en una de ellas y la responsable de la ejecución de este trabajo, el aprendizaje autónomo. El aprendizaje autónomo o Machine Learning (ML), es una herramienta capaz de ser usada en múltiples tareas de clasificación, regresión o agrupación, útiles en procesamiento de datos y aplicaciones de predicción.

Un mantenimiento predictivo necesita inicialmente una gran cantidad de datos capaces de otorgar información necesaria que será usada para predecir el comportamiento de la máquina y así adelantarse a sus futuras fallas, en este capítulo se explica cómo vibraciones adquiridas de un tractor agrícola son procesadas y usadas en la ejecución del Machine Learning. Si bien existen múltiples softwares que permiten el aprendizaje autónomo, MATLAB® ha sido la mejor opción en cuanto a su fácil entrenamiento y posterior manejo en la realización de predicciones. De igual forma el programa permite el proceso completo tanto en procesamiento de datos, entrenamiento del algoritmo de clasificación y respectivo uso en predicciones.

2.1 MATERIALES.

2.1.1 TRACTOR AGRÍCOLA

El automotor seleccionado para este estudio es un tractor agrícola propiedad de la Universidad Técnica del Norte, producto de una tesis anterior de la Carrera de Ingeniería en Mantenimiento Automotriz. El automotor se encuentra en óptimas condiciones por lo que es adecuado para las pruebas necesarias de esta investigación.

El vehículo es un tractor agrícola de la marca Internacional Harvester serie 523, propulsado por un motor de ciclo diésel, su estado y funcionalidad son los esperados para un automotor de este tipo. Es capaz de accionar múltiples componentes adicionales a través de su toma fuerza y sistema hidráulico por lo que es una maquinaria completamente adecuada para ser usada en trabajos de agricultura sin ningún problema.

Dentro de algunas de sus características están, un motor de 3 cilindros en línea con un desplazamiento de 2,9 litros, capaz de otorgar 52 hp a 2100 rpm, acotado una transmisión de 8 velocidades hacia adelante y 4 en reversa. Consta también de un sistema hidráulico a 165 Bar con un caudal de 7.5 galones por minuto, suficientes para el accionamiento de sus cilindros hidráulicos.

A pesar de que el tractor agrícola funciona correctamente en todos sus componentes y diferentes sistemas, para el desarrollo de esta investigación únicamente se tomara en cuenta al motor, ya que es de aquí de donde se obtendrá los datos de vibraciones para el entrenamiento y validación del algoritmo de aprendizaje autónomo.

Especificaciones técnicas del motor.

Al enfocarnos principalmente en el motor, se ha tomado en cuenta sus características más importantes sobre su funcionamiento y constitución, la tabla 2.1 especifica los datos más relevantes.

Tabla 2.1. Datos del motor del tractor agrícola.

Motor IH 523	
Número de cilindros	3
Cilindrada	2.9 L
Diámetro de cilindros	98 mm
Carrera del pistón	129 mm
Aspiración	Atmosférica.
Potencia	52 hp a 2100 rpm
Relación de compresión.	15:1
Suministro de combustible.	Inyección directa.
Orden de encendido	1 - 3 - 2
Refrigeración	Liquida.
Capacidad de lubricante	6.4 L

Fuente. (TractorData, 2016)

Especificaciones generales del tractor.

La tabla 2.2 muestra las especificaciones generales sobre el tractor agrícola, su chasis, dimensiones, sistema hidráulico, entre otros.

Tabla 2.2. Datos generales del tractor agrícola.

Producción:	
Fabricante:	Cosechadora Internacional
Fábrica:	Neuss, Alemania
Capacidad:	
Combustible:	18,5 galones (70L).
Sistema Hidráulico:	4,3 galones (16,3L).
Enganche de 3 puntos:	
Tipo trasero:	II / I
Elevador trasero:	3 748 libras (1700kg)
Toma de Fuerza (PTO):	
TDF trasera:	Independiente.
RPM Traseras:	540 – 540/1000
Dimensiones y Neumáticos:	
Distancia entre ejes:	78,74 pulgadas (199 cm).
Peso:	5380 a 6295 libras.
Neumático delantero:	6.00-16
Llanta Trasera:	11x32

Tabla 2.2. Datos generales del tractor agrícola **(Continuación).**

Hidráulica:	
Capacidad:	4,3 galones (16,3L).
Caudal de la Bomba:	7,5 gal/min (28,4 L/min).
Eléctrico:	
Cargando sistema:	Generador
Amperios de carga:	10.8
Batería:	135 amperios hora.
Mecánico:	
Chasis:	4x2 2WD – 4x4 TDM 4WD
Bloqueo diferencial:	Trasero mecánico.
Frenos:	Dto
Dirección:	Manual

Fuente. (TractorData, 2016)

2.1.2 SENSOR DE VIBRACIONES.

Dentro de un motor de combustión interna los sensores piezoeléctricos han sido usados en distintos monitoreos de diferentes partes del motor, poseen una gran precisión y trabajan bajo una frecuencia natural con un alto rango en la obtención de datos.

Un sensor de mayor precisión, el acelerómetro piezoeléctrico 603C01, puede monitorear vibraciones en el motor independientemente de si este se encuentre estacionado o en movimiento. El funcionamiento del sensor es sencillo, gracias a sus cualidades piezoeléctricas puede generar tensión por factores externos como deformación o presión, el caso de este elemento en particular es accionado por una señal de voltaje y responde con puntos de tensión a lo largo de la vibración captada.

El sensor posee características según su construcción y funcionamiento, trabaja bajo la tecnología ICP, específico para máquinas rotativas y a frecuencias de 0.5 kHz y 10 kHz.

Figura 2.1. Sensor de vibraciones.

2.1.3 TARJETA DE ADQUISICIÓN DE DATOS

La tarjeta de adquisición de datos es un módulo de entrada de sonido de 2 canales a una tasa de actualización de 102.4 kS/s cada canal simultáneamente, trabaja a una tensión de 5V y es ideal para medir señales desde sensores piezoeléctricos electrónicos integrados, al igual que acelerómetros. La tarjeta de adquisición de datos posee filtros anti-aliasing en cada canal, esto permite ajustar automáticamente la velocidad de muestreo de datos.

Figura 2.2. Tarjeta de adquisición de datos.

2.2 SOFTWARE.

2.2.1 LabVIEW.

El software Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) es un sistema de programación gráfica en el cual se puede usar en adquisición de datos, análisis y control de ellos (Roncancio & Cifuentes, 2000). Fue creado en 1976 por una empresa llamada National Instruments, pero fue lanzado hasta el año 1986.

LabVIEW, se desarrolló de tal forma que permita la creación de aplicaciones por medio de lenguaje de programación similar al del lenguaje C o BASIC. Una de las cualidades que destaca el programa es su facilidad de uso, su interfaz es muy intuitiva por lo que no se requiere programadores expertos para escribir un programa. LabVIEW ha sido y es un programa usado frecuentemente por ingenieros en el manejo de datos, frecuentemente se usa para comunicar un computador con equipos o circuitos externos, siendo así una opción ideal para proyectos grandes donde sea necesario conocimientos de electrónica, mecánica, robótica y programación generalmente (Masterhacks.net, 2013).

2.2.2 MATLAB®.

MATLAB® es un programa ideal para trabajar con vectores y matrices, además que también se pueden introducir cadenas de caracteres y estructuras de información. El programa no solo permite un alto rendimiento en cálculos técnicos, permite también un entorno de programación, así es posible crear herramientas de trabajo según las tareas que se desee realizar (Servicio Informatico de Sistemas, 2020).

El código MATLAB® permite fácilmente crear programación propia o funciones de trabajo, el programa entre sus funciones permite el cálculo matricial, cálculo de álgebra lineal, manejar polinomios, funciones, ecuaciones diferenciales, gráficos, entre otras.

2.3 MÉTODOS.

El software empleado, MATLAB®, tiene una infinidad de ocupaciones con las cuales es posible realizar trabajos con grandes cantidades de datos y además actualmente consta con una aplicación de Machine Learning ya pre-empaquetada en su sistema, la cual ahorra la elaboración de un algoritmo de aprendizaje autónomo y permitiendo realizar el entrenamiento de forma directa.

En la aplicación de Machine Learning que MATLAB® ofrece hay 8 distintas opciones, como muestra la figura 2.3, de las cuales para el fin de esta investigación únicamente se usara una, la de Classification Learner. Esta opción es completamente automatizada, ya que permite realizar el entrenamiento de forma directa y no requiere una programación previa.

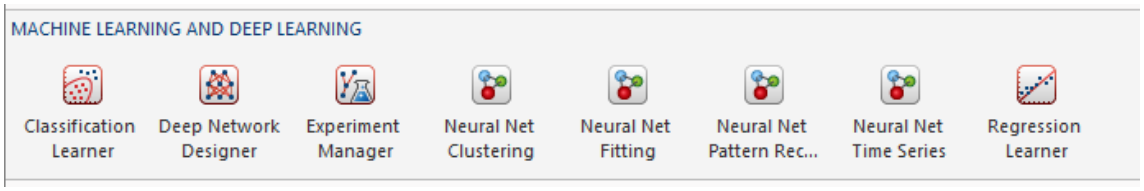


Figura 2.3. Opciones de Machine Learning y Deep Learning dentro de MATLAB® (Matlab®, 2020)

Al iniciar la aplicación de clasificación esta permite usar la tabla de datos para el entrenamiento ya sea del Workspeace o File.

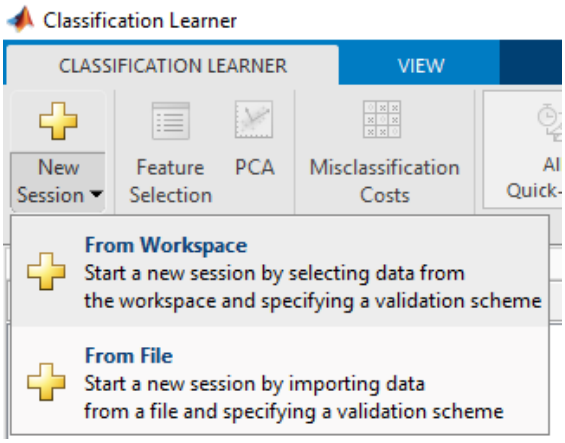


Figura 2.4. Opciones de inicio de sesión para Classification Learner. (Matlab®, 2020)

En esta ventanilla (Figura 2.5) se puede verificar si las características y etiquetas del conjunto de datos son las correctas para iniciar el aprendizaje, si no lo son, se debe seleccionar manualmente. También se puede seleccionar entre 3 opciones de validación del aprendizaje autónomo.

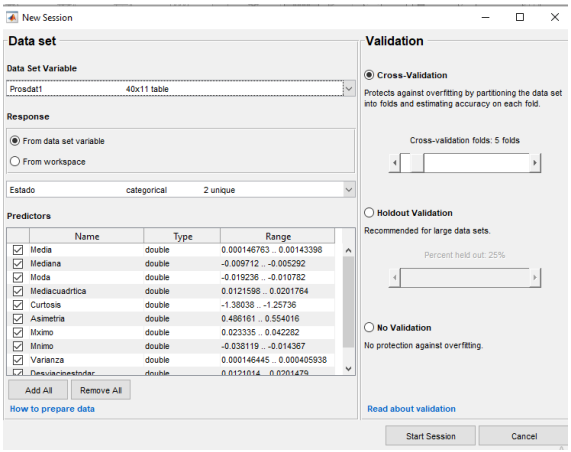


Figura 2.5. Ventana de inicio de sesión de Classification Learner.
(Matlab®, 2020)

Ya iniciado sesión se abre la ventana mostrada en la figura 2.6 donde hay diferentes opciones para el entrenamiento.

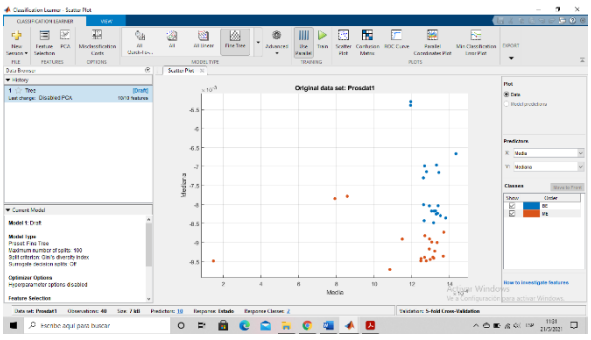


Figura 2.6. Ventana de Classification Learner.
(Matlab®, 2020)

La figura 2.7 muestra el tipo de entrenamiento que se quiere aplicar al conjunto de datos, se puede seleccionar varios al mismo tiempo para verificar cual es el más eficiente.

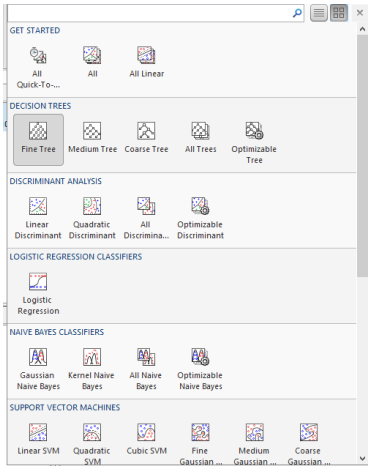


Figura 2.7. Opciones para el entrenamiento dentro de Classification Learner.
(Matlab®, 2020)

El botón Train (Figura 2.8) es el que da inicio al entrenamiento y las opciones a la derecha permiten conocer de diferentes formas graficas la eficiencia del entrenamiento realizado.

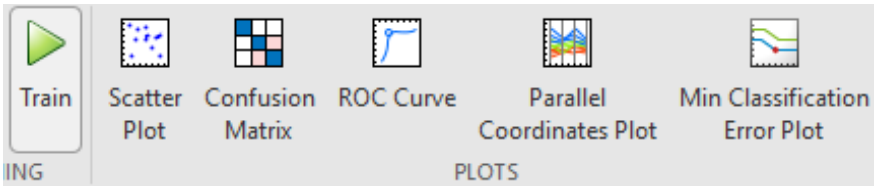


Figura 2.8. Opciones visuales para el entrenamiento del algoritmo de clasificación.
(Matlab®, 2020)

El botón Export (Figura 2.9) ofrece 3 opciones de las cuales se pueden exportar un gráfico del aprendizaje, generar una función respectiva al algoritmo de clasificación o exportar el modelo de predicción.

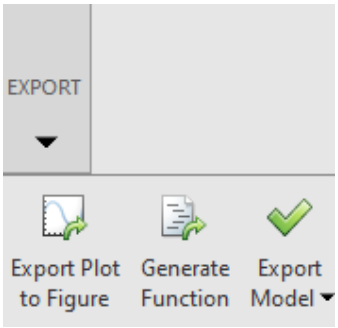


Figura 2.9. Opciones de exportación del modelo de predicción obtenidos por el aprendizaje autónomo.
(Matlab®, 2020)

2.3.1 PROCESAMIENTO DE DATOS.

El procesamiento de datos es la parte inicial y fundamental que permite el entrenamiento del algoritmo de clasificación. El conjunto de datos adquiridos representa vibraciones producidas por ciertos elementos de un motor de combustión interna, cada conjunto de datos consta de puntos captados con un valor numérico los cuales gráficamente pueden representar las oscilaciones características de una vibración.

Para un entrenamiento de Machine Learning es importante otorgar conjuntos de datos con características y etiquetas que faciliten el aprendizaje por parte del algoritmo. Los datos proporcionados de vibraciones al ser valores numéricos que varían acorde a la forma de la oscilación se los puede procesar de tal forma que aumente la cantidad de características, siendo un grupo extenso de valores es posible obtener cifras estadísticas que sean capaces de diferenciar a un grupo del otro al momento de la clasificación.

2.3.1.1 Obtención de datos estadísticos.

Si bien el conjunto de datos es capaz de describir vibraciones medidas no se las puede usar directamente en el proceso de entrenamiento del algoritmo de clasificación, es necesario simplificar el enorme grupo de datos a valores más característicos que representen todo el conjunto en sí. Para esto calcular valores estadísticos es lo adecuado, donde por la variación de valores entre los conjuntos de datos las cifras obtenidas servirán como características de cada muestra.

Si bien las vibraciones constan de oscilaciones fácilmente cuantificables ya sea por su amplitud, periodo y frecuencia, estas no son suficiente para obtener un aprendizaje altamente eficiente, por lo que aumentar el número de características por grupo de datos es una buena alternativa y gracias a las operaciones estadísticas es posible. Cada grupo de 40 000 datos numéricos será sometido a diferentes operaciones las cuales dará como resultado 10 diferentes características las que el algoritmo de clasificación tomará como referencia en su entrenamiento.

Al ser conjuntos de datos extensos se ha planteado realizar las siguientes acciones estadísticas:

Media.

La media o promedio, es una operación matemática que consiste en encontrar un dígito numérico que corresponda a la suma de todos los valores dividida para la cantidad de valores que contenga el grupo (Salazar, 2018). MATLAB® permite realizar esta operación estadística de forma automática mediante fórmulas o comandos, debido a la gran cantidad de valores numéricos en cada muestra de datos usar comandos son la mejor opción.

$$\bar{\boldsymbol{x}} = \frac{\boldsymbol{1}}{\boldsymbol{n}} . \left(\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{xi} \right) \tag{1}$$

Donde:

$\bar{x}$ = media aritmética o promedio.
xi = datos de la variable
n= tamaño de la muestra.
$\sum_{i=1}^{n} xi$= suma de todos los valores

El comando usado para realizar este procedimiento matemático es “mean(Datos)” que permite obtener la media de forma instantánea.

Mediana.

La mediana es una operación matemática que se basa en obtener un número central dentro del conjunto de datos, es decir la mitad de los números tendrán valores superiores y la otra mitad valores inferiores a dicho número (Alea V. & Jimenes E., 2015).

Para conjuntos de datos impares.

$$\boldsymbol{Me} = \boldsymbol{X}_{\frac{\boldsymbol{N+1}}{\boldsymbol{2}}}$$

Para conjuntos de datos pares. (2)

$$\boldsymbol{Me} = \frac{\boldsymbol{1}}{\boldsymbol{2}} . \left(\boldsymbol{X}_{\frac{\boldsymbol{N}}{\boldsymbol{2}}} + \boldsymbol{X}_{\frac{\boldsymbol{N}}{\boldsymbol{2}}\boldsymbol{+1}} \right)$$

Donde: (3)

Me= mediana

N= tamaño de la muestra

El comando usado para su obtención mediante la programación de MATLAB® es "median(Datos)", permite obtener el valor de la mediana de forma automática.

Moda.

La moda se busca en los grupos de datos identificando que valor numérico es el que más se repite o el que más veces se presenta dentro del conjunto de datos (Salazar, 2018).

$$\boldsymbol{Mo = Li + \left(\frac{f_i - f_{i-1}}{f_i - f_{i-1} + f_i - f_{i+1}} . a\right)}$$

Donde: (4)

Mo= moda

Li=límite inferior del intervalo con mayor frecuencia absoluta.

Fi-1= es la frecuencia absoluta anterior a la de mayor frecuencia.

Fi+1= es la frecuencia absoluta del siguiente intervalo.

a= es la amplitud del intervalo de mayor frecuencia absoluta.

Dentro de la programación de MATLAB® es más apropiado y eficiente usar el comando específico para esta opción, "mode(Datos)".

Media cuadrática.

La media cuadrática es una operación matemática la cual se obtiene sumando todos los dígitos del conjunto de datos, elevándolos al cuadrado, sacando su media y por último obteniendo su raíz cuadrada (García et al., 2014).

$$\boldsymbol{MC = \sqrt{\frac{\sum x^2}{n}}}$$

Donde: (5)

MC= media cuadrática

X= datos de la muestra.

n= tamaño de la muestra

Dentro de la programación en MATLAB ® se puede realizar esta operación por separado, ya que no existe un comando que permita calcular este valor.

Curtosis.

La curtosis es una medida estadística capaz de determinar el nivel de concentración de los valores alrededor de una zona central de distribución de frecuencias, es un valor muy usado en grupos de datos que describen parábolas u oscilaciones (Alvarez, 2012). Este valor es fácilmente calculable en MATLAB®.

$$\boldsymbol{K = \frac{\sum_{i=1}^{N}(xi - \bar{x})^4}{(N-1)s^4}} \quad (6)$$

Donde:

s= desviación estándar.

$\bar{x}$= la media de la distribución

N= número de observaciones de la muestra.

El comando empleado para obtener este valor mediante la programación de MATLAB ® es "kurtosis(Datos)".

Asimetría.

La asimetría al igual que la curtosis es un proceso matemático de gran uso en conjuntos de datos que describen parábolas u oscilaciones, la asimetría describe la cantidad de datos existentes en la parte superior e inferior de datos a partir de la media (Alea V. & Jimenes E., 2015).

$$CA_F = \frac{\sum_{i=1}^{N}(xi - \bar{x})^3}{N.{s_x}^3} \quad (7)$$

Donde:

s= desviación estándar.

$\bar{x}$= la media de la distribución

N= número de observaciones de la muestra.

La asimetría es fácilmente obtenible dentro de la programación de MATLAB ® por medio del comando “skewness(Datos)”

Máximo.

El máximo en un conjunto de datos equivale al número con la cifra más alta, no existe una fórmula exacta para su cálculo por este motivo es necesario obtenerlo de forma más rápida mediante medios informáticos. Este valor lo se lo obtiene usando el comando “max(datos)”, dentro de la programación de MATLAB ®.

Mínimos.

El mínimo en un conjunto de datos es la cifra con menor valor numérico, al igual que el máximo es necesario obtener este valor de forma rápida por medios informáticos. El valor mínimo al igual que el máximo únicamente necesita un comando de MATLAB® para su obtención, “min(datos)”.

Varianza.

La varianza se representa como una medida de dispersión que describe la variabilidad de datos respecto a la media (Alea V. & Jimenes E., 2015), aunque se puede calcular de forma manual se puede usar comandos de MATLAB®.

$$\sigma^2 = \frac{1}{N}.\sum_{i=1}^{N}(xi - \bar{x})^2 \quad (8)$$

Donde:

σ^2= varianza.

N= número de muestra de datos.

$\bar{x}$= media de la muestra.

El comando de MATLAB ® necesario para obtener este valor es “var(datos)”.

Desviación estándar.

La desviación estándar es un valor numérico dentro de un conjunto de datos que permite cuantificar la variación dentro los datos (Alea V. & Jimenes E., 2015), su obtención se simplifica si ya se tiene la varianza calculada, ya que únicamente se debe sacar la raíz cuadrada de dicho valor.

$$\sigma = \sqrt{\frac{1}{N}.\sum_{i=1}^{N}(xi - \bar{x})^2} \quad (9)$$

Donde:

σ= varianza.

N= número de muestra de datos.

$\bar{x}$= media de la muestra.

Aunque para obtener este valor es necesario únicamente sacar la raíz de la varianza, es posible también usar el comando de MATLAB® “std(datos)”.

2.3.1.2 Caracterización y etiquetado de conjuntos de datos.

La aplicación de Machine Learning de MATLAB® que se usa en esta investigación, Classification Learner, es un tipo de aprendizaje autónomo supervisado, es decir, los datos proporcionados para el entrenamiento deben contener variables independientes y dependientes o comúnmente llamadas características y etiquetas. Las características son usadas por el algoritmo como referencia para su aprendizaje, y las etiquetas son los grupos a los que pertenecen dichas características.

Al obtener grupos estadísticos de las muestras de datos se puede usarlos cada uno como una característica individual, teniendo así 10 variables independientes. La etiqueta en esta investigación se refiere al estado del componente al cual se realizó la toma de vibraciones, al ser un estudio enfocado al mantenimiento vehicular son necesarias vibraciones que representan el buen y mal estado (BE, ME) de los componentes medidos.

Tabla 2.3. Nomenclatura usada para las etiquetas de las muestras de datos.

Nomenclatura	**Significado**
BE	Pertenece a los datos de vibraciones pertenecientes al buen estado del motor.
ME	Pertenece a los datos de vibraciones pertenecientes al mal estado del motor.
MEF1	Pertenece a los datos de vibraciones pertenecientes al mal estado del motor, captados dentro de la primera falla simulada.
MEF2	Pertenece a los datos de vibraciones pertenecientes al mal estado del motor, captados dentro de la segunda falla simulada.
MEF3	Pertenece a los datos de vibraciones pertenecientes al mal estado del motor, captados dentro de la tercera falla simulada.

Para ejecutar el entrenamiento del algoritmo de clasificación es necesario otorgar al programa una tabla de datos que contenga características y etiquetas claramente comprensibles, de esta forma el software es capaz de identificar cada una automáticamente. La tabla 2.4 muestra la forma de la tabla obtenida posteriormente al procesamiento de datos, donde se aprecia las 10 respectivas características y en la última fila las etiquetas, las cuales en este caso son únicamente 2.

Tabla 2.4. Formato de las tablas usadas en el entrenamiento del algoritmo de clasificación.

Características										Etiquetas
Media	Mediana	Moda	Media cuadrática	Curtosis	Asimetría	Máximo	Mínimo	Varianza	Desviación estándar	Estado.
-	-	-	-	-	-	-	-	-	-	BE
-	-	-	-	-	-	-	-	-	-	ME

La cantidad de grupos de datos que se procese y se adjunte a la tabla dependerá directamente en la eficiencia del aprendizaje, entre más muestras de datos contenga la tabla mejor será el entrenamiento del algoritmo de clasificación.

2.3.2 ENTRENAMIENTO DEL ALGORITMO DE MACHINE LEARNING.

El entrenamiento del algoritmo de clasificación es el inicio del aprendizaje autónomo, aquí se proporciona los datos al ordenador para que este los analice y procese. Aunque la fase de Machine Learning empieza ya con la caracterización y etiquetado de datos el entrenamiento es donde se ejecuta el aprendizaje efectivo y de donde se obtiene los primeros resultados de este estudio.

2.3.2.1 Importación de datos a MATLAB®.

Los datos tomados de las vibraciones son guardados en hojas de Excel, para su respectivo procesamiento es necesario importarlos hacia MATLAB®, la opción Import Data es capaz de importar archivos y toda su información contenida. Usando comandos también es posible importar hojas de Excel, siendo esta opción la usada dentro de la programación de procesamiento de datos, xslread(“archivo”) es el comando mencionado.

2.3.2.2 Exportación de información hacia aplicación de clasificación.

Para iniciar el aprendizaje autónomo dentro de MATLAB® se inicia la aplicación de Classification Learner, se la puede encontrar dentro del apartado de Machine Learning and Deep Learning, para iniciar esto la tabla que será usada dentro del entrenamiento ya se debe encontrar dentro del Worckspace.

El paso siguiente es iniciar una nueva sesión usando la tabla de entrenamiento ubicada en el Worckspace, es necesario verificar las categorías y predictores para así seleccionar únicamente las que serán de utilidad al iniciar el aprendizaje. Es importante ajustar la cantidad de folders que serán usados para el entrenamiento, realizado esto ya se iniciara una sesión de Machine Learning capaz de ejecutar un entrenamiento de clasificación.

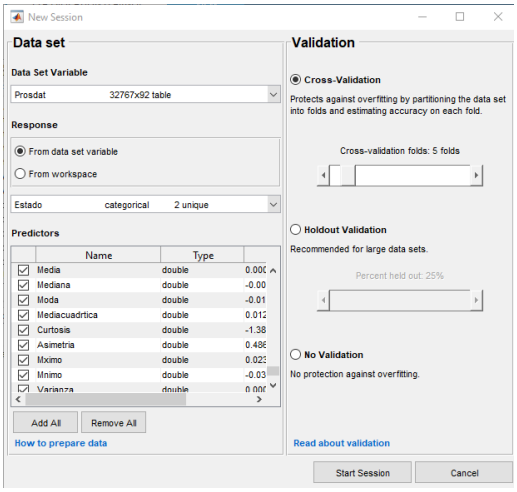


Figura 2.10. Selección de características y etiquetas del conjunto de datos. (Matlab®, 2020)

2.3.2.3 Entrenamiento por árbol de decisión.

Para realizar el entrenamiento existen varias opciones de las cuales se puede elegir, dependiendo de cada una de ellas se obtiene un modelo predictivo de mayor o menor eficiencia. El conjunto de datos se basa en realizar una clasificación binaria, por lo que es adecuado usar árboles de decisión y realizar el entrenamiento usando todas las opciones, y así validar la que mejor porcentaje de efectividad tenga.

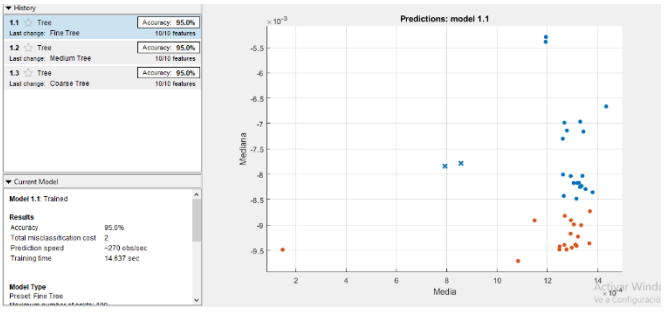


Figura 2.11. Ventana de inicio de Clasification Lerner con los datos ingresados. (Matlab®, 2020)

Los 3 modelos dan como resultado del aprendizaje una precisión del 95%, este porcentaje es producto de un buen procesamiento de datos inicial al elaborar la tabla inicial usada para el entrenamiento.

2.3.2.4 Interpretación de Plots (gráficos).

Scatter Plot (gráfico de dispersión)

La siguiente figura (Figura 2.12) se genera al terminar el entrenamiento del algoritmo de clasificación, cada punto azul representa a un grupo de datos con la etiqueta de buen estado (BE) y cada punto naranja representa a un grupo de datos con la etiqueta de mal estado (ME). Siguiendo esto, se aprecia dos puntos alejados marcados con una x, son 2 grupos de datos que el algoritmo tuvo problemas de clasificar, por este motivo el 95% de precisión.

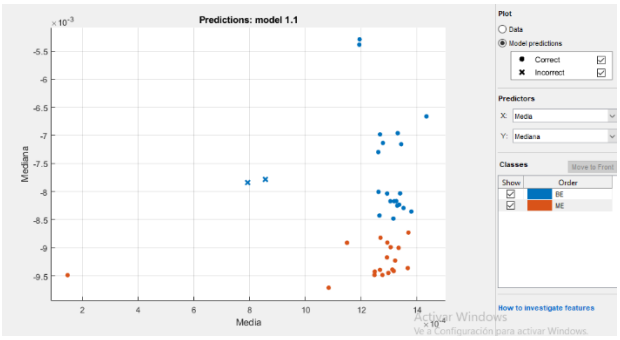


Figura 2.12. Diagrama de dispersión de Classification Learner.
(Matlab®, 2020)

Confusion Matrix (matriz de confusión).

La matriz de confusión tiene una fácil interpretación, al tener únicamente 2 grupos de clasificación, se simplifica. Hay 4 recuadros, cada uno contiene el número de grupos de datos que se le ha otorgado en la clasificación, los recuadros azules son los que se realizó correctamente. Dentro del recuadro BE está la cifra 20, esto significa que de 20 grupos de datos en buen estado todos fueron correctamente clasificados mientras que, en el recuadro ME únicamente hay 18 grupos clasificados correctamente, entonces, en el tercer recuadro pintado están 2 grupos de datos los cuales el algoritmo no pudo procesarlos, esto representa el 5% de fallo del entrenamiento.

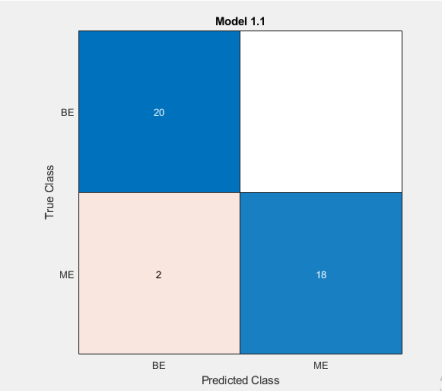


Figura 2.13. Diagrama de dispersión producto del entrenamiento del Classification Learner.
(Matlab®, 2020)

ROC curve (curva ROC)

La curva ROC es fácil de interpretar, dentro de la zona celeste se encuentra los conjuntos de datos que, si fueron clasificados correctamente, por ese motivo está marcada esta zona con AUC=0.95 que representa al 95%. El otro 5 por ciento restante se queda fuera del gráfico y se lo puede apreciar por la línea inclinada de la izquierda, entre más la línea se acerque al 1 mejor será el entrenamiento.

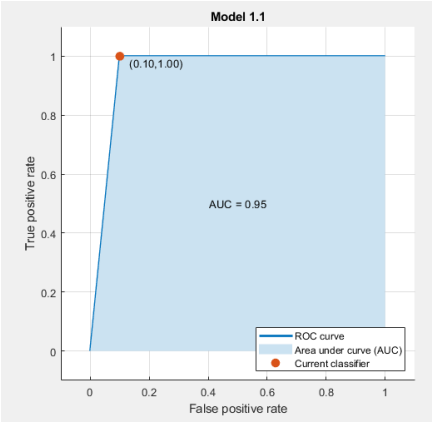


Figura 2.14. Curva de ROC.
(Matlab®, 2020)

Parallel Coordinates Plot (gráfico de coordenadas paralelas).

Dentro de la figura 2.15 se aprecia cada una de las características o predictores de los datos ingresados, cada grupo de datos tiene su propia línea que se grafica respectivamente por sus

características, dentro de BE están las líneas azules y dentro de ME las líneas rojas. Las líneas continuas se otorgan a los datos que, si fueron clasificados correctamente, únicamente hay 2 líneas entrecortadas y con una x en cada predictor, esto significa que ellos no fueron clasificados dentro de ningún grupo.

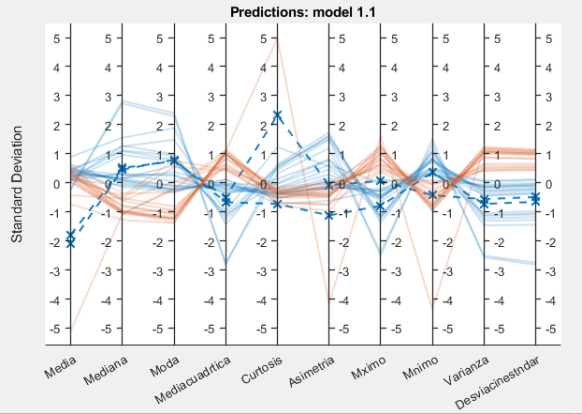


Figura 2.15. Gráfico de coordenadas paralelas.
(Matlab®, 2020)

2.3.3 OBTENCIÓN DE ECUACIÓN DE PREDICCIÓN.

El entrenamiento del algoritmo de clasificación se ejecuta con un fin, el ser usado como un modelo de predicción y clasificación el cual permita procesar nuevos grupos de datos. El algoritmo de clasificación de MATLAB® permite extraer el modelo de predicción de forma directa y fácil, de igual forma su uso no es nada complejo.

El botón Export permite ejecutar 3 opciones, una de ellas permite extraer la ecuación de predicción directamente hacia hoja de comandos. La opción Export Model exporta el modelo de predicción de forma completa o compacta, en ambas opciones se obtiene una ecuación de predicción la cual podrá ser usada con nuevos datos.

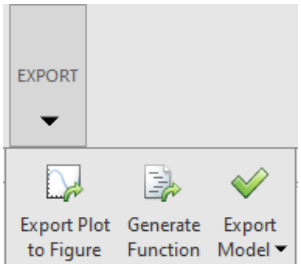


Figura 2.16. Opciones de exportación del modelo de predicción.

(Matlab®, 2020)

La ecuación de predicción se genera de forma automática, en la siguiente figura se visualiza el modelo generado dentro de la hoja de comandos, de aquí lo que realmente se emplea es la ecuación yfit = trainedModel2.predictFcn(T), únicamente modificándola ya será capaz de usarse.

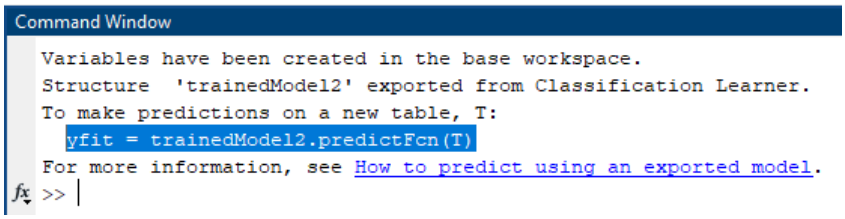


Figura 2.17. Modelo de predicción.

(Matlab®, 2020)

2.3.3.1 Uso de la ecuación de predicción.

Para la realización de predicciones que es por lo que realmente se emplea el aprendizaje autónomo, es necesario tener una tabla nueva de datos con la estructura exactamente igual a la usada en el entrenamiento, con la única diferencia que no posea el recuadro de etiquetas, ya que esto será lo que se va a predecir.

Tabla 2.5. Formato de tabla usada en nuevas predicciones.

Media	Mediana	Moda	Media cuadrática	Curtosis	Asimetría	Máximo	Mínimo	Varianza	Desviación estándar
-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-

La tabla debe encontrarse dentro del Workspase, por lo que si no se encuentra en formato de MATLAB® es necesario exportarla hacia ahí, posteriormente se copia la ecuación de predicción y dentro de los paréntesis donde se ubica una T, se debe remplazar por el nombre de la tabla a clasificar, se debe presionar Enter y el modelo se encargara de realizar el trabajo de forma automática.

```
>>   yfit = trainedModel2.predictFcn(Prosdat11)

yfit =

  categorical

     BE
```

Figura 2.18. Ecuación de predicción y su resultado.

El resultado de la clasificación se basa en el porcentaje de eficiencia del aprendizaje, por lo que para realizar predicciones tiene un marguen de error muy bajo, el modelo es funcional y trabaja acorde a lo esperado.

2.3.4 OBTENCIÓN DE FUNCIÓN DE MACHINE LEARNING

Dentro del apartado 2.2.3 se mencionó que es posible exportar un modelo de predicción hacia la hoja de comandos de MATLAB®, de igual forma es posible generar una función que no solo permita realizar predicciones, sino también reentrenar el algoritmo con nuevas tablas de datos. Esta opción es aún más completa y se la puede guardar como un programa independiente el cual no necesita abrir las aplicaciones de Machine Learning para funcionar.

Una vez ejecutado el aprendizaje la opción Export permite generar una función acorde al entrenamiento, usando los predictores y categorías introducidas inicialmente se genera una función dentro de la ventana perteneciente al editor de programación. La función creada contiene además instrucciones de su uso lo que hace más fácil su ejecución, hay que tomar en cuenta que es necesario modificar algunas líneas de la programación exportada dentro de la función de clasificación.

2.3.4.1 Modificación de la función de predicción.

Para que la programación creada en la función de predicción corra de forma correcta es necesario modificar algunas líneas, estas líneas corresponden a nuevas tablas de datos para el entrenamiento y el nombre del modelo de predicción.

En la primera línea (figura 2.19), se debe borrar el apartado de función, de esta forma se extrae toda la programación a un estado ordinario. Al final de la línea dentro de los paréntesis se debe cambiar la palabra “training data” y colocar el nombre de la tabla que se usará para el reentrenamiento, este cambio se lo debe realizar en todas las líneas donde se encuentre “training data”.

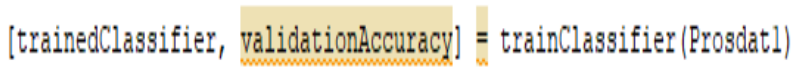


Figura 2.19. Primera línea de programa de Machine Learning.

De igual forma en todas las líneas donde esté la palabra trainClassifier, se la debe sustituir con el nombre en el que se guardó el algoritmo de predicción, en este estudio se dejó el mismo nombre, ya que no genera ningún inconveniente al correr el programa y simplifica la edición del mismo.

2.3.4.2 Predicción de nuevo conjunto de datos.

Al ejecutar el programa ya modificado, se genera dentro de la hoja de comandos el resultado del nuevo entrenamiento, entre lo más relevante se tiene la ecuación de predicción y la validez del nuevo modelo de clasificación.

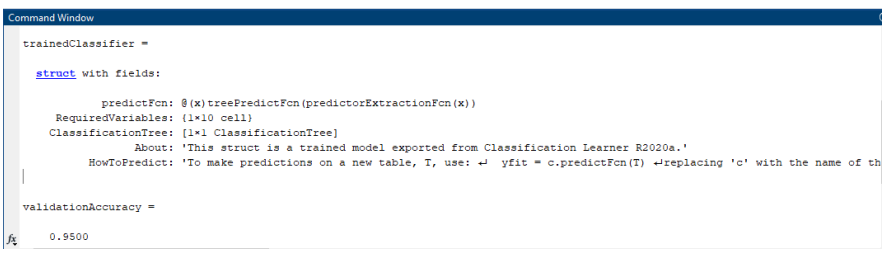


Figura 2.20. Resultado de un nuevo entrenamiento del programa de Machine Learning.

Para realizar nuevas predicciones con este modelo se realiza lo mencionado ya en el punto 2.2.4.1.

2.3.5 PROCESAMIENTO DE NUEVOS DATOS.

Para la ejecución de la predicción o clasificación es necesario nuevas muestras de datos de vibraciones, se debe caracterizar cada una tomando en cuenta que únicamente se requiere los predictores y no las categorías, ya que estas serán las que el algoritmo de clasificación asigne. La programación de MATLAB® permite dicho procesamiento de datos al mismo tiempo que genera la tabla correspondiente al formato necesario para realizar una predicción.

2.3.5.1 Programación para procesamiento de nuevos datos.

Los conjuntos de datos adquiridos contienen 40 000 datos, mediante programación de MATLAB® se puede extraer las 10 operaciones estadísticas usadas como características o predictores de forma automática, generando una tabla directamente dentro del Workspase.

La programación que se generó para el procesamiento de datos se encarga inicialmente de importar la información o los valores de vibraciones de un archivo de Excel, una vez la información dentro de MATLAB® se aplican los comandos mostrados en el punto 2.3.1.1 y se genera una tabla con estos valores.

```
Vibraciones=xlsread("Dat1"); %Ingrese el nombre entre las comillas de la tabla que contiene los datos a procesar.
Datos=[Vibraciones];

Media=mean(Datos);
Mximo=max(Datos);
Mnimo=min(Datos);
Mediana=median(Datos);
Desviacinestndar=std(Datos);
Varianza=var(Datos);
Curtosis=kurtosis(Datos);
Moda=mode(Datos);
Asimetria=skewness(Datos);
Dc=(Datos).^2;
Mdc=mean(Dc);
Mediacuadrtica=((Mdc).^(1/2));
Tabla1=table(Media,Mediana,Moda,Mediacuadrtica,Curtosis,Asimetria,Mximo,Mnimo,Varianza,Desviacinestndar);
disp("Procesamiento de valores");
disp(Tabla1);
plot(Datos);
```

Figura 2.21. Programa responsable del procesamiento de datos.

2.3.6 UNIÓN DE PROGRAMACIÓN Y FUNCIÓN DE MACHINE LEARNING.

El tercer objetivo de esta investigación consiste en realizar una programación que permita la ejecución del Machine Learning, de esta forma obtener predicciones de nuevos datos de vibraciones y así esto sea usado para la gestión del mantenimiento predictivo. El programa de procesamiento datos y la función de clasificación ya se encuentran trabajado correctamente por separado, la unión de ambos programas ofrece una predicción más automatizada reduciendo cada vez más la intervención humana, haciendo que su uso sea más sencillo.

En la programación unificada se colocó el apartado de procesamiento de datos inicialmente, de esta forma se genera la tabla necesaria para la predicción antes de ejecutar el reentrenamiento del algoritmo. Al final del programa está ya la ecuación de predicción lista, de esta forma únicamente corriendo el programa se obtendrá el procesamiento de las vibraciones, el reentrenamiento del algoritmo y la predicción del estado de forma inmediata.

2.3.7 TOMA DE DADOS DE VIBRACIONES.

La toma de vibraciones hacia el motor del tractor agrícola se las realiza con el fin de alimentar las tablas de datos necesarias para el reentrenamiento de los algoritmos de Machine Learning, al mismo tiempo que permite realizar las pruebas correspondientes para evaluar su eficacia en la predicción de fallas. El procedimiento es sencillo y es necesario el sensor de vibraciones, la tarjeta de adquisición de datos y el software LabVIEW.

2.3.7.1 Localización del sensor de vibraciones.

Para que el sensor responsable de captar las vibraciones producidas por el motor de combustión interna realice mediciones de buena calidad es necesario que sea colocado en un sitio completamente plano y donde el imán se pueda adherir sin problemas. Mientras el motor este encendido este produce vibraciones que se trasmiten a todos los componentes cercanos a él, esto no significa que el sensor pueda ser ubicado en cualquier lugar

perteneciente al MCI, es necesario que sea ubicado en el bloque motor a la altura del PMS de uno de sus cilindros.

Figura 2.22. Sensor de vibraciones ubicado en el bloque motor.

Para realizar las mediciones pertinentes a esta investigación el sensor se ubicó a la altura del punto muerto superior del cilindro número 3, se seleccionó este lugar por su accesibilidad e inexistencia de numeraciones u otros relieves que impidieran la correcta adherencia del imán.

2.3.7.2 Registro de datos por LabVIEW

El software LabVIEW permite una interacción entre el ordenador y el equipo de adquisición de datos, por medio de un programa es posible visualizar las vibraciones en forma de voltaje captadas por el sensor y además guardarlas en una hoja de Excel.

El programa es bastante intuitivo, se inicia la visualización de las vibraciones, se selecciona el archivo donde se ubicarán los datos y se los empieza a guardar. Se guardó un promedio de 40 muestras de datos de un aproximado de 8 segundos cada una.

2.3.8 SIMULACIÓN DE FALLA DEL MOTOR.

Dentro de la tabla 2.4, usada para el entrenamiento del algoritmo de clasificación es necesario datos de vibraciones del motor en buen funcionamiento y con algún fallo. La

simulación de fallas permite adquirir datos con diferentes irregularidades dentro de la combustión en la mezcla que se encuentre en cada cilindro, la solución óptima es alterar la alimentación de combustible hacia los inyectores.

2.3.8.1 Simulación de falla 1 (desregulación de presión de apertura de los inyectores).

El motor del tractor agrícola en el que se está trabajando tiene un sistema de alimentación de combustible de inyección directa, dichos inyectores son de accionamiento mecánico y se abren al exceder cierta presión del combustible. Para simular la primera falla es necesario variar la presión de apertura de los inyectores por medio de su tornillo de reglaje.

Figura 2.23. Variación de tornillos de reglaje de los inyectores, falla 1.

La descalibración se realiza uniformemente a cada inyector, se afloja el tornillo de reglaje un cuarto de vuelta de su calibración correcta disminuyendo así su presión de apertura ocasionando así un exceso de combustible inyectado dentro de la cámara de combustión.

2.3.8.2 Simulación de falla 2 (desregulación de presión de apertura de los inyectores).

Inicialmente se simuló una ligera descalibración de parte de los inyectores (Falla 1) donde se disminuyó levemente la presión necesaria para su apertura, ahora es necesario simular una falla con un sobre exceso de combustible donde a todos los inyectores se les descalibró el tornillo de reglaje girando 3 cuartos de vuelta hacia la derecha.

Figura 2.24. Desregulación de tornillo de reglaje, falla 2.

Se realizo la variación de presión a todos lo inyectores, ocasionando un fallo en general del motor, es perceptible el cambio en el sonido del motor más la cantidad de humo que ahora sale por el tubo de escape, por ende, la falla simulada es más notoria.

2.3.8.3 Simulación de falla 3 (desregulación de presión de apertura de los inyectores).

La tercera falla se la hizo de la misma forma que las anteriores, la única diferencia es que se usó diferentes reglajes para cada cilindro, de esta forma la combustión de cada uno será diferente a los otros 2. Esta es una falla más realista, ya que simula un escenario donde se desgastan los inyectores de una forma irregular entre sí.

Figura 2.25. Desregulación irregular del tornillo de reglaje, falla 3.

La regulación de los inyectores se realizó de tal forma que únicamente uno mantenga la presión de apertura óptima, los otros dos se varió levente y gravemente. El inyector número

1 se redujo su presión de apertura aflojando un cuarto de vuelta del tornillo de reglaje, el inyector número 2 es el único que mantiene su presión de apertura correcta, el inyector número 3 se redujo su presión de apertura aflojando el tornillo de reglaje 3 cuartos de vuelta. Todos los inyectores trabajan bajo una presión diferente de apertura y es notorio al escuchar el sonido del motor encendido.

2.3.9 VALIDACIÓN DEL APRENDIZAJE AUTÓNOMO.

Los datos de vibraciones adquiridos tanto en buen funcionamiento del motor como en simulación de fallas sirven no solo para el entrenamiento del algoritmo de clasificación, sino también para validar si dicho entrenamiento funciona correctamente usando datos que no fueron tomados en cuenta dentro del entrenamiento, y así realizar pruebas de su eficacia.

Es necesario realizar entrenamientos independientes de cada caso realizado, es decir de cada falla simulada. Entre mayor número de pruebas se realice se verifica mejor si el porcentaje dado por el entrenamiento corresponde a la funcionalidad del algoritmo clasificando datos de vibraciones.

CAPÍTULO III

3. RESULTADOS

La obtención de resultados fue posible inicialmente gracias a la captación de vibraciones del motor del tractor agrícola por medio del sensor y la tarjeta de adquisición de datos. Las vibraciones captadas son resultados iniciales los cuales se debe procesar por medio de MATLAB® para la respectiva obtención de tablas necesarias para el entrenamiento del algoritmo de clasificación.

Para esta investigación fue necesario crear un programa en MATLAB® y unirlo a la función de Machine Learning, así es posible procesar nuevos datos de vibraciones al mismo tiempo que se conoce el estado real del motor sin la necesidad de extraer sus piezas para las respectivas revisiones, las cuales quitan mucho tiempo. Los resultados obtenidos son la eficiencia del aprendizaje por parte del algoritmo de clasificación y si este puede ayudarnos a predecir fallas y así mejorar la gestión del mantenimiento vehicular.

3.1 CARACTERIZACIÓN Y ETIQUETADO DE DATOS DE VIBRACIONES.

El aprendizaje autónomo supervisado requiere datos previamente caracterizados y etiquetados, es decir, necesita una intervención humana que otorgue la información inicial de tal forma que el algoritmo únicamente la use para aprender de ella. Para las 3 simulaciones de fallas es necesario una tabla de datos para cada una, los datos de buen estado son los mismos para todas, mientras que los datos en mal estado son los que varían y corresponden a cada escenario.

3.1.1 DATOS EN BUEN ESTADO.

Inicialmente se tomó los datos en buen estado donde no se presentaba ninguna falla en suministro de combustible del motor, de igual forma el motor se encuentra en buen estado y

con todas sus piezas y reglajes en óptimas condiciones. Las mediciones son del motor a régimen de ralentí.

Tabla 3.1. Resultado del procesamiento de datos del motor en buen estado, 20 muestras.

	1 Media	2 Mediana	3 Moda	4 Mediacuadrtica	5 Curtosis	6 Asimetria	7 Mximo	8 Mnimo	9 Varianza	10 Desviacinestndar	11 Estado
1	0.0013	0.0013	0.0014	0.0017	33.6637	-0.2297	0.0149	-0.0126	1.1894e-06	0.0011	BE
2	0.0013	0.0013	0.0014	0.0017	33.7581	-0.1721	0.0155	-0.0112	1.1526e-06	0.0011	BE
3	0.0013	0.0013	0.0013	0.0017	33.2743	-0.1205	0.0162	-0.0132	1.0713e-06	0.0010	BE
4	0.0013	0.0013	0.0014	0.0017	34.3967	0.0231	0.0178	-0.0113	1.1871e-06	0.0011	BE
5	0.0013	0.0013	0.0014	0.0017	32.1675	-0.1219	0.0155	-0.0122	1.1273e-06	0.0011	BE
6	0.0013	0.0013	0.0013	0.0017	34.5679	-0.1257	0.0164	-0.0126	1.2241e-06	0.0011	BE
7	0.0013	0.0013	0.0013	0.0017	36.9350	-0.1769	0.0170	-0.0129	1.2284e-06	0.0011	BE
8	0.0013	0.0013	0.0014	0.0017	34.5982	-0.0841	0.0157	-0.0126	1.1107e-06	0.0011	BE
9	0.0014	0.0014	0.0014	0.0018	45.6971	-0.1620	0.0197	-0.0159	1.4579e-06	0.0012	BE
10	0.0014	0.0014	0.0013	0.0018	44.0875	-0.3713	0.0168	-0.0164	1.3445e-06	0.0012	BE
11	0.0014	0.0014	0.0014	0.0019	57.6449	-0.0188	0.0211	-0.0181	1.5845e-06	0.0013	BE
12	0.0014	0.0014	0.0013	0.0018	52.8084	-0.0093	0.0191	-0.0175	1.4975e-06	0.0012	BE
13	0.0014	0.0014	0.0014	0.0018	59.6483	-0.0121	0.0199	-0.0181	1.4370e-06	0.0012	BE
14	0.0014	0.0014	0.0014	0.0018	65.5890	-0.3797	0.0191	-0.0196	1.3950e-06	0.0012	BE
15	0.0014	0.0014	0.0014	0.0018	61.5211	-0.2103	0.0192	-0.0188	1.3851e-06	0.0012	BE
16	0.0014	0.0014	0.0013	0.0018	64.7375	0.1369	0.0190	-0.0182	1.2368e-06	0.0011	BE
17	0.0013	0.0014	0.0014	0.0018	65.2355	-0.0977	0.0190	-0.0175	1.3150e-06	0.0011	BE
18	0.0013	0.0013	0.0014	0.0018	71.0543	0.1627	0.0253	-0.0202	1.6789e-06	0.0013	BE
19	0.0014	0.0013	0.0013	0.0021	57.6474	0.3881	0.0264	-0.0228	2.3792e-06	0.0015	BE
20	0.0012	0.0012	0.0014	0.0016	58.6319	-0.2285	0.0191	-0.0158	1.1725e-06	0.0011	BE

La tabla 3.1 contiene los valores correspondientes al procesamiento de 20 muestras de vibraciones donde se obtuvo de 10 medidas estadísticas cada una. Esta tabla se toma como referencia en los 3 entrenamientos donde identifica que características identifican un buen estado por parte del motor.

3.1.2 FALLA NÚMERO 1 (desregulación de presión de apertura de los inyectores).

La falla número uno correspondiente a una ligera des calibración de los inyectores donde se redujo levemente la presión de apertura de ellos, para tomar los datos en este estado se realizó de igual forma a un régimen del motor de ralentí.

Tabla 3.2. Resultado del procesamiento de datos del motor con la falla 1, 20 muestras.

	1 Media	2 Mediana	3 Moda	4 Mediacuadrtica	5 Curtosis	6 Asimetria	7 Mximo	8 Mnimo	9 Varianza	10 Desviacinestndar	11 Estado
21	0.0013	0.0013	0.0015	0.0017	28.0024	-0.3411	0.0121	-0.0105	1.1409e-06	0.0011	ME
22	0.0013	0.0013	0.0013	0.0017	28.6812	-0.0722	0.0125	-0.0096	1.1319e-06	0.0011	ME
23	0.0013	0.0013	0.0014	0.0017	30.6003	0.0161	0.0145	-0.0110	1.1922e-06	0.0011	ME
24	0.0013	0.0014	0.0014	0.0017	31.1485	0.1208	0.0132	-0.0102	1.2265e-06	0.0011	ME
25	0.0013	0.0013	0.0013	0.0017	33.6642	-0.0317	0.0146	-0.0109	1.2153e-06	0.0011	ME
26	0.0013	0.0014	0.0015	0.0017	33.4928	0.2243	0.0134	-0.0097	1.1965e-06	0.0011	ME
27	0.0013	0.0013	0.0013	0.0017	33.8853	-0.0238	0.0129	-0.0111	1.1558e-06	0.0011	ME
28	0.0013	0.0013	0.0013	0.0017	31.3287	0.1498	0.0131	-0.0098	1.1317e-06	0.0011	ME
29	0.0013	0.0014	0.0014	0.0017	31.8178	0.2203	0.0142	-0.0104	1.1580e-06	0.0011	ME
30	0.0014	0.0014	0.0014	0.0017	31.4194	0.1696	0.0150	-0.0087	1.0591e-06	0.0010	ME
31	0.0013	0.0014	0.0015	0.0017	32.6119	0.3739	0.0132	-0.0141	1.1660e-06	0.0011	ME
32	0.0013	0.0014	0.0014	0.0017	32.4647	0.1357	0.0140	-0.0109	1.1139e-06	0.0011	ME
33	0.0013	0.0014	0.0012	0.0017	32.4044	0.2844	0.0138	-0.0100	1.1372e-06	0.0011	ME
34	0.0013	0.0014	0.0014	0.0017	35.3724	-0.0719	0.0150	-0.0134	1.1896e-06	0.0011	ME
35	0.0013	0.0014	0.0014	0.0017	31.0121	0.0913	0.0139	-0.0109	1.2134e-06	0.0011	ME
36	0.0013	0.0014	0.0015	0.0017	31.8950	0.1915	0.0152	-0.0116	1.2183e-06	0.0011	ME
37	0.0013	0.0014	0.0014	0.0017	32.2763	0.2184	0.0144	-0.0120	1.0584e-06	0.0010	ME
38	0.0014	0.0014	0.0014	0.0018	35.8074	0.2097	0.0153	-0.0113	1.2533e-06	0.0011	ME
39	0.0013	0.0014	0.0014	0.0018	27.8994	-0.0110	0.0141	-0.0109	1.3164e-06	0.0011	ME
40	0.0017	0.0017	0.0015	0.0020	33.1523	-0.0707	0.0139	-0.0147	1.1827e-06	0.0011	ME

Aunque la falla simulada era leve, los datos obtenidos fueron capaces de otorgar una tabla con valores diferentes a los datos en buen estado, estas 20 muestras son suficientes para ser usadas en el primer entrenamiento.

3.1.3 FALLA NÚMERO 2 (desregulación de presión de apertura de los inyectores).

La falla número 2 corresponde a una descalibración de inyectores más grave, donde la presión necesaria para la apertura de los inyectores se redujo notablemente, aunque las mediciones se realizaron a ralentí es notorio el cambio en la marcha del motor.

Tabla 3.3. Resultado del procesamiento de datos del motor con la falla 2, 20 muestras.

	1 Media	2 Mediana	3 Moda	4 Mediacuadrtica	5 Curtosis	6 Asimetria	7 Mximo	8 Mnimo	9 Varianza	10 Desviacinestndar	11 Estado
21	0.0013	0.0013	0.0013	0.0023	59.4133	-0.3707	0.0257	-0.0239	3.4128e-06	0.0018	ME
22	0.0013	0.0013	0.0013	0.0024	54.5686	-0.2800	0.0269	-0.0243	3.8568e-06	0.0020	ME
23	0.0012	0.0012	0.0013	0.0022	53.9359	-0.3655	0.0254	-0.0239	3.5446e-06	0.0019	ME
24	0.0013	0.0013	0.0012	0.0022	52.4926	-0.3839	0.0245	-0.0236	3.2504e-06	0.0018	ME
25	0.0013	0.0014	0.0015	0.0022	54.6426	-0.4449	0.0246	-0.0243	3.1775e-06	0.0018	ME
26	0.0013	0.0014	0.0013	0.0023	52.5961	-0.3869	0.0239	-0.0239	3.3807e-06	0.0018	ME
27	0.0013	0.0014	0.0016	0.0023	49.1612	-0.3437	0.0231	-0.0247	3.3715e-06	0.0018	ME
28	0.0013	0.0013	0.0017	0.0023	50.0499	-0.3090	0.0246	-0.0234	3.7270e-06	0.0019	ME
29	0.0014	0.0014	0.0015	0.0023	50.2041	-0.5463	0.0261	-0.0230	3.6427e-06	0.0019	ME
30	0.0014	0.0014	0.0014	0.0024	52.2586	-0.6509	0.0245	-0.0250	3.7463e-06	0.0019	ME
31	0.0014	0.0014	0.0014	0.0024	46.6655	0.0370	0.0244	-0.0234	3.7213e-06	0.0019	ME
32	0.0014	0.0014	0.0014	0.0024	47.8872	-0.2703	0.0253	-0.0253	3.7829e-06	0.0019	ME
33	0.0014	0.0014	0.0015	0.0024	45.5536	-0.2446	0.0254	-0.0236	3.9010e-06	0.0020	ME
34	0.0014	0.0014	0.0014	0.0024	46.4772	-0.5453	0.0258	-0.0234	3.7154e-06	0.0019	ME
35	0.0014	0.0014	0.0013	0.0023	45.7874	-0.5346	0.0238	-0.0237	3.5388e-06	0.0019	ME
36	0.0011	0.0011	0.0012	0.0024	42.7698	-0.4135	0.0255	-0.0243	4.3422e-06	0.0021	ME
37	0.0014	0.0014	0.0015	0.0025	50.6467	-0.1912	0.0299	-0.0263	4.2817e-06	0.0021	ME
38	0.0014	0.0014	0.0012	0.0024	50.9763	-0.2945	0.0271	-0.0248	3.9109e-06	0.0020	ME
39	0.0014	0.0014	0.0013	0.0024	45.9476	-0.4419	0.0245	-0.0241	3.7179e-06	0.0019	ME
40	0.0014	0.0014	0.0016	0.0025	50.0597	-0.9037	0.0254	-0.0263	4.1255e-06	0.0020	ME

Al ser una falla más grave, los valores dentro de la tabla 3.3 cambian considerablemente en comparación a la tabla de datos en buen estado. Las 20 muestras de igual forma serán usadas en el segundo entrenamiento del algoritmo y así poder predecir fallas con síntomas similares a esta.

3.1.4 FALLA NÚMERO 3 (desregulación de presión de apertura de los inyectores).

La falla número 3 corresponde a un caso más real donde cada inyector tiene un reglaje diferente, esto asemeja un desgaste desigual en todos los inyectores, de igual forma que un diferente suministro de combustible para cada cilindro alterando así la combustión de cada uno. Las mediciones se realizaron a régimen de ralentí.

Tabla 3.4. Resultado del procesamiento de datos del motor con la falla 3, 20 muestras.

	1 Media	2 Mediana	3 Moda	4 Mediacuadrtica	5 Curtosis	6 Asimetria	7 Mximo	8 Mnimo	9 Varianza	10 Desviacinestndar	11 Estado
21	0.0013	0.0013	0.0014	0.0018	56.4743	1.3050	0.0259	-0.0165	1.5882e-06	0.0013	ME
22	0.0013	0.0013	0.0013	0.0018	60.0396	1.4077	0.0265	-0.0169	1.6180e-06	0.0013	ME
23	0.0013	0.0013	0.0012	0.0019	52.8266	1.2446	0.0249	-0.0161	1.6929e-06	0.0013	ME
24	0.0013	0.0013	0.0013	0.0019	53.5253	1.1708	0.0252	-0.0174	1.6565e-06	0.0013	ME
25	0.0014	0.0014	0.0014	0.0019	58.3446	1.2559	0.0264	-0.0178	1.8652e-06	0.0014	ME
26	0.0014	0.0014	0.0015	0.0019	56.4667	1.0247	0.0276	-0.0196	1.8339e-06	0.0014	ME
27	0.0013	0.0013	0.0013	0.0019	52.2270	1.0904	0.0247	-0.0172	1.8133e-06	0.0013	ME
28	0.0013	0.0013	0.0013	0.0019	43.6932	0.6594	0.0240	-0.0191	1.7298e-06	0.0013	ME
29	0.0013	0.0013	0.0013	0.0019	43.4650	0.6882	0.0236	-0.0180	1.7288e-06	0.0013	ME
30	0.0014	0.0013	0.0013	0.0019	41.9541	0.7627	0.0242	-0.0189	1.7066e-06	0.0013	ME
31	0.0013	0.0013	0.0013	0.0019	40.7189	0.6136	0.0239	-0.0191	1.8895e-06	0.0014	ME
32	0.0016	0.0015	0.0015	0.0023	43.2867	0.5636	0.0299	-0.0238	2.5615e-06	0.0016	ME
33	0.0014	0.0014	0.0013	0.0021	69.9472	0.6784	0.0282	-0.0247	2.3838e-06	0.0015	ME
34	0.0013	0.0013	0.0013	0.0019	66.0878	0.8145	0.0272	-0.0220	2.1299e-06	0.0015	ME
35	0.0014	0.0014	0.0014	0.0021	62.2332	0.6136	0.0295	-0.0272	2.3935e-06	0.0015	ME
36	0.0013	0.0013	0.0013	0.0019	66.1047	0.8528	0.0297	-0.0233	1.8505e-06	0.0014	ME
37	0.0014	0.0014	0.0014	0.0019	51.6161	0.6975	0.0252	-0.0208	1.7255e-06	0.0013	ME
38	0.0013	0.0013	0.0013	0.0018	40.1608	0.6732	0.0228	-0.0170	1.6465e-06	0.0013	ME
39	0.0014	0.0014	0.0014	0.0019	48.4579	0.7153	0.0250	-0.0203	1.7732e-06	0.0013	ME
40	0.0014	0.0013	0.0014	0.0019	46.0829	0.8555	0.0238	-0.0183	1.7689e-06	0.0013	ME

Las vibraciones en esta falla son más notorias e irregulares y es exactamente lo que refleja la tabla 3.4, perteneciente a este caso, los valores en comparación a los de buen estado son completamente diferentes, por lo que las 20 muestras deben proporcionar un entrenamiento altamente eficiente.

3.2 ENTRENAMIENTO DEL ALGORITMO DE CLASIFICACIÓN.

Los entrenamientos se realizaron con las tablas 3.1, 3.2, 3.3 y 3.4 respectivamente, compartiendo únicamente los valores de buen estado, cada tabla se diferencia por contener valores correspondientes a cada falla simulada. Las tablas de entrenamientos contienen 40 muestras de datos donde la mitad corresponde a una etiqueta (BE) y la otra mitad a (ME).

El entrenamiento es realizado inicialmente por medio de la aplicación de Machine Learning perteneciente a MATLAB®, específicamente la de clasificación. Dependiendo de la diferenciación de características entre los datos de buen estado con los de mal estado el algoritmo tendrá la facilidad de clasificarlos y presentará un porcentaje respectivo a la eficiencia de dicho entrenamiento.

3.2.1 PRIMER ENTRENAMIENTO.

El primer entrenamiento se efectuó con el uso de una tabla que contenía 2 estados del motor (unión de tablas 3.1 y 3.2), las primeras 20 muestras de datos pertenecen a un estado óptimo de funcionamiento del motor, mientras que las 20 muestras consiguientes son el producto de la Falla número 1. Ambas categorías dentro del algoritmo de clasificación son una base de aprendizaje para el mismo usando estos datos como referencia en futuras predicciones.

Eficiencia del entrenamiento.

El entrenamiento dio como resultado una eficiencia del 97.5%, un porcentaje muy bueno dentro del Machine Learning y con la suficiente credibilidad para ser usado dentro de mantenimientos vehiculares.

Resultados gráficos del entrenamiento.

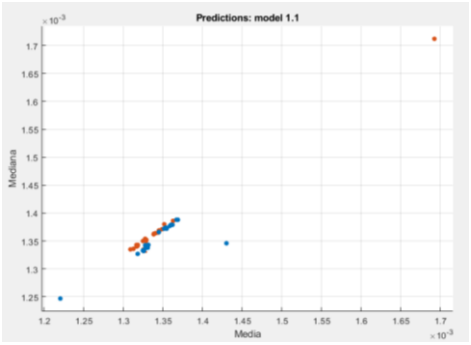
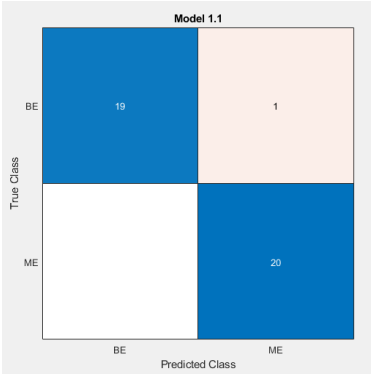
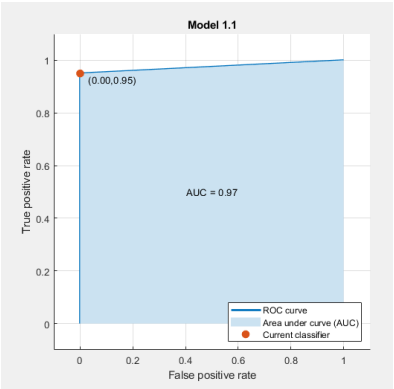
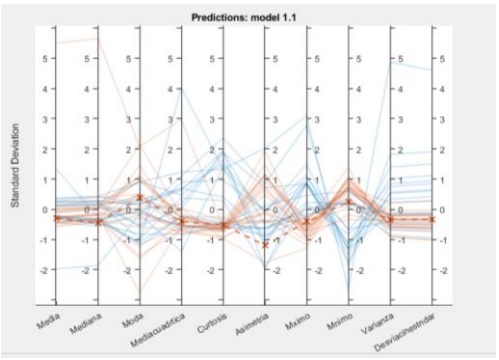
A	B
	
C	**D**
	

Figura 3.1. Resultados del Primer Entrenamiento. A: Diagrama de dispersión, B: Matriz de confusión, C: Curva de ROC, D: Grafico de coordenadas paralelas.

Dentro del diagrama de dispersión es apreciable la diferencia entre ambos grupos de datos, los pertenecientes al buen estado y mal estado del motor, es notorio que se encuentran muy cerca, ya que la falla simulada perteneciente a este entrenamiento era muy leve, y la diferencia entre ambos estados es apenas perceptible.

La matriz de confusión representa una única muestra de datos la cual no se pudo clasificar correctamente, ya que el algoritmo la entrecruzo con ambos estados, por ese motivo la eficiencia del 97,5%.

La curva ROC es una representación apreciable a la eficiencia del entrenamiento, indica el progreso de este y en qué sector se encuentra, si las coordenadas más se acercan en el eje de las ordenadas al 1 más eficiente es el proceso.

El gráfico de coordenadas paralelas es una representación del camino que sigue el algoritmo al clasificar los datos, pasa por cada característica por un punto específico asignado, es apreciable ver que cada grupo de datos siguen los mismos caminos, excepto el que no se pudo clasificar correctamente que se encuentra representado por líneas entrecortadas. Aunque en el diagrama de dispersión los datos son muy cercanos aquí ya se encuentra una diferencia notoria entre ambos estados.

3.2.2 SEGUNDO ENTRENAMIENTO.

Conjunto a los datos de buen estado que serán usados en todos los entrenamientos se usan las vibraciones captadas al simular la falla número 2 (tabla 3.3), esta falla es más severa que la anterior por lo que debe ser más notoria la diferencia tanto en el procesamiento de datos como en el entrenamiento.

Eficiencia del entrenamiento.

El entrenamiento dio como resultado una eficiencia del 97.5%, similar al anterior, aunque aquí la falla simulada es aún más severa, esto es una clara señal que el algoritmo de clasificación puede fácilmente identificar variaciones en datos diferentes a los pertenecientes del buen estado del motor.

Resultados gráficos del entrenamiento.

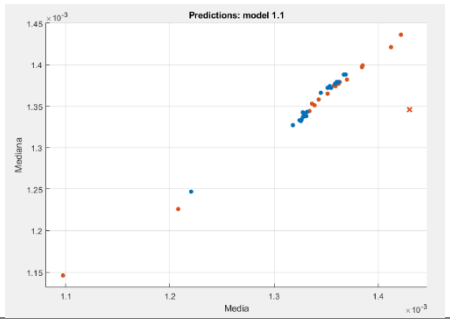
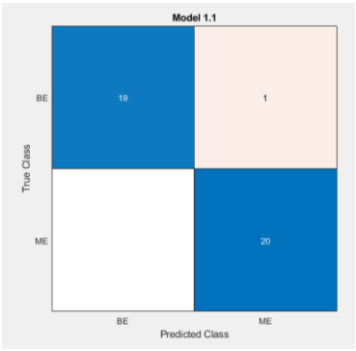
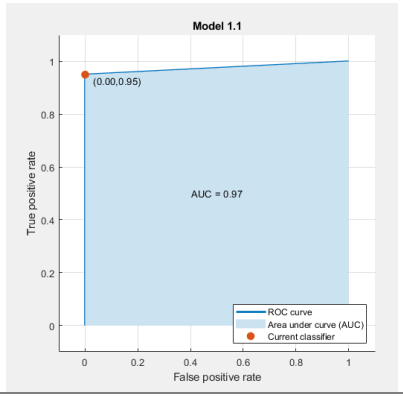
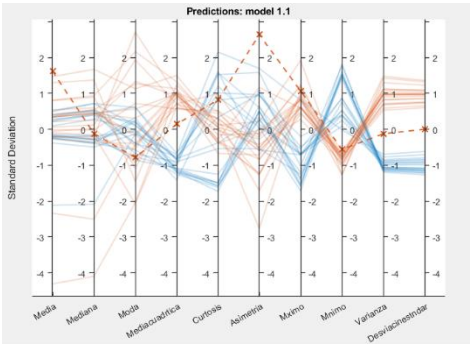
A	B
	
C	**D**
	

Figura 3.2. Resultados del Segundo Entrenamiento. A: Diagrama de dispersión, B: Matriz de confusión, C: Curva de ROC, D: Grafico de coordenadas paralelas.

Los resultados obtenidos dentro de las gráficas son similares a los del primer entrenamiento, la principal diferencia notoria es dentro del diagrama de dispersión donde los puntos representantes a cada muestra de datos de cada grupo ya se encuentran más dispersos a lo largo del plano, de igual forma dentro del diagrama de coordenadas paralelas se aprecia como los datos pertenecientes a la falla simulada toman un diferente camino al anterior entrenamiento, esto significa que si existe una variación notoria dentro de ambos entrenamientos.

3.2.3 TERCER ENTRENAMIENTO.

El último entrenamiento de forma individual es realizado usando los datos recopilados en la falla simulada número 3 (tabla 3.4), esta falla es más severa y real que las anteriores, ya que produce vibraciones irregulares con una desincronización entre los 3 inyectores, de igual forma que el entrenamiento número 2 aquí se usaron datos notoriamente diferentes a los de buen estado.

Eficiencia del entrenamiento.

El entrenamiento dio como resultado una eficiencia del 97.5%, igual a los entrenamientos anteriores, son porcentajes muy buenos y ya solo vasta validar su eficiencia con pruebas reales usando nuevos datos.

Resultados gráficos del entrenamiento.

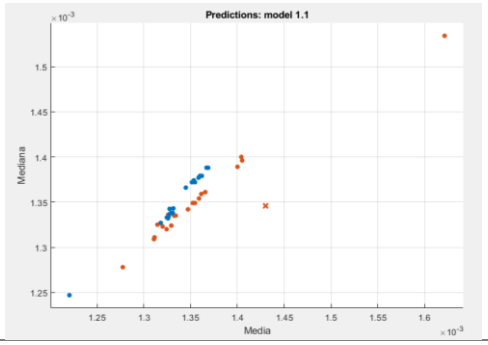
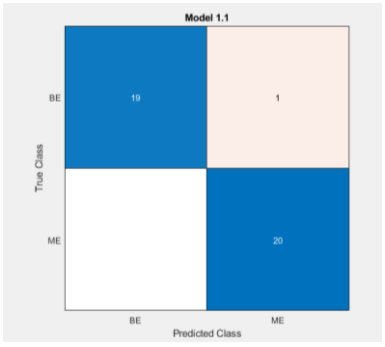
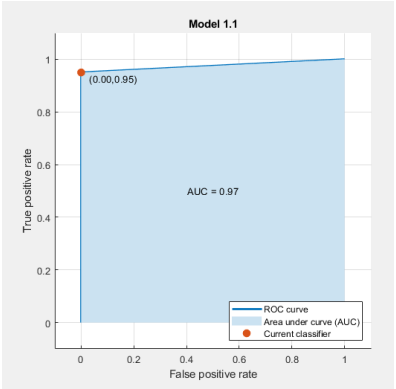
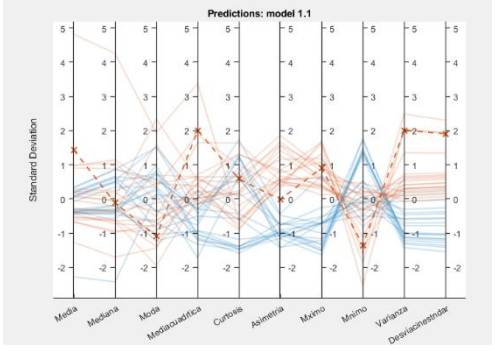
A	B
	
C	**D**
	

Figura 3.3. Resultados del tercer Entrenamiento. A: Diagrama de dispersión, B: Matriz de confusión, C: Curva de ROC, D: Grafico de coordenadas paralelas.

Inicialmente en la gráfica de dispersión es posible ver lo separado que se encuentran aquí ambas muestras de datos y como se apoderan de mayor superficie del plano, en cuanto a la matriz de confusión y curva ROC permanecen igual que los anteriores entrenamientos, al final dentro del gráfico de coordinadas paralelas se visualiza una variación en el camino que siguen los datos de mal estado, aun así, fueron clasificados adecuadamente.

3.3 PROGRAMACIÓN PARA PROCESAMIENTO DE DATOS Y PREDICCIÓN.

La programación final obtenida es capaz de procesar desde una hoja de Excel los datos obtenidos de las vibraciones del motor y genera una tabla que contiene las 10 medidas estadísticas (apartado 2.3.1.1) usadas como características necesarias para el algoritmo de clasificación. El algoritmo de clasificación se entrena cada vez que se corra el programa y realiza la clasificación pertinente con la tabla de datos obtenida del procesamiento previo.

3.3.1 PROCESAMIENTO DE DATOS.

El procesamiento de datos es la primera parte de la programación en donde se genera la tabla necesaria para el algoritmo de clasificación, esta tabla contiene las características ya establecidas en el entrenamiento, pero no su etiqueta (estado), cada hoja de Excel procesada genera una tabla con el nombre Tabla 1, si se procesa más datos se mantendrá el nombre, pero cambiaran sus valores a los correspondientes del nuevo proceso.

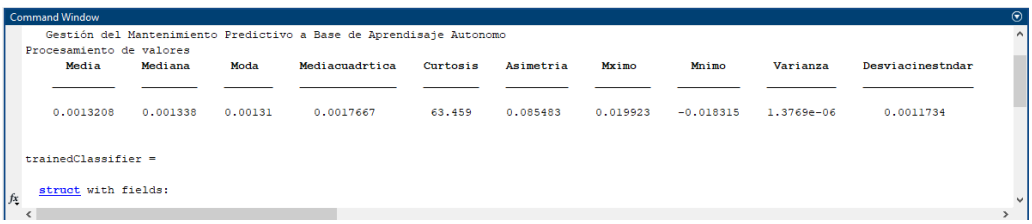


Figura 3.4. Tabla producto del procesamiento de datos dentro del Workspeace

Tabla1

1x10 table

	1 Media	2 Mediana	3 Moda	4 Mediacuadrtica	5 Curtosis	6 Asimetria	7 Mximo	8 Mnimo	9 Varianza	10 Desviacinestndar
1	0.0013	0.0013	0.0013	0.0018	63.4591	0.0855	0.0199	-0.0183	1.3769e-06	0.0012
2										

Figura 3.5. Tabla producto del procesamiento de datos.

Las figuras 3.4 y 3.5 muestran el resultado obtenido tras el procesamiento de datos efectuado por la programación de MATLAB®.

3.3.2 REENTRENAMIENTO DEL ALGORITMO.

El reentrenamiento del algoritmo es necesario para evitar el entrenamiento por medio de la aplicación de Machine Learning, de esta forma únicamente se corre la programación de forma directa. La programación requiere una tabla con un formato igual a la usada en el entrenamiento inicial, dándonos como resultado una eficiencia del entrenamiento y una ecuación necesaria para la predicción de la nueva tabla de datos.

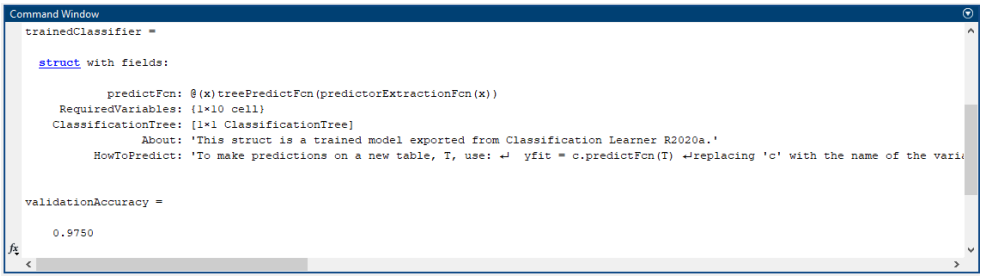


Figura 3.6. Resultado obtenido del reentrenamiento del algoritmo de clasificación.

3.3.3 PREDICCIÓN DE LA TABLA DE DATOS PROCESADA.

La tercera parte de la programación se encarga de clasificar la tabla de datos creada en el procesamiento inicial, después del reentrenamiento se genera una ecuación de predicción la cual se usa con la tabla generada y esto dará como resultado el estado real sobre el funcionamiento del motor.

Al ser una clasificación binaria únicamente se puede conocer 2 estados del motor, con falla (ME) y en óptimas condiciones (BE). A continuación, se muestra la ecuación producida después del entrenamiento y que será útil para realizar nuevas predicciones.

$$\text{yfit} = \text{trainedClassifier.predictFcn(T)} \quad (10)$$

Donde:

Yfit= nueva predicción.

trainedClassifier= modelo entrenado de clasificación.

predictFcn= función del modelo de predicción.

T= nueva tabla de datos a clasificar.

Inicialmente dentro de los entrenamientos que trabajan únicamente dentro de una clasificación binaria, es decir BE y ME, las opciones de los resultados son 2.

Resultado de la clasificación de un grupo de datos perteneciente a vibraciones del motor en buen estado, como se indica en la figura 3.7 una vez ejecutada la ecuación de predicción, esta se encarga de clasificar la nueva muestra según la categoría en la que se encuentren sus características, para este caso la etiqueta corresponde a BE.

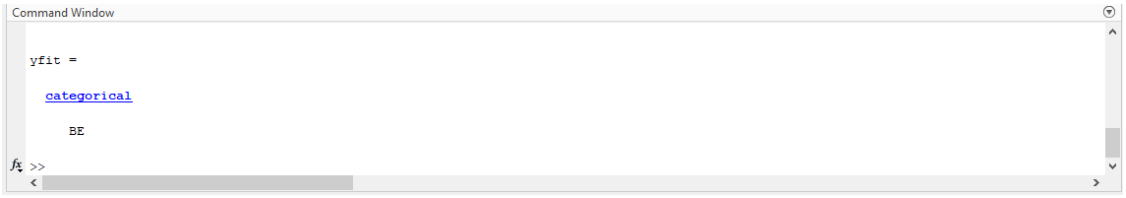


Figura 3.7. Resultado obtenido de una clasificación perteneciente a buen estado.

La figura 3.8 es el resultado de la clasificación de un grupo de datos perteneciente a vibraciones del motor en mal estado, es decir, tras el procesamiento de datos, las características que estas mostraron son pertenecientes a las que se encuentran dentro de la categoría de ME.

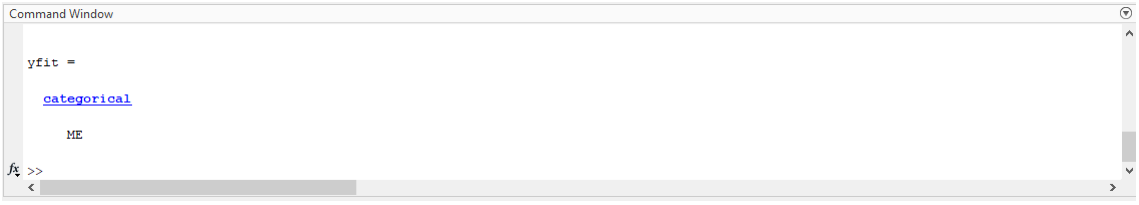


Figura 3.8. Resultado obtenido de una clasificación perteneciente a mal estado.

3.3.4 GRÁFICA DE LA VIBRACIÓN CAPTADA.

Como ayuda visual se genera un gráfico con los datos pertenecientes a las vibraciones, aquí es posible apreciar la forma de cada una y los intervalos de valores en los que se encuentra.

Gráfico BE.

El gráfico (Figura 3.9) de las vibraciones pertenecientes a los datos de buen estado muestra una simetría y homogeneidad a lo largo de su trayectoria, conteniendo como pico máximo un valor de 0.021 v y en su pico inferior un valor mínimo de -0.017 v, si se toma en cuenta que el origen de su amplitud empieza en el punto 0.0013v como punto medio de la vibración el gráfico se distribuye de forma equitativa en ambas direcciones. En la mayoría de los gráficos pertenecientes a la categoría de buen estado se da esta forma, variando levemente unos de otros.

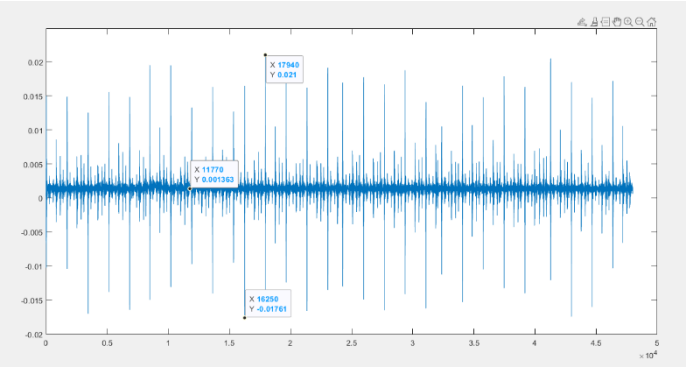


Figura 3.9. Vibraciones obtenidas del motor en estado óptimo de funcionamiento.

Gráfico ME falla 1.

La falla número 1 produce una gráfica (figura 3.10) de la siguiente forma, es apreciable que la cantidad de vibraciones captadas han aumentado y en ciertos puntos la distribución de oscilaciones es desproporcionada, además los picos máximo y mínimo han cambiado siendo aquí el valor máximo captado 0.013V y el valor mínimo -0-009, si el punto de origen de las oscilaciones permanece en 0.0013 indica una desproporción entre ambos sentidos.

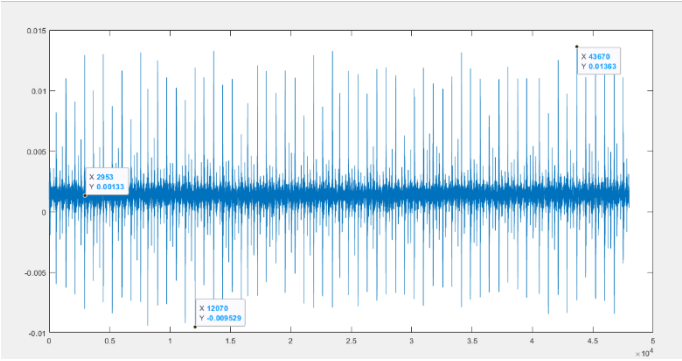


Figura 3.10. Vibraciones obtenidas del motor al simular la primera falla.

Gráfico ME falla 2.

La falla número 2 al ser más severa que la número 1 muestra una desproporción entre oscilaciones produciendo una gráfica (figura 3.11) completamente asimétrica, la cantidad de vibraciones también han aumentado y sus valores de picos máximo y mínimo han variado de igual forma, el punto de origen de las oscilaciones aquí es de 0.0015v y de aquí el valor máximo registrado sube hasta 0,026v, su valor mínimo registra -0.025 v.

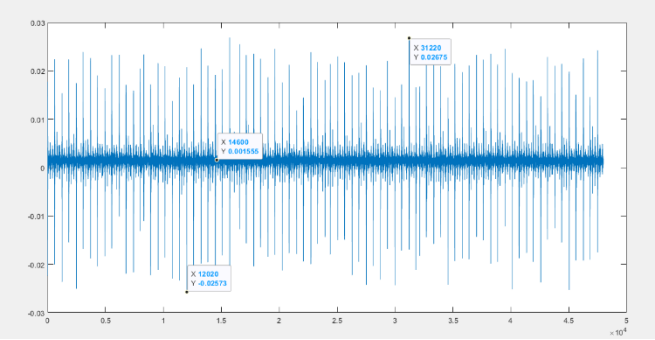


Figura 3.11. Vibraciones obtenidas del motor al simular la segunda falla.

Gráfico ME falla 3.

Dentro de la falla número 3 existe una variación, a pesar de que se tomó las muestras bajo el mismo tiempo que los anteriores estados aquí hay menos datos, el gráfico (figura 3.12) contiene en su zona centran mayor cantidad de vibraciones y cierta simetría en el resto de su forma. Esta falla es la más severa simulada y aun que aquí solo se nota una diferencia en la zona central respecto a la gráfica de buen estado sus picos máximo y mínimo si son diferentes, su valor máximo alcanzado es de 0.023v, su valor mínimo registrado es de -0.019 v y su origen se da en el punto 0.014.

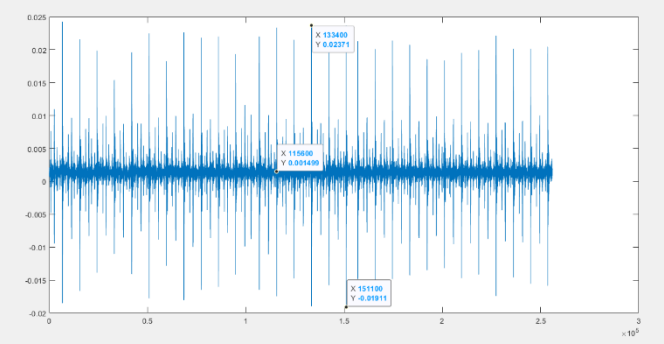


Figura 3.12. Vibraciones obtenidas del motor al simular la tercera falla.

3.4 VALIDACIÓN DEL APRENDIZAJE AUTÓNOMO.

El entrenamiento en las 3 fallas creadas da una eficiencia del 97.5%, es decir, el algoritmo tiene dificultad en clasificar 1 de cada 40 pruebas. Para verificar este porcentaje se evalúa cada entrenamiento usando 40 (20 en BE y 20 en ME) muestras de datos que no fueron usadas inicialmente, muestras de datos nuevas.

Tabla 3.5. Resultado de la validación de los 3 entrenamientos.

Prueba.	Entrenamiento número:	Categoría	Clasificaciones correctas.	Clasificaciones incorrectas.	Validez.	Validez por entrenamiento.
1	1	BE	19	1	95%	85%
2	1	ME	15	5	75%	
3	2	BE	19	1	95%	97.5%
4	2	ME	20	0	100%	
5	3	BE	20	0	100%	97.5%
6	3	ME	19	1	95%	

La tabla 3.5 muestra los resultados de las pruebas realizadas a los entrenamientos pertenecientes a cada falla, el primer entrenamiento se simuló una falla leve la cual no generaba mucha diferencia entre los datos de BE y ME, a pesar de que la valides inicial es de 97.5% aquí se puede observar que dicho porcentaje baja al 85%, en su mayoría el algoritmo tiene problemas en clasificar los datos de mal estado y es aceptable, ya que no es una falla con muchas variaciones respecto al buen estado del motor. Las pruebas realizadas a los entrenamientos pertenecientes a las fallas 2 y 3 corroboran el porcentaje de validez dado inicialmente, el algoritmo tiene dificultad en clasificar 1 de las 40 muestras, esto es más acertado, ya que las fallas son más severas y reales, la diferencia entre valores (BE Y ME) es notoria.

3.5 COMPILACIÓN DE LAS 3 FALLAS EN UN MISMO ENTRENAMIENTO.

Si bien el entrenamiento individual de cada falla contiene porcentajes de eficacia altos, no son la mejor alternativa dentro de un diagnóstico, el técnico tardaría demasiado tiempo verificando falla por falla, por lo que es necesario una compilación total de todas las fallas en una sola tabla de datos. Los entrenamientos iniciales únicamente se entrenaron con 2 categorías, buen estado (BE) y mal estado (ME), si el número de fallas que se registre en la tabla de entrenamiento sube, las categorías subirán de igual forma, teniendo así una predicción eficiente y rápida en un solo entrenamiento.

Dentro de esta investigación se trabajó con datos de 4 estados del motor, buen estado (BE) con el motor en óptimas condiciones de funcionamiento, mal estado de la falla 1 (MEF1) para la primera falla que se simuló, mal estado de la falla 2 (MEF2) para los datos pertenecientes a la falla número 2 y mal estado de la falla 3 (MEF3) para los datos receptados al simular la tercera falla. Para este entrenamiento y respectivas predicciones las categorías suben entonces a 4 y la cantidad de muestras a 80.

Tabla 3.6. Características y etiquetas de la compilación de fallas.

Características										Etiquetas
Media	Mediana	Moda	Media cuadrática	Curtosis	Asimetría	Máximo	Mínimo	Varianza	Desviación estándar	Estado.
-	-	-	-	-	-	-	-	-	-	BE
-	-	-	-	-	-	-	-	-	-	MEF1
-	-	-	-	-	-	-	-	-	-	MEF2
-	-	-	-	-	-	-	-	-	-	MEF3

3.5.1 EFICIENCIA DEL ENTRENAMIENTO.

Entre mayor sea la cantidad de datos la eficiencia del entrenamiento debe ser mejor, aquí hay 4 diferentes estados los cuales cuentan cada uno como una diferente etiqueta. El algoritmo de clasificación no tiene dificultan en diferenciar los datos de cada categoría e inicialmente se obtiene un resultado de eficiencia del 95%, el algoritmo tiene dificultad en clasificar 4 de las 80 muestras, 1 perteneciente a la falla 1, 1 de la falla 2 y 2 a la falla 3.

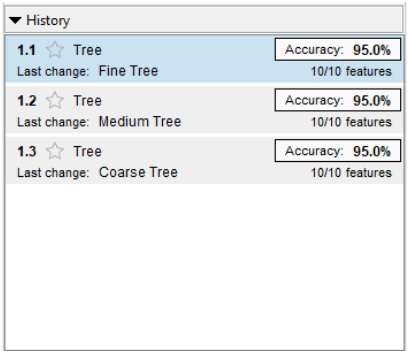


Figura 3.13. Resultado de la eficiencia del entrenamiento.

3.5.2 DIAGRAMA DE DISPERSIÓN.

Los datos ingresados se dispersan dentro de la gráfica en grupos por colores, siendo los puntos cada muestra clasificada de forma correcta y las X las muestras donde el algoritmo tuvo dificultad. Los colores pertenecen; el azul para los datos de buen estado (BE), el anaranjado para la falla 1, el amarillo para la falla 2 y el morado para la falla 3.

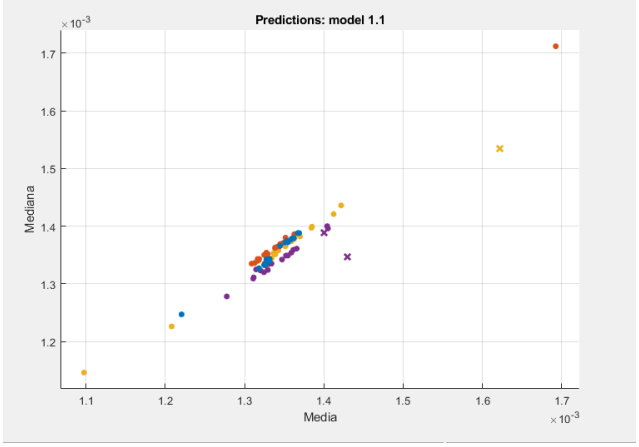


Figura 3.14. Diagrama de dispersión del entrenamiento.

3.5.3 MATRIZ DE CONFUSIÓN.

La matriz de confusión expresa de forma gráfica donde exactamente existe una confusión de parte del algoritmo al realizar la clasificación de datos. Cada cruce entre 2 categorías de diferente denominación son una incorrecta clasificación de parte del algoritmo, para este caso se tiene 4 fallas de las 80 muestras, corrobora el resultado del 95% de eficiencia.

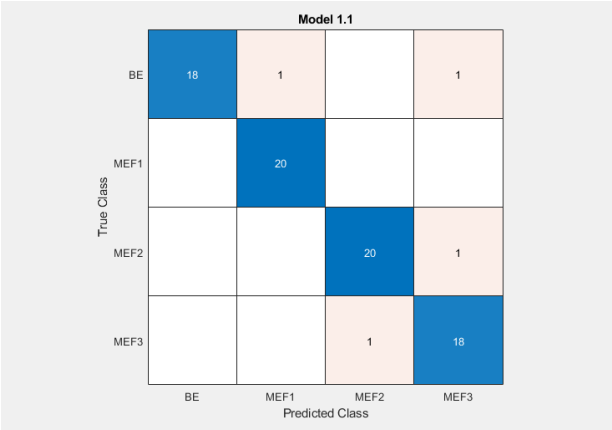


Figura 3.15. Matriz de confusión

3.5.4 CURVA ROC.

La curva ROC es una gráfica (Figura 3.16) dentro de un plano que expresa la variación de la eficiencia obtenida por el entrenamiento siendo las coordenadas 1:1 el valor equivalente a un 100%, aquí se ve inicialmente que se acerca al 0:0.9 y luego sube al 0:0.95, de igual forma corrobora la eficiencia obtenida del 95%.

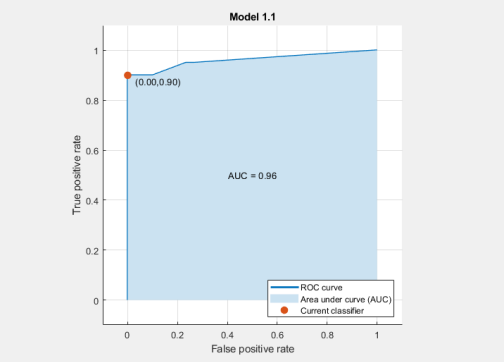


Figura 3.16. Curva ROC

3.5.5 GRÁFICA DE COORDENADAS PARALELAS.

La matriz de coordenadas paralelas indica el camino de cada muestra de datos a través de sus características, es apreciable como cada categoría sigue el mismo camino y varía notoriamente de las otras. Las líneas continuas son las que si han sido clasificadas correctamente mientras que las entrecortadas pertenecen a las muestras que crearon confusión dentro del algoritmo.

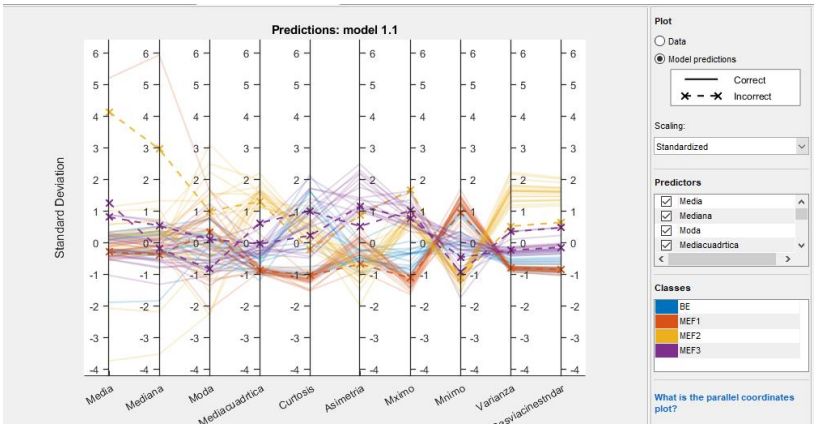


Figura 3.17. Grafica de coordenadas paralelas.

3.5.6 REENTRENAMIENTO

Es necesaria la tabla inicial para un reentrenamiento y respectivo uso del aprendizaje, aquí ya el porcentaje de eficiencia dado sube hasta el 96.25%, y es posible realizar las predicciones a nuevas muestras de datos ya entre las 4 categorías existentes dentro de la base de datos.

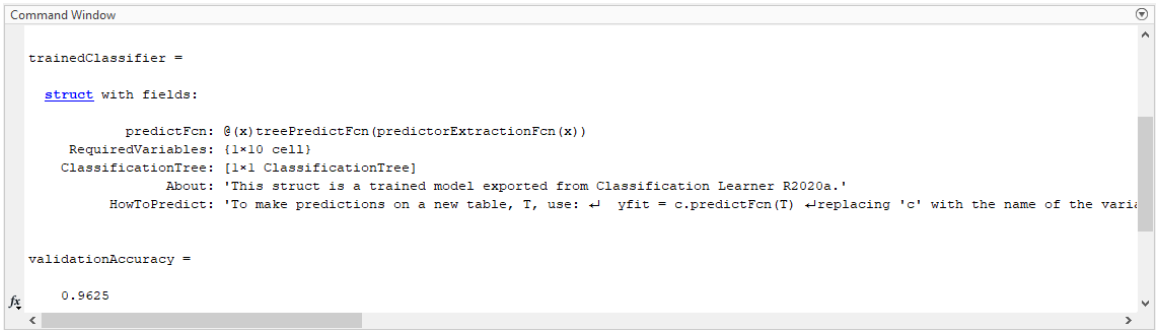


Figura 3.18. Resultado del reentrenamiento de la compilación de fallas.

3.6 VALIDACIÓN DEL ALGORITMO DE CLASIFICACIÓN CON LOS 4 ESTADOS DEL MOTOR.

Es necesario verificar los valores obtenidos y validarlos mediante pruebas con nuevos datos, se usan 20 muestras para cada estado y de esta forma validar de forma individual y colectiva la clasificación que el algoritmo realice. La tabla 3.7 contiene los resultados de las pruebas de cada estado, siendo las pruebas 1, 2 y 3 las que mejor eficiencia muestran y únicamente la prueba 2 es la que presenta una eficacia del 75%, de todas formas, la validez del entrenamiento colectivo es superior al 90%.

Tabla 3.7. Resultados de la validación del entrenamiento con los 4 estados del motor.

Prueba.	Categoría	Muestras	Clasificaciones correctas.	Clasificaciones incorrectas.	Validez.	Validez del entrenamiento.
1	BE	20	20	0	100%	92.5%
2	MEF1	20	15	5	75%	
3	MEF2	20	20	0	100%	
4	MEF3	20	19	1	95%	

La figura 3.19 es el resultado de una clasificación realizada a una muestra de datos de vibraciones perteneciente a un buen estado del motor, es decir, este resultado será observable cada vez que se procese y clasifique datos con estas características.

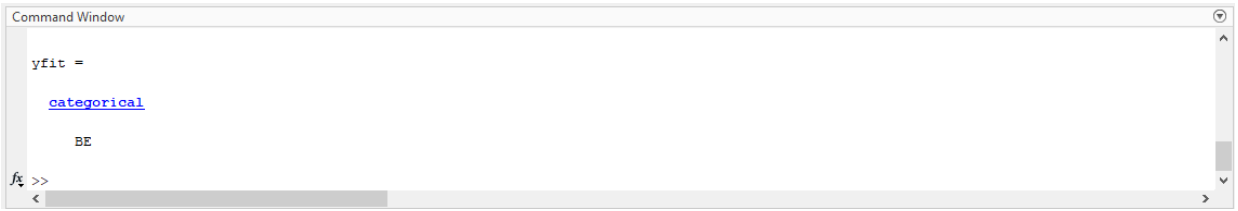


Figura 3.19. Resultado de una clasificación de datos en buen estado.

La figura 3.20 corresponde al resultado de la clasificación realizada por medio del algoritmo de aprendizaje autónomo y muestra como resultado una categoría de MEF1, es decir, pertenece a un mal estado del motor, específicamente a la falla simulada número 1.

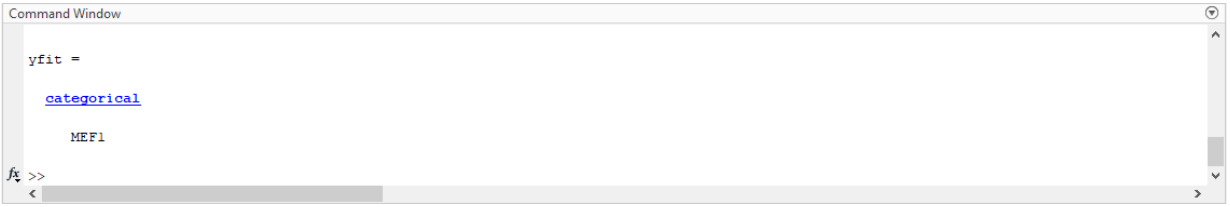


Figura 3.20. Resultado de una clasificación de datos en mal estado perteneciente a la primera falla simulada.

La figura 3.21 pertenece a la clasificación de una muestra de datos perteneciente a un mal estado del motor, producido por la falla simulada número 2, el resultado nos lo expresa claramente como MEF2.

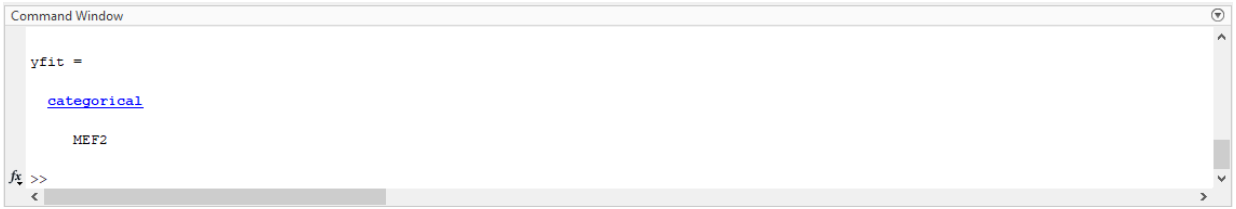


Figura 3.21. Resultado de una clasificación de datos en mal estado perteneciente a la segunda falla simulada.

La figura 3.22 es la correspondiente a la última prueba en cuanto a la validación del entrenamiento, corresponde a una clasificación que dio como resultado la categoría MEF3, esto significa un mal estado del motor producido por la falla simulada número 3.

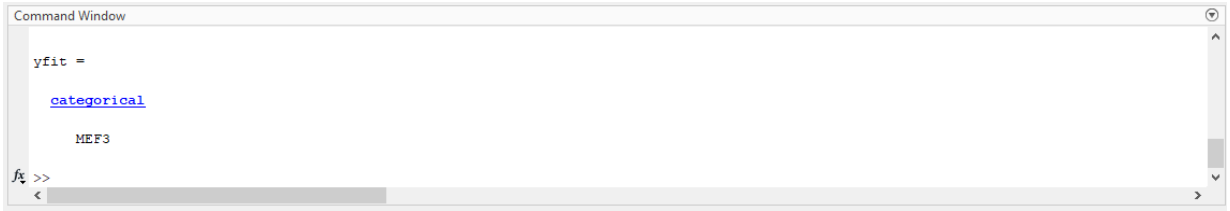


Figura 3.22. Resultado de una clasificación de datos en mal estado perteneciente a la tercera falla simulada.

Para cada diferente estado el algoritmo lo clasifica correctamente dependiendo de sus características, es necesario únicamente introducir el nombre del archivo Excel al cual se adjuntó los datos captados y el programa se encarga automáticamente de realizar el trabajo, tanto el procesamiento de datos como la predicción que es la que se necesita para la ejecución de un mantenimiento predictivo.

CAPÍTULO IV

4.CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

Los entrenamientos individuales realizados que simulan 3 fallas del motor han dado como resultado una eficiencia del 97.5% en cada caso, es decir 1 de cada 40 muestras de datos no será correctamente clasificada, al validar cada entrenamiento individualmente es notorio que este porcentaje varía en la primera falla donde se simuló un problema de inyectores leve, bajando aquí la eficiencia hasta un 85%, dentro de las otras 2 fallas las cuales fueron más severas si se conserva el porcentaje inicialmente dado del 97.5%. Entre mayor sea la cantidad de datos dentro de la tabla de entrenamiento mejor será su validación, dentro del entrenamiento final se hizo una compilación de todos los estados dando como resultado una eficiencia inicial del 95%, la cual al ser validada se redujo al 92.5%, aun así, es un porcentaje confiable y capaz de ser usado dentro de un diagnóstico real del automotor.

Las gráficas pertenecientes a las vibraciones de cada categoría son una ayuda visual complementaria que expresa de forma correcta al estado al que representan, es observable dentro de la categoría de BE las oscilaciones producidas son simétricas y homogéneas a lo largo de su recorrido, mientras tanto en las gráficas de vibraciones pertenecientes a las fallas simuladas es apreciable que dicha simetría se ha perdido y además los valores entre picos máximos y mínimos se han alterado llegando hasta los 0,025 y -0,015 voltios en la falla más severa, en comparación a los de buen estado 0,021 y -0,017 se ve un cambio notorio dentro de la amplitud de las oscilaciones.

Para una correcta gestión dentro del mantenimiento vehicular es necesario una compilación de todos los estados del motor dentro de una misma tabla, esto permite un diagnóstico directo y rápido usando una sola ecuación de predicción para conocer instantáneamente el estado real en el que se encuentra el motor dentro de las fallas registradas, mediante esta investigación se comprobó mediante los 4 estados usados que el algoritmo los identifica correctamente cada uno, con un porcentaje superior al 85% dentro de la primera falla (MEF1) y una eficiencia superior al 95% para las categorías de BE, MEF2 y MEF3.

El mantenimiento predictivo es posible gracias al algoritmo de clasificación empleado, al introducir nuevos datos no necesariamente deben ser iguales a los ya usados dentro del entrenamiento, simplemente el algoritmo lo clasificará dentro del grupo al que más se asemeje, por esta cualidad si el motor se acerca más a un estado de fallo que al de óptimas condiciones instantáneamente se puede predecir una falla y tomar acciones antes de que suceda, programando así un mantenimiento basando en el estado real del automotor, ahorrando tiempo y dinero. La eficiencia captada mayor a un 90% en todas las pruebas es suficiente base como para confiar en este nuevo método de diagnóstico.

4.2 REOMENDACIONES

Es recomendable determinar que componentes del automotor producen vibraciones medibles y capaces de proporcionar información del estado real del vehículo, como referencia cada mecanismo que use movimientos rotacionales es una alternativa, siendo el tren motriz la mejor opción en cuanto al empleo de este tipo de diagnóstico.

Es importante tener una biblioteca de datos correspondientes a diferentes fallas, registrar nuevas fallas y adjuntarlas a la tabla de entrenamiento mejorará el diagnóstico del vehículo siendo el técnico capaz de identificar anomalías únicamente con una muestra de datos de vibraciones de 8 segundos, mejorando así la gestión dentro del mantenimiento.

Dentro de cada entrenamiento se debe caracterizar y etiquetar bien cada muestra de datos, si no se hace esto correctamente es posible crear confusión dentro del algoritmo de clasificación y tener predicciones erróneas, además, cada vez que se adjunte un nuevo estado a la tabla de entrenamiento es necesario reentrenar otro algoritmo que contenga esta nueva característica y exista dentro de la función de clasificación.

BIBLIOGRAFÍA

Abbas, A. K., Al-haideri, N. A., & Bashikh, A. A. (2019). Implementing artificial neural networks and support vector machines to predict lost circulation. *Egyptian Journal of Petroleum*, *28*(4), 339–347. https://doi.org/10.1016/j.ejpe.2019.06.006

Alea V. & Jimenes E. (2015). *Estadística I (Ade): Teoría Y Ejercicios*. 5. http://diposit.ub.edu/dspace/bitstream/2445/66107/1/EstadisticaI_2016.pdf

Ali, M., Xianjun, H., Abdelkareem, M., Gulzar, M., & Elsheikh, A. H. (2018). Novel approach of the graphene nanolubricant for energy saving via anti-friction/wear in automobile engines. *Tribology International*, *124*, 209–229. https://doi.org/10.1016/j.triboint.2018.04.004

Alvarez, F. (2012). Métodos Estadísticos Aplicados a las Auditorías Sociolaborales. *Universidad de Cordoba*, 1–47. http://www.uco.es/zootecniaygestion/img/pictorex/27_12_49_7.pdf

Amihai, I., Gitzel, R., Kotriwala, A. M., Pareschi, D., Subbiah, S., & Sosale, G. (2018). An industrial case study using vibration data and machine learning to predict asset health. *Proceeding - 2018 20th IEEE International Conference on Business Informatics, CBI 2018*, *1*, 178–185. https://doi.org/10.1109/CBI.2018.00028

Anastasi, S., Madonna, M., & Monica, L. (2021). Implications of embedded artificial intelligence - Machine learning on safety of machinery. *Procedia Computer Science*, *180*(2019), 338–343. https://doi.org/10.1016/j.procs.2021.01.171

Batista, R. M., Tarragó, J. C. P., Herrera, H. C. T., & López, O. B. A. (2011). Vibraciones Mecanicas. In *Vibraciones Mecanicas Volumen 1*.

Brusco, P. (2017). Machine learning ¿El fin de la programación? *Machine Learning*, *45*(13), 40–48.

Carvalho, T., Soares, F., Vita, R., & Bastos, J. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. *Computers and Industrial Engineering*, *137*(August), 106024. https://doi.org/10.1016/j.cie.2019.106024

Farag, O., Selim, M., & Emam, S. (2017). Time and frequency analyses of dual-fuel engine block vibration. *Fuel*, *203*, 884–893. https://doi.org/10.1016/j.fuel.2017.05.034

Gao, K., Mei, G., Piccialli, F., Cuomo, S., Tu, J., & Huo, Z. (2020). Julia language in machine learning: Algorithms, applications, and open issues. *Computer Science*

Review, *37*, 100254. https://doi.org/10.1016/j.cosrev.2020.100254

García, A. P., Álgebra, D., & Estudios, F. De. (2014). *Las medias estadísticas : ejercicios motivadores Venancio Tomeo Perucha*.

Jardine, A. K. S., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

Kanawaday, A., & Sane, A. (2018). Machine learning for predictive maintenance of industrial machines using IoT sensor data. *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, *2017-Novem*(Figure 2), 87–90. https://doi.org/10.1109/ICSESS.2017.8342870

Kimera, D., & Nangolo, F. N. (2020). Predictive maintenance for ballast pumps on ship repair yards via machine learning. *Transportation Engineering*, *2*(July), 100020. https://doi.org/10.1016/j.treng.2020.100020

Masterhacks.net. (2013). *Manual Básico De Programación En*. https://masterhacks.net/

Mathew, V., Toby, T., Singh, V., Rao, B. M., & Kumar, M. G. (2018). Prediction of Remaining Useful Lifetime (RUL) of turbofan engine using machine learning. *IEEE International Conference on Circuits and Systems, ICCS 2017*, *2018-Janua*(Iccs), 306–311. https://doi.org/10.1109/ICCS1.2017.8326010

MathWorks, & MATLAB. (2020a). *Cluster data by training a self-organizing maps network - MATLAB*. Neural Net Clustering. https://www.mathworks.com/help/deeplearning/ref/neuralnetclustering-app.html

MathWorks, & MATLAB. (2020b). *Design and run experiments to train and compare deep learning networks - MATLAB*. Experiment Manager. https://www.mathworks.com/help/deeplearning/ref/experimentmanager-app.html

MathWorks, & MATLAB. (2020c). *Fit data by training a two-layer feed-forward network - MATLAB*. Neural Net Fitting. https://www.mathworks.com/help/deeplearning/ref/neuralnetfitting-app.html

MathWorks, & MATLAB. (2020d). *Get Started with Deep Network Designer - MATLAB & Simulink*. Get Started with Deep Network Designer. https://www.mathworks.com/help/deeplearning/gs/get-started-with-deep-network-designer.html

MathWorks, & MATLAB. (2020e). *Regression Learner App - MATLAB & Simulink*. Regression Learner App. https://www.mathworks.com/help/stats/regression-learner-

app.html

MathWorks, & MATLAB. (2020f). *Solve a nonlinear time series problem by training a dynamic neural network - MATLAB*. Neural Net Time Series. https://www.mathworks.com/help/deeplearning/ref/neuralnettimeseries-app.html

MathWorks, & MATLAB. (2020g). *Train models to classify data using supervised machine learning - MATLAB*. Classification Learner. https://www.mathworks.com/help/stats/classificationlearner-app.html

Montero Jimenez, J. J., Schwartz, S., Vingerhoeds, R., Grabot, B., & Salaün, M. (2020). Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics. *Journal of Manufacturing Systems*, *56*(July), 539–557. https://doi.org/10.1016/j.jmsy.2020.07.008

Paolanti, M., Romeo, L., Felicetti, A., Mancini, A., Frontoni, E., & Loncarski, J. (2018). Machine Learning approach for Predictive Maintenance in Industry 4.0. *2018 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, MESA 2018*, 1–6. https://doi.org/10.1109/MESA.2018.8449150

Praveenkumar, T., Saimurugan, M., Krishnakumar, P., & Ramachandran, K. I. (2014). Fault diagnosis of automobile gearbox based on machine learning techniques. *Procedia Engineering*, *97*, 2092–2098. https://doi.org/10.1016/j.proeng.2014.12.452

Ramiro, A., & Alvarado, V. (2020). *Machine Learning para Todos Aldo Valdez Alvarado*. *January 2019*. https://doi.org/10.13140/RG.2.2.13786.70086

Roncancio, H., & Cifuentes, H. (2000). Tutorial de LABVIEW. *Labview Tutorial*, *1*, 20. http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Tutorial+de+labview#0%5Cnhttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Tutorial+de+LABVIEW#0

Salazar, C. (2018). *Fundamentos Básicos De Estadística*.

Sandoval, L. (2018). *Algoritmos de aprendizaje automático para análisis y predicción de datos*. 36–40.

Santos, F., Tschiptschin, A., & Goldenstein, H. (2018). Effects of ethanol content on cast iron cylinder wear in a flex-fuel internal combustion engine–A case study. *Wear*, *406–407*, 105–117. https://doi.org/10.1016/j.wear.2018.04.003

Secretaría Nacional de Planificación y Desarrollo. (2017). *Plan Nacional de Desarrollo 2017-2021-Toda una Vida*. 84. http://www.planificacion.gob.ec/wp-content/uploads/downloads/2017/10/PNBV-26-OCT-FINAL_0K.compressed1.pdf

Servicio Informatico de Sistemas. (2020). *Manual Básico De Matlab*.

Siles, H. (2012). *"Mantenimiento De Vehículos."* 8.

Taghizadeh-Alisaraei, A., & Mahdavian, A. (2019). Fault detection of injectors in diesel engines using vibration time-frequency analysis. *Applied Acoustics*, *143*, 48–58. https://doi.org/10.1016/j.apacoust.2018.09.002

Xu, Y., Zhou, Y., Sekula, P., & Ding, L. (2021). Machine learning in construction: From shallow to deep learning. *Developments in the Built Environment*, *6*(February), 100045. https://doi.org/10.1016/j.dibe.2021.100045

Zhao, Z., Xu, X., Yang, J., Chang, L., Yan, X., Wang, G., & Xu, X. (2020). Machine learning-based wear fault diagnosis for marine diesel engine by fusing multiple data-driven models. *Knowledge-Based Systems*, *190*. https://doi.org/10.1016/j.knosys.2019.105324

ANEXOS.

Anexo 1. Sensor de vibraciones correctamente colocado en el bloque motor.

Anexo 2. Motor del tractor agrícola marca Internacional.

Anexo 3. Simulación de fallas por medio dele tornillo de reglaje de los inyectores.

Anexo 4. Programación de Matlab para el procesamiento de datos de vibraciones.

```
clc;

disp('                    Universidad Técnica del Norte');
disp('               Ingeniería en Mantenimiento Automotriz');
disp('                          Trabajo de Grado');
disp('   Gestión del Mantenimiento Predictivo a Base de Aprendisaje
Autonomo');

Vibraciones=xlsread("ME2-PS-800-5"); %Ingrese el nombre entre las
comillas de la tabla que contiene los datos a procesar.
Datos=[Vibraciones];

Media=mean(Datos);
Mximo=max(Datos);
Mnimo=min(Datos);
Mediana=median(Datos);
Desviacinestndar=std(Datos);
Varianza=var(Datos);
Curtosis=kurtosis(Datos);
Moda=mode(Datos);
Asimetria=skewness(Datos);
Dc=(Datos).^2;
Mdc=mean(Dc);
Mediacuadrtica=((Mdc).^(1/2));
Tabla1=table(Media,Mediana,Moda,Mediacuadrtica,Curtosis,Asimetria,Mximo,M
nimo,Varianza,Desviacinestndar);
disp("Procesamiento de valores");
disp(Tabla1);
plot(Datos);
```

Anexo 5. Función de Machine Learning producto del entrenamiento del algoritmo de clasificación.

```
function [trainedClassifier, validationAccuracy] =
trainClassifier(Prosdat12)
% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData)
% Returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
%  Input:
%      trainingData: A table containing the same predictor and response
%       columns as those imported into the app.
%
%  Output:
%      trainedClassifier: A struct containing the trained classifier. The
%       struct contains various fields with information about the trained
%       classifier.
%
%      trainedClassifier.predictFcn: A function to make predictions on
new
%       data.
%
%      validationAccuracy: A double containing the accuracy in percent.
In
```

```
%       the app, the History list displays this overall accuracy score
for
%       each model.
%
% Use the code to train the model with new data. To retrain your
% classifier, call the function from the command line with your original
% data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
%   [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data
T2,
% use
%   yfit = trainedClassifier.predictFcn(T2)
%
% T2 must be a table containing at least the same predictor columns as
used
% during training. For details, enter:
%   trainedClassifier.HowToPredict

% Auto-generated by MATLAB on 08-Jul-2021 11:43:34


% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = Prosdat12;
predictorNames = {'Media', 'Mediana', 'Moda', 'Mediacuadrtica',
'Curtosis', 'Asimetria', 'Mximo', 'Mnimo', 'Varianza',
'Desviacinestndar'};
predictors = inputTable(:, predictorNames);
response = inputTable.Estado;
isCategoricalPredictor = [false, false, false, false, false, false,
false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the
classifier.
classificationTree = fitctree(...
    predictors, ...
    response, ...
    'SplitCriterion', 'gdi', ...
    'MaxNumSplits', 100, ...
    'Surrogate', 'off', ...
    'ClassNames', categorical({'BE'; 'MEF1'; 'MEF2'; 'MEF3'}, {'BE' 'ME'
'MEF1' 'MEF2' 'MEF3'}));

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
trainedClassifier.predictFcn = @(x)
treePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Asimetria', 'Curtosis',
'Desviacinestndar', 'Media', 'Mediacuadrtica', 'Mediana', 'Mnimo',
'Moda', 'Mximo', 'Varianza'};
```

```
trainedClassifier.ClassificationTree = classificationTree;
trainedClassifier.About = 'This struct is a trained model exported from
Classification Learner R2020a.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new
table, T, use: \n  yfit = c.predictFcn(T) \nreplacing ''c'' with the name
of the variable that is this struct, e.g. ''trainedModel''. \n \nThe
table, T, must contain the variables returned by: \n  c.RequiredVariables
\nVariable formats (e.g. matrix/vector, datatype) must match the original
training data. \nAdditional variables are ignored. \n \nFor more
information, see <a href="matlab:helpview(fullfile(docroot, ''stats'',
''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to
predict using an exported model</a>.');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = Prosdat12;
predictorNames = {'Media', 'Mediana', 'Moda', 'Mediacuadrtica',
'Curtosis', 'Asimetria', 'Mximo', 'Mnimo', 'Varianza',
'Desviacinestndar'};
predictors = inputTable(:, predictorNames);
response = inputTable.Estado;
isCategoricalPredictor = [false, false, false, false, false, false,
false, false, false, false];

% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationTree,
'KFold', 7);

% Compute validation predictions
[validationPredictions, validationScores] =
kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');
```

Anexo 6. Programación final que permite la predicción de nuevos datos de forma instantánea.

```
clc;

disp('                    Universidad Técnica del Norte');
disp('               Ingeniería en Mantenimiento Automotriz');
disp('                          Trabajo de Grado');
disp('   Gestión del Mantenimiento Predictivo a Base de Aprendisaje
Autonomo');

Vibraciones=xlsread("ME2-PS-800-5"); %Ingrese el nombre entre las
comillas de la tabla que contiene los datos a procesar.
Datos=[Vibraciones];

Media=mean(Datos);
Mximo=max(Datos);
Mnimo=min(Datos);
Mediana=median(Datos);
Desviacinestndar=std(Datos);
Varianza=var(Datos);
```

```
Curtosis=kurtosis(Datos);
Moda=mode(Datos);
Asimetria=skewness(Datos);
Dc=(Datos).^2;
Mdc=mean(Dc);
Mediacuadrtica=((Mdc).^(1/2));
Tabla1=table(Media,Mediana,Moda,Mediacuadrtica,Curtosis,Asimetria,Mximo,M
nimo,Varianza,Desviacinestndar);
disp("Procesamiento de valores");
disp(Tabla1);
plot(Datos);

[trainedClassifier, validationAccuracy] = trainClassifier(Prosdat12)
% [trainedClassifier, validationAccuracy] = trainClassifier(trainingData)
% Returns a trained classifier and its accuracy. This code recreates the
% classification model trained in Classification Learner app. Use the
% generated code to automate training the same model with new data, or to
% learn how to programmatically train models.
%
%  Input:
%      trainingData: A table containing the same predictor and response
%       columns as those imported into the app.
%
%  Output:
%      trainedClassifier: A struct containing the trained classifier. The
%       struct contains various fields with information about the trained
%       classifier.
%
%      trainedClassifier.predictFcn: A function to make predictions on
new
%       data.
%
%      validationAccuracy: A double containing the accuracy in percent.
In
%       the app, the History list displays this overall accuracy score
for
%       each model.
%
% Use the code to train the model with new data. To retrain your
% classifier, call the function from the command line with your original
% data or new data as the input argument trainingData.
%
% For example, to retrain a classifier trained with the original data set
% T, enter:
%   [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% To make predictions with the returned 'trainedClassifier' on new data
T2,
% use
%   yfit = trainedClassifier.predictFcn(T2)
%
% T2 must be a table containing at least the same predictor columns as
used
% during training. For details, enter:
%   trainedClassifier.HowToPredict

% Auto-generated by MATLAB on 08-Jul-2021 11:43:34
```

```
% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = Prosdat12;
predictorNames = {'Media', 'Mediana', 'Moda', 'Mediacuadrtica',
'Curtosis', 'Asimetria', 'Mximo', 'Mnimo', 'Varianza',
'Desviacinestndar'};
predictors = inputTable(:, predictorNames);
response = inputTable.Estado;
isCategoricalPredictor = [false, false, false, false, false, false,
false, false, false, false];

% Train a classifier
% This code specifies all the classifier options and trains the
classifier.
classificationTree = fitctree(...
    predictors, ...
    response, ...
    'SplitCriterion', 'gdi', ...
    'MaxNumSplits', 100, ...
    'Surrogate', 'off', ...
    'ClassNames', categorical({'BE'; 'MEF1'; 'MEF2'; 'MEF3'}, {'BE' 'ME'
'MEF1' 'MEF2' 'MEF3'}));

% Create the result struct with predict function
predictorExtractionFcn = @(t) t(:, predictorNames);
treePredictFcn = @(x) predict(classificationTree, x);
trainedClassifier.predictFcn = @(x)
treePredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct
trainedClassifier.RequiredVariables = {'Asimetria', 'Curtosis',
'Desviacinestndar', 'Media', 'Mediacuadrtica', 'Mediana', 'Mnimo',
'Moda', 'Mximo', 'Varianza'};
trainedClassifier.ClassificationTree = classificationTree;
trainedClassifier.About = 'This struct is a trained model exported from
Classification Learner R2020a.';
trainedClassifier.HowToPredict = sprintf('To make predictions on a new
table, T, use: \n  yfit = c.predictFcn(T) \nreplacing ''c'' with the name
of the variable that is this struct, e.g. ''trainedModel''. \n \nThe
table, T, must contain the variables returned by: \n  c.RequiredVariables
\nVariable formats (e.g. matrix/vector, datatype) must match the original
training data. \nAdditional variables are ignored. \n \nFor more
information, see <a href="matlab:helpview(fullfile(docroot, ''stats'',
''stats.map''), ''appclassification_exportmodeltoworkspace'')">How to
predict using an exported model</a>.');

% Extract predictors and response
% This code processes the data into the right shape for training the
% model.
inputTable = Prosdat12;
predictorNames = {'Media', 'Mediana', 'Moda', 'Mediacuadrtica',
'Curtosis', 'Asimetria', 'Mximo', 'Mnimo', 'Varianza',
'Desviacinestndar'};
predictors = inputTable(:, predictorNames);
response = inputTable.Estado;
isCategoricalPredictor = [false, false, false, false, false, false,
false, false, false, false];
```

```
% Perform cross-validation
partitionedModel = crossval(trainedClassifier.ClassificationTree,
'KFold', 7);

% Compute validation predictions
[validationPredictions, validationScores] =
kfoldPredict(partitionedModel);

% Compute validation accuracy
validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun',
'ClassifError');

yfit = trainedClassifier.predictFcn(Tabla1)
```