DECANATURA DE INGENIERÍA INDUSTRIAL
DECANATURA DE INGENIERÍA DE SISTEMAS
DECANATURA DE MATEMÁTICAS
MAESTRÍA EN CIENCIA DE DATOS
FORMATO DE ENTREGA TRABAJO DE GRADO

Fecha de entrega: 06/11/2023

Estudiante: Laura Melisa Patarroyo Godoy

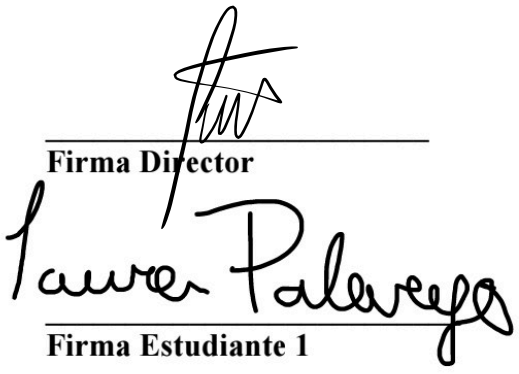
Santiago Jején Salinas

Director: Javier Alberto Chaparro Preciado

Codirector:

El presente documento avala la entrega del trabajo de grado por parte del director y codirector.

Documentos anexos: copia digital del Trabajo de Grado (1).



Firma Director

Firma Codirector

Firma Estudiante 1

Firma Estudiante 2

Detección del tipo de terreno sobre el que transita un Rover, a través de la aplicación de técnicas de machine learning.

Laura Melisa Patarroyo Godoy

Santiago Jején Salinas

Escuela Colombiana de Ingeniería Julio Garavito
Decanatura de Ingeniería Industrial
Decanatura de Ingeniería de Sistemas
Decanatura de Matemáticas
Maestría en Ciencia de Datos
Bogotá D.C., Colombia
2023

Detección del tipo de terreno sobre el que transita un Rover, a través de la aplicación de técnicas de machine learning.

Laura Melisa Patarroyo Godoy
Santiago Jején Salinas

Trabajo de grado para optar al título de
Magíster en Ciencia de Datos

Director
Javier Alberto Chaparro Preciado
Doctor

Escuela Colombiana de Ingeniería Julio Garavito
Decanatura de Ingeniería Industrial
Decanatura de Ingeniería de Sistemas
Decanatura de Matemáticas
Maestría en Ciencia de Datos
Bogotá D.C., Colombia
2023

Publicado en 2013 por la Escuela Colombiana de Ingeniería Julio Garavito. Avenida 13 No 205-59 Bogotá. Colombia
TEL: +57 – 1 668 36 00

Reconocimiento o Agradecimientos

A nuestros padres, hermanas y abuelas, por estar siempre motivándonos y apoyándonos para el crecimiento académico.

A nuestro tutor, Javier Alberto Chaparro Preciado, por su adecuada guía, confianza y paciencia durante el proceso de desarrollo del proyecto y todo muestro proceso de formación siempre nos estuvo apoyándonos y guiándonos. Sus aportes y conocimiento fueron imprescindibles durante este tiempo.

Resumen

El objetivo principal de este proyecto fue emplear sensores de una Unidad de Medición Inercial (IMU), específicamente un acelerómetro y un giroscopio, montados en un robot todoterreno (Rover). El Rover se desplazó por diversos tipos de superficies, como arena, ladrillo, asfalto, piedra y pasto, con el propósito de recolectar datos y generar señales. Estos datos se utilizaron para crear conjuntos de datos que fueron sometidos a procesos de análisis y procesamiento de datos. En este proyecto, se aplicaron diferentes algoritmos de aprendizaje automático (machine learning) y se realizó una extensa ingeniería de características, explorando tanto el dominio del tiempo como el dominio de la frecuencia. La meta principal del proyecto fue desarrollar un modelo capaz de predecir automáticamente en qué tipo de terreno se encontraba el Rover basándose únicamente en las lecturas de los sensores IMU. Como parte de la propuesta final, se buscó la implementación de un modelo de clasificación en lenguaje C para optimizar su eficiencia, rendimiento y despliegue sobre pequeños microcontroladores.

Abstract

The project's objective was to make use of sensors from an Inertial Measurement Unit (IMU), such as the accelerometer and gyroscope, based on a Rover that moved across different types of terrain, including sand, brick, asphalt, stone, and grass. The goal was to collect data, generate signals, and build a dataset, which was later analyzed and processed using machine learning algorithms, and extensive feature engineering was performed, exploring both the time and frequency domains to obtain a model capable of classifying the type of terrain the Rover was traversing. As a final part of the project, the implementation of a classification model in C language was proposed.

Tabla de contenido

Lista de Figuras

Lista de Tablas

1 El Proyecto

1.1 Introducción

Se estima que la población mundial crezca hasta los 9.700 millones de personas para el año 2050, (United Nations, 2022), por lo tanto, la producción agrícola debe hacerlo hasta cerca de un 70% para poder responder al crecimiento, y debe hacerlo cuidando el medio ambiente. Los diferentes avances tecnológicos, han transformado poco a poco la agricultura, permitiendo responder a las necesidades de la población, haciéndola más rentable, precisa y sostenible.

Con el objetivo de hacer más sencillas y eficientes las actividades agrícolas como la siembra y la cosecha, tecnificando el uso de la tierra sin hacer mayor uso de recursos, se han empezado a emplear vehículos autónomos que realicen tareas repetitivas y que, a su vez, permitan tener una mayor productividad.

Para lo anterior, es importante que el vehículo tenga un adecuado funcionamiento, lo que implica una correcta detección de la superficie sobre la que se está movilizando, ya que, de acuerdo con esta detección, este puede realizar determinada tarea para la cual se le programe.

En este proceso, se pueden emplear diferentes tipos de sensores, que generan gran cantidad de información sobre la cual se pueden realizar diferentes análisis para detectar patrones. Esto, sumado al uso de técnicas de machine learning, permite desarrollar modelos de predicción y clasificación, que dan solución a diferentes tipos de necesidades dentro del agro.

En este proyecto, se presenta el uso de sensores (acelerómetro y giroscopio) en un Rover, la obtención y construcción de conjuntos de datos, análisis, y la construcción del modelo que permitió detectar y clasificar el tipo de superficie sobre el que transitó el Rover.

1.2 Justificación

La clasificación de tipos de superficies en entornos de exploración espacial y terrestre es un desafío crucial en la robótica y la exploración. Los Rovers y vehículos autónomos deben adaptarse a una variedad de terrenos para llevar a cabo tareas de exploración y muestreo. La capacidad de determinar con precisión el tipo de superficie que un Rover está atravesando es esencial para la seguridad y la eficiencia de las misiones.

Hasta la fecha, la clasificación de superficies se ha basado en sensores como cámaras, lidar y radares, lo que a menudo requiere un consumo significativo de energía y ancho de banda de comunicación. Este enfoque tradicional puede ser insuficiente en entornos remotos o en misiones de larga duración donde los recursos son limitados. (Foil, 2013)

La importancia de este enfoque radica en su potencial para mejorar la eficiencia y la autonomía de los rovers en misiones de exploración, reduciendo al mismo tiempo los costos asociados a la implementación de sensores adicionales. Además, la capacidad de clasificar superficies con precisión puede tener un impacto significativo en la seguridad y la calidad de la toma de decisiones de los vehículos autónomos. (Foil, 2013)

Por otro lado, la innovación en la agricultura, haciendo uso de los avances tecnológicos, representa una oportunidad de crecimiento sostenible. Los robots móviles o rovers son una alternativa atractiva en la industria agrícola, ya que pueden llevar a cabo tareas complejas y repetitivas de manera autónoma, lo que reduce la necesidad de intervención humana y mejora la eficiencia en el campo, que, junto a un equipamiento de sensores, permite recopilar datos útiles para los agricultores, como información sobre la calidad del suelo, la humedad, la temperatura y la presencia de plagas y enfermedades.

La capacidad de un Rover para reconocer y adaptarse al terreno sobre el que se está moviendo es crucial para garantizar una navegación autónoma y segura de los mismos en entornos desconocidos. Así como en (Huang, Yu Huang, & Chen, 2017), donde se hizo uso de las señales provenientes de la IMU, con el fin de predecir la actitud del vehículo: en línea recta, girando a la derecha o izquierda, quieto, entre otras. Hicieron uso del acelerómetro y el giroscopio, y las señales recolectadas las introdujeron a redes convolucionales o CNN. Otro uso de una IMU (Unidad de medida Inercial), fue presentada en (Christian, y otros, 2019), donde la IMU estuvo sujeta a un lápiz mientras se dibujaban diferentes números. Aquí, hicieron uso del aprendizaje profundo con el fin de clasificar correctamente los números escritos.

Por lo anterior, el uso de técnicas de aprendizaje automático para la detección de terreno ofrece una solución flexible y escalable; puesto que con el uso de algoritmos de machine learning se pueden analizar grandes cantidades de datos y aprender a reconocer patrones que permitan identificar los diferentes tipos de superficies.

1.3 Objetivos

A continuación, se encuentra el objetivo general y los objetivos específicos sobre los que se trabajó la propuesta de grado.

1.3.1 Objetivo General

Diseñar y evaluar un sistema de machine learning para identificar el terreno sobre el cual transita un Rover.

1.3.2 Objetivos Específicos

- Ambientar los diferentes escenarios por los que puede transitar un Rover, de cara a la recolección y creación del conjunto de datos.
- Acondicionar el Rover, junto con los elementos necesarios para la adquisición de datos.
- Caracterizar los diferentes terrenos en los que se puede encontrar el Rover, a partir de las señales obtenidas por este, con el fin de identificar patrones.
- Plantear la arquitectura de datos que se usará desde la recolección de datos hasta el ingreso al modelo, para el correcto flujo y funcionamiento del sistema.
- Investigar, aplicar y evaluar técnicas de machine learning, sobre el conjunto de datos, para poder construir el modelo de detección del terreno sobre el que transita el Rover.

1.4 Alcance y Limitaciones

A continuación, se presenta el alcance y limitaciones del proyecto:

- El conjunto de datos fue construido en su totalidad, realizando la recolección de los datos que arrojó la IMU al movilizar el Rover sobre varios tipos de superficie, usando un Arduino nano sense 33 ble.
- Se trabajó con 5 tipos de superficies: pasto, ladrillo, arena, piedra y asfalto.
- El modelo se enfocó en identificar el tipo de superficie sobre la que transita el Rover.

2 Fundamentación Teórica

2.1 Vehículos Autónomos

Los vehículos autónomos, son aquellos equipados con tecnología para omitir la dependencia total o parcial del control humano. Estos, interpretan su entorno a través de sensores, actuadores, sistemas GPS, cámaras, software, entre otros, para generar datos y permitir la toma de decisiones. Para realizar actividades como siembra automática, irrigación, recolección de productos, mediciones y recolección de información, u otras, se pueden utilizar vehículos autónomos aéreos, fijos o terrestres.

2.1.1 Robots Agrícolas

Los robots agrícolas están diseñados con la finalidad de sustituir actividades dentro del sector, automatizando tareas repetitivas y de gran esfuerzo dentro del campo. El uso de estos robots tiene diferentes beneficios, desde lograr una eficiencia en el uso de recursos, disminuir costos, hasta aumentar la productividad. Actualmente, existen diferentes tipos de robots, que realizan

diferentes actividades, de acuerdo con la necesidad que se desea suplir. Entre los más comunes se encuentran el robot cosechador, el robot recolector, el robot de supervisión y mantenimiento, entre otros.

2.2 Machine Learning

Machine learning se entiende como la disciplina que permite a los ordenadores tener la capacidad de identificar patrones en grandes volúmenes de datos y elaborar predicciones.
"El aprendizaje automático se define como un conjunto de métodos que pueden detectar automáticamente patrones en los datos y luego usarlos para predecir datos futuros, o para realizar otros tipos de toma de decisiones bajo incertidumbre "(Murphy, 2012)

Los algoritmos de machine learning, se dividen en 3 categorías:

- Aprendizaje supervisado: posee un aprendizaje previo basado en etiquetas asociadas a los datos (es decir, datos que ya han sido clasificados previamente), que le permiten tomar decisiones o hacer predicciones de nuevos datos de entrada.
- Aprendizaje no supervisado: su objetivo es encontrar patrones en un conjunto de datos, que permitan ordenar o clasificarlos, sin un aprendizaje previo.
- Aprendizaje por refuerzo: su objetivo es aprender a partir de la experiencia.

2.3 Feature Engineering

La ingeniería de características, o *feature engineering*, es un proceso fundamental en el análisis de datos y en la creación de modelos de aprendizaje automático. Este proceso implica la transformación de los datos brutos en características o variables que permiten representar de manera efectiva la información contenida en los datos y que son útiles para un modelo de aprendizaje automático. En el presente estudio, la ingeniería de características es especialmente importante para extraer información relevante de las señales de vibración de la IMU y transformarla en características útiles para la clasificación. Un buen *feature engineering* puede mejorar significativamente la precisión y el rendimiento del modelo de aprendizaje automático utilizado en este trabajo, lo que permitirá obtener resultados más precisos y confiables.

2.4 Media

La media aritmética, es una variable estadística que se define como la suma de todos los valores dividido el número total de estos. Su cálculo viene dado por la ecuación:

Ecuación 1. Media

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Comúnmente, es usada en la etapa de preprocesamiento y análisis exploratorio de los datos, para luego trabajar con los modelos de aprendizaje automático. Las técnicas o usos más comunes se encuentran en:

- Imputación de valores faltantes: los valores faltantes de un conjunto de datos se reemplazan por la media para preservar la tendencia de los datos
- Normalización: esto implica restar la media de una característica y dividirla por la desviación estándar. Esto se realiza con el fin de que las características posean escalas comparables.
- Análisis exploratorio: el cálculo y exploración de la media de las características, proporciona información sobre la tendencia de los datos y permite identificar valores atípicos.
- Evaluación del rendimiento: en algunos modelos, se usa el error cuadrático medio (MSE) como métrica para evaluar su rendimiento, dónde la media se usa como punto de referencia para el mismo.

2.5 Varianza

La varianza es una medida estadística que representa la dispersión de los datos con respecto a la media de la muestra. Es dada en unidades al cuadrado. Su cálculo viene dado por la ecuación:

Ecuación 2. Varianza

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{n}(x_i - \bar{x})^2,$$

Es usada en machine learning, usualmente para el análisis de datos y evaluación del rendimiento del modelo.

- Selección de características: se usa para simplificar el modelo, ya que las características con baja varianza pueden eliminarse del conjunto de datos.
- Análisis exploratorio: es importante en esta etapa, ya que permite comprender la variabilidad en las características del conjunto de datos.
- Evaluación del rendimiento: en modelos de clasificación, puede usarse para evaluar la dispersión de las predicciones del modelo en clases. Una varianza alta, puede indicar que el modelo tiene dificultades para clasificar correctamente.

2.6 Entropía

La entropía, se usa para medir la incertidumbre de una distribución de probabilidad. Está definida cómo la suma ponderada de las probabilidades de sus posibles valores por el logaritmo de estas en base 2, por -1; y representada por la ecuación:

Ecuación 3. Entropía

$$H(x) = -\sum[P(x) * log_2(P(x))]$$

Cuanto mayor sea el valor de la entropía, mayor será la incertidumbre en la distribución de probabilidad.

Es útil en machine learning en el análisis y exploración de datos, así como en la construcción de los modelos:

- Análisis exploratorio: al calcular la entropía de las características, una entropía alta puede significar que contiene información útil (variabilidad relevante); también, puede ayudar a comprender la variabilidad de los datos y priorizar la selección de ciertas características.
- Modelos: en los árboles de decisión, le entropía es usada para medir la pureza del conjunto de datos en un nodo.

2.7 Simetría

Dentro de un conjunto de datos, una característica es simétrica si sus valores poseen una distribución similar a ambos lados de un punto central, siendo útil para entender la distribución de los datos. Una distribución perfectamente simétrica tiene un coeficiente de asimetría igual a cero.

Usualmente, se usa el coeficiente de asimetría skewness, para la distribución de datos, y está dado como:

Ecuación 4. Simetría

$$S = \frac{\sum(X_i - \mu)^3}{n * \sigma^3}$$

En este contexto es usado comúnmente para:

- Caracterizar datos: ya que proporciona información sobre la asimetría de una distribución de datos, permite comprender la forma y estructura de los datos, pilar en la exploración de datos.
- Identificar sesgos: si los datos poseen asimetría relevante, puede afectar el rendimiento de un modelo que asuma que los datos tienen un distribución normal o simétrica.
- Selección de características: puede ser usado como criterio para la selección de características a usar del conjunto de datos.

2.8 Curtosis

La curtosis es una medida estadística, utilizada para describir la forma de una distribución de datos en términos de la concentración de valores alrededor de la media y la presencia de *outliers*, o valores extremos. Su cálculo está dado por la ecuación:

Ecuación 5. Curtosis

$$K = \frac{\sum(x_i - \mu)^4}{(n - \sigma^4)} - 3$$

Si K es igual a 3, el conjunto de datos posee una distribución con curtosis mesocúrtica, es decir, se asemeja a un distribución normal.
Si K es mayor a 3, el conjunto de datos, posee una distribución con curtosis leptocúrtica, es decir, más concentrada en comparación a una distribción normal.
Si K es menor a 3, el conjunto de datos posee una distribución con curtosis platicúrtica, es decir, más dispera en comparación con una distribución normal.

2.9 Transformada de Fourier

La transformada de Fourier, es una transformación matemática, que permite descomponer señales al dominio de la frecuencia, y analizar como cada frecuencia contribuye a la señal global. La transformada de Fourier se puede aplicar a señales continuas o discretas, y es especialmente útil para analizar señales periódicas.

Teniendo una función original x(t), la transformada de Fourier X(f) se calcula:

Ecuación 6. Transformada de Fourier

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi k}{N}}$$

Donde:

- $X[k]$es el valor en el dominio de la frecuencia en k.
- $x[n]$es el valor en el dominio del tiempo en el instante n.
- N es la longitud de la señal.
- j es la unidad imaginaria.

Es ampliamente utilizada en análisis de señales y sistemas, procesamiento de señales de audio y video, y codificación de información.

2.9.1 Transformada rápida de Fourier

La Transformada Rápida de Fourier (FFT) es un algoritmo que revoluciona este proceso de obtención de la transformada de Fourier. Fue desarrollada por Cooley y Tukey en la década de 1960 y permite calcular eficientemente la Transformada de Fourier. La FFT divide una señal en segmentos más pequeños y calcula las Transformadas de Fourier de cada segmento, reduciendo significativamente la complejidad computacional. Esta técnica es ampliamente utilizada en una variedad de aplicaciones, desde procesamiento de señales digitales hasta análisis de imágenes y cálculos científicos. (William H. Press, 2007)

La FFT se ha convertido en una herramienta fundamental en la caracterización de señales y el análisis de datos en tiempo real. Su capacidad para descomponer señales en sus componentes de frecuencia ha tenido un impacto significativo en campos como la teledetección, la comunicación inalámbrica y la ingeniería de audio, entre otros. En el contexto de este proyecto, la FFT se aplica para analizar las señales generadas por los sensores de una IMU, proporcionando información crucial para la clasificación de terrenos.

2.9.2 Frecuencia Dominante

Es un concepto importante en el análisis de señales, particularmente en el dominio de la transformada de Fourier. Se refiere a la frecuencia que tiene la mayor amplitud o energía en una señal dada. En otras palabras, es la frecuencia que predomina en una señal, lo que significa que tiene la mayor contribución en términos de amplitud o potencia. (Downey, 2016)

Cuando se aplica una transformada de Fourier a una señal, se descompone en sus componentes de frecuencia individuales. La frecuencia dominante es aquella que corresponde al pico más alto en el espectro de frecuencia de la señal. Esta frecuencia suele ser la que más caracteriza el comportamiento de la señal y puede proporcionar información importante sobre su contenido.

2.9.3 Ancho de Banda

En el contexto del procesamiento de señales se refiere a una medida que describe cuán amplio o estrecho es el rango de frecuencias en el que se concentra la energía de una señal. En otras palabras, el ancho de banda indica cuántas frecuencias diferentes contribuyen significativamente a una señal y cuánto espacio ocupa este rango de frecuencias. (Robert W. Heath, 2017)

Para comprender mejor el concepto de ancho de banda, es útil imaginar el espectro de frecuencia de una señal, que es una representación gráfica de las componentes de frecuencia de la señal en función de su amplitud

2.9.4 Potencia Total

En el contexto de la Transformada Rápida de Fourier (FFT) se refiere a la cantidad total de energía contenida en una señal en el dominio de la frecuencia. (Robert W. Heath, 2017) La potencia total es una medida importante ya que proporciona información sobre cuánta energía hay distribuida en las diferentes componentes de frecuencia de la señal, matemáticamente, la potencia total se puede expresar como:

Ecuación 7. Potencia Total

$$Ptotal = \sum_{i=0}^{N-1} |X[i]|^2$$

Donde:

- $Ptotal$ es la potencia total.
- N es el número de muestras en la señal.
- $X[i]$ es el coeficiente de la FFT en la posición i.

2.9.5 Entropía Espectral

Es una métrica utilizada en el análisis de señales y el procesamiento de señales para caracterizar la distribución de energía de una señal en el dominio de la frecuencia. Esta métrica se deriva de

la teoría de la información y se utiliza para evaluar la dispersión de la potencia espectral en una señal de forma cuantitativa. (Robert W. Heath, 2017) La entropía espectral proporciona información sobre cuán diversificadas o uniformes son las componentes de frecuencia de una señal. Para calcular la entropía espectral se emplea:

Ecuación 8. Entropía Espectral

$$H(f) = -\sum_{i=1}^{N} P(f_i) \log_2\big(P(f_i)\big)$$

Donde:

- $H(f)$ es la entropía espectral.
- $P(f_i)$ es la densidad espectral de potencia en la frecuencia
- N es el número de frecuencias discretas en el espectro.

2.10 Minería de datos

La minería de datos hace referencia al uso de técnicas y tecnologías para explorar y analizar grandes volúmenes de datos. Actualmente, por los avances tecnológicos, los datos pueden entenderse como materia prima bruta, por lo que el objetivo de la minería es encontrar patrones y/o tendencias que expliquen el comportamiento de los datos en un contexto, para agregarles valor y con ello poder tomar decisiones.

Existen diferentes metodologías que establecen como realizar el proceso y las tareas dentro de cada una de las fases; para el desarrollo del proyecto, se implementó la metodología KDD, cuyo modelo establece que "la minería de datos es la etapa dentro del proceso en la cual se realiza la extracción de patrones a partir de los datos" (Moine, Haedo, & Gordillo)

2.11 Reducción de la Dimensionalidad

La reducción de la dimensionalidad es una técnica fundamental en la investigación de análisis de datos y aprendizaje automático. Su objetivo principal es abordar el desafío inherente a conjuntos de datos de alta dimensionalidad, caracterizados por tener un gran número de variables o características, buscando disminuir dicho número sin afectar el objeto de estudio, con la finalidad de obtener un conjunto más manejable. (Géron, 2022) Existen varios métodos de reducción de la dimensionalidad, sin embargo, en este trabajo se empleó el método de análisis de componentes

principales.

2.11.1 Análisis de componentes principales

El Análisis de Componentes Principales (PCA: *Principal Component Analysis*) es una técnica fundamental en la minería de datos y el análisis de datos multivariados. Su objetivo principal radica en reducir la dimensionalidad de un conjunto de datos, manteniendo la mayor cantidad de información posible y destacando las relaciones más significativas entre las variables.

PCA aborda el desafío de trabajar con conjuntos de datos de alta dimensionalidad, una problemática común en la ciencia de datos. Al proyectar los datos en un nuevo espacio, denominado espacio de componentes principales, PCA permite representar la variabilidad de los datos en un número menor de dimensiones. Esto es esencial para simplificar la complejidad de los datos y facilitar análisis posteriores.

Un aspecto fundamental del PCA es que busca encontrar componentes principales que sean estadísticamente independientes y ortogonales entre sí en el nuevo espacio. Esto significa que las variables en el nuevo espacio están no correlacionadas, lo que facilita la interpretación de las relaciones entre ellas. Al preservar las características más relevantes y eliminar la redundancia, PCA mejora la eficiencia en la exploración de datos.

PCA se destaca por su capacidad para capturar la varianza de los datos. El primer componente principal captura la mayor cantidad de varianza, y los componentes subsiguientes se seleccionan de manera que retengan la varianza restante en orden descendente. Esto asegura que se mantenga la información más relevante, lo que resulta en una representación compacta pero informativa de los datos.

El PCA tiene una amplia gama de aplicaciones en la minería de datos, incluyendo la reducción de ruido en datos, la identificación de patrones ocultos, la visualización de datos de alta dimensión y la compresión de información.

2.12 Marco Metodológico

Como marco metodológico para el desarrollo del proyecto, se empleó la metodología KDD (*Knowledge Discovery in Databases*), definida como "El proceso no trivial de identificación de patrones válidos, novedosos, potencialmente útiles y fundamentalmente entendibles al usuario a partir de los datos" (Fayyad, Piatetsky-Shapiro, & Smyth, 1996).

Este proceso involucra diferentes etapas, que son:

2.12.1 Selección

En esta etapa, se debe realizar un entendimiento del objetivo del estudio, obtener conocimientos previos e identificar la meta del proyecto. También, se establece el conjunto de datos objetivo (totalidad o muestra), en concordancia con su finalidad.

2.12.2 Preprocesamiento / Limpieza

En esta etapa, se analizan los datos en términos de calidad, es decir, se llevan a cabo diversas tareas para preparar los datos para el análisis. Esto puede incluir la limpieza de los datos, la transformación de variables, la eliminación de valores faltantes y la selección de características relevantes. Aquí se aplican técnicas para remover ruido, se establece como va a ser el manejo de datos desconocidos y de datos nulos. Todos estos valores, se pueden ignorar, o se reemplazan, haciendo uso de métricas estadísticas como la media, moda, mínimo, máximo, etc.

2.12.3 Transformación / Reducción

En esta etapa, se transforman los datos preprocesados en un formato que pueda ser utilizado por los algoritmos de minería de datos. Esto puede incluir la reducción de la dimensionalidad, la normalización de los datos y la selección de características.

2.12.4 Data Mining

En esta etapa, se aplican técnicas de minería de datos para extraer patrones, tendencias y relaciones de los datos transformados. Esto puede incluir la aplicación de algoritmos de aprendizaje automático, análisis estadístico y minería de redes.

2.12.5 Interpretación / Evaluación

En esta etapa, se evalúan los resultados obtenidos a partir de la minería de datos. Esto incluye la validación de modelos, la comparación de diferentes algoritmos y la interpretación de los resultados obtenidos, que luego se utilizan para resolver el problema de negocio que se quería abordar. Aquí se puede identificar oportunidades de negocio, tomar decisiones e implementar soluciones basadas en los resultados obtenidos.

2.13 Proceso Metodológico

El proyecto se desarrolló sobre la metodología KDD (K*nowledge Discovery in Databases*), y aplicado al proyecto, se estableció de la siguiente forma:

Figura 1. Proceso Metodológico
Fuente. Elaboración propia

2.13.1 Selección

Durante esta etapa se definieron y prepararon los diferentes escenarios para la movilización del rover. Se identificaron y seleccionaron ubicaciones en la Universidad con terrenos planos que representaran las diferentes superficies a recolectar: arena, ladrillo, pasto, piedra y asfalto.

Para la adquisición de datos, se empleó un Arduino Nano Sense 33 equipado con una IMU (Unidad de Medición Inercial) de 9 ejes además de otra variedad de sensores. En particular, se utilizó el acelerómetro y el giroscopio de la IMU para capturar las señales relevantes. La programación se llevó a cabo utilizando el entorno de desarrollo proporcionado por Arduino, lo que permitió configurar el dispositivo para registrar las mediciones de manera precisa.

Una vez identificados estos escenarios, se procedió a la recolección de datos. El conjunto de datos incluyó las señales del acelerómetro en los ejes X, Y, y Z, así como del giroscopio en los ejes X, Y, y Z. Estas señales fueron registradas para cada tipo de superficie.

Las señales capturadas se transmitieron desde el Arduino al computador mediante una conexión por puerto serie. Cada conjunto de señales correspondiente a un tipo de superficie y a una toma específica se almacenó en archivos separados por comas (formato CSV). Dado que se realizaron múltiples tomas para cada tipo de superficie, se generaron archivos individuales para mantener los datos organizados y facilitar el posterior procesamiento y análisis. Este enfoque garantizó la integridad de los datos y permitió una gestión eficiente de la información recopilada.

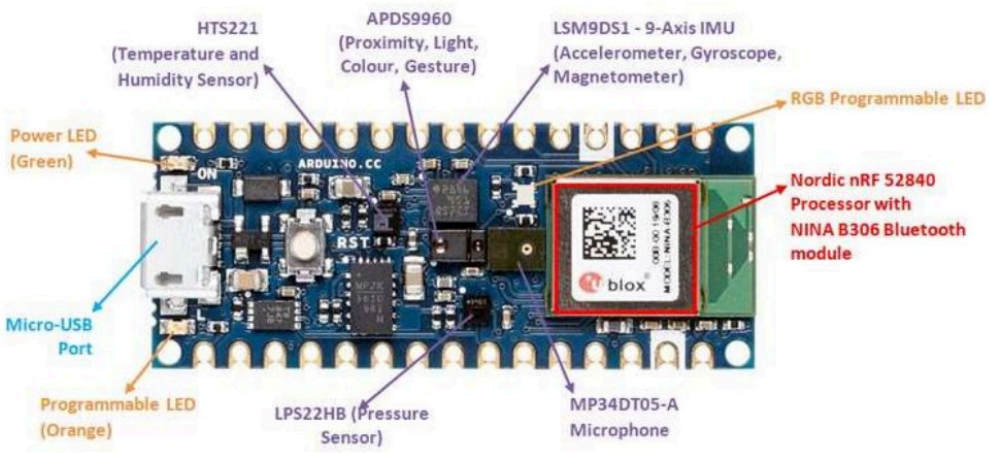


Figura 2. Arduino Nano 33 BLE SENSE Board
Fuente. (https://makerbazar.in/products/arduino-nano-33-ble-sense-board)

Se hizo uso de un rover tortuga adecuado con una tarjeta sense 33 ble encima, como se muestra en la Figura 3.

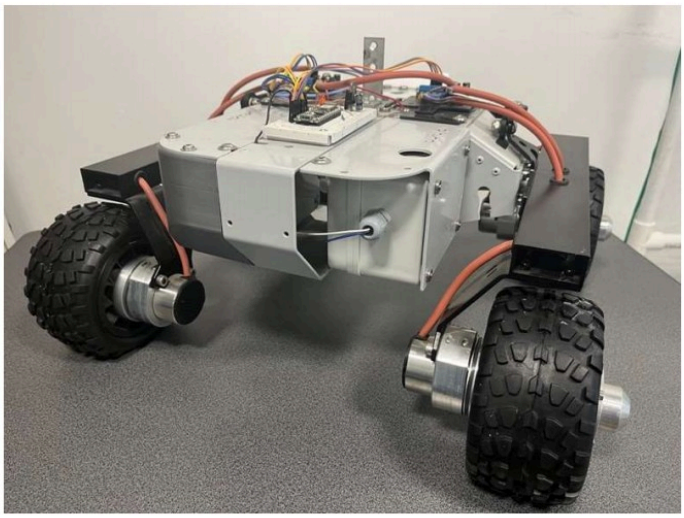


Figura 3. Rover tortuga. Fuente: Elaboración Propia

El proceso de recolección de datos siguió siempre el mismo orden:

- Encendido del rover.
- Inicialización del programa realizado para la lectura de datos a través de la tarjeta conectada al computador a través del puerto USB.
- Colocación del rover sobre la superficie determinada.

- Movilización del rover durante 15 segundos, leyendo y transfiriendo los datos cada 1/100Hz, es decir, cada 0.01 segundos.
- Apagado del rover.

Cada ejecución del proceso descrito anteriormente permitió recolectar 1500 datos, pero, ya que en el proceso de encendido del rover y ubicación en el terreno, así como en el de apagado del mismo, se mantuvo en el aire o quieto por algunos segundos, se eliminaron lo primeros y últimos 500 datos, dejando como resultado de la ejecución únicamente 500 datos. Por cada tipo de superficie se realizaron 5 ejecuciones del proceso, obteniendo 2500 datos totales para cada una de ellas, sumando 12500 datos en total por todos los tipos de superficie.

2.13.2 Preprocesamiento

En esta etapa se almacenaron e integraron los datos recolectados. Cada vez que se ejecutó una toma de datos, estos se guardaron en un archivo CSV. En total, se generaron 5 archivos CSV por cada tipo de superficie, lo que sumó un total de 25 archivos CSV.
El análisis de estos datos se llevó a cabo utilizando Python y la librería scikit-learn en un ambiente colaborativo como Google Colab. Para integrar los datos, se unieron los cinco conjuntos de datos de cada tipo de superficie (Arena, Piedra, Asfalto, Pasto y Ladrillo) para obtener un conjunto de datos único por tipo de superficie. Posteriormente, se agregó la columna 'suelo' a cada conjunto de datos y se asignó un valor numérico del 0 al 4 en función del tipo de superficie. Estos cinco conjuntos de datos se combinaron en un conjunto de datos final con el que se trabajó.

Además, se generaron gráficos de dispersión que agrupaban de a 2 en todas las posibles combinaciones de las 6 variables. Este paso permitió identificar visualmente las variables que podrían ser determinantes para marcar una diferenciación según el tipo de terreno.

2.13.3 Transformación

En esta etapa, se definieron dos análisis a realizar: análisis en el dominio del tiempo y análisis en el dominio de la frecuencia.

2.13.3.1 Transformación en el dominio del tiempo

Para el análisis en el dominio del tiempo, se realizaron 5 caracterizaciones de las señales:

- Caracterización por medias
- Caracterización por varianza
- Caracterización por entropía
- Caracterización por simetría

- Caracterización por curtosis

También se consideraron diferentes transformaciones, como escalar o normalizar los datos. Se definieron funciones para ventanear los datos y aplicar la medida estadística correspondiente (media, varianza, simetría, entropía o curtosis). Este proceso generó tres conjuntos de datos, cada uno derivado de la variación del tamaño de la ventana (20, 15 y 10). Se varió la medida estadística aplicable entre media, varianza, simetría, curtosis y entropía. Cada vez que se varió esta medida, también se ejecutaron los subprocesos posteriores (modelado e interpretación).
En total, se obtuvieron 75 dataframes que fueron utilizados como insumo para la etapa de modelado.

2.13.3.2 Transformación en el dominio de la frecuencia

En esta sección, se describe el enfoque que se llevó a cabo para el análisis de las seis señales de interés obtenidas de la IMU, usando ingeniería de características con transformada rápida de Fourier. La Figura 4 proporciona una representación visual del proceso general llevado para una señal.

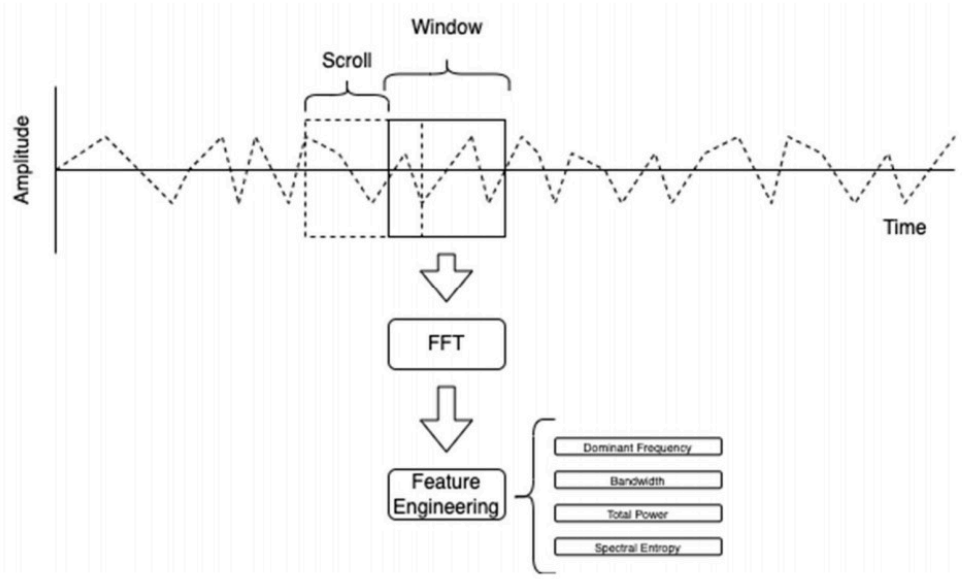


Figura 4. Esquema general caracterización usando transformada de Fourier.
Fuente. Elaboración propia.

En la Figura 4, las "Windows" se refieren a segmentos de las señales de interés. Cada ventana comprende un conjunto de muestras de la señal. Estas ventanas se deslizan a lo largo de las señales con un "scroll", lo que implica un desplazamiento secuencial y parcial de las ventanas a lo largo de las señales.

A cada ventana, se le aplica la Transformada rápida de Fourier por Ventanas (FFT). La FFT permite transformar las señales del dominio del tiempo al dominio de la frecuencia. Esto nos proporciona información detallada sobre las componentes de frecuencia presentes en cada ventana de la señal.

El enfoque anterior es aplicado para la clasificación de palabras en señales de audio, como lo describe (Warden & Situnayake, 2019) en su libro TinyMl, que emplean un microcontrolador con un micrófono para clasificar en tiempo real las palabras "Yes", "No" o palabra desconocida.

Una vez que hemos calculado la FFT para cada ventana, se realiza un proceso de "feature engineering" en el dominio de la frecuencia. Este proceso implica la extracción de diversas métricas clave que caracterizan cada ventana y proporcionan información relevante para nuestro análisis.

Durante el proceso de análisis de las señales, se extraen diversas métricas clave que desempeñan un papel fundamental en la caracterización de los eventos y comportamientos de interés. Estas métricas incluyen:

- Frecuencia Dominante: Esta métrica permite identificar la frecuencia más prominente en cada ventana. Su capacidad para señalar comportamientos específicos o patrones de interés es invaluable en nuestro análisis.
- Ancho de Banda: El ancho de banda proporciona información detallada sobre el rango de frecuencias significativas presentes en cada ventana de señal. Esta información resulta esencial para distinguir eventos y comportamientos en los datos.
- Potencia Total: Medir la potencia total en la ventana de frecuencia es crucial para reflejar la intensidad de las señales en diferentes segmentos. Esta métrica aporta una comprensión más profunda de la fuerza de los eventos detectados.
- Entropía Espectral: La evaluación de la distribución de frecuencias en cada ventana es llevada a cabo por la entropía espectral. Esta métrica desempeña un papel crítico en la comprensión de la variabilidad en las señales y la identificación de patrones.

Esta etapa de análisis permitió derivar un conjunto de nuevas variables, cada una de las cuales se relaciona directamente con una métrica específica y un eje de medición. Por ejemplo, se generaron variables como "Frecuencia Dominante de Aceleración en el Eje X", y así sucesivamente para todas las combinaciones posibles de métricas y ejes. En total, se obtuvieron variables como:

- 'Ax_Dominant_Frequency', 'Ax_Bandwidth', 'Ax_Total_Power', 'Ax_Spectral_Entropy'.
- 'Ay_Dominant_Frequency', 'Ay_Bandwidth', 'Ay_Total_Power', 'Ay_Spectral_Entropy'.
- 'Az_Dominant_Frequency', 'Az_Bandwidth', 'Az_Total_Power', 'Az_Spectral_Entropy'.

- 'Gx_Dominant_Frequency', 'Gx_Bandwidth', 'Gx_Total_Power', 'Gx_Spectral_Entropy'.
- 'Gy_Dominant_Frequency', 'Gy_Bandwidth', 'Gy_Total_Power', 'Gy_Spectral_Entropy'.
- 'Gz_Dominant_Frequency', 'Gz_Bandwidth', 'Gz_Total_Power', 'Gz_Spectral_Entropy'.

2.13.3.2.1 Selección de la ventana y desplazamiento

En el análisis de señales generadas por una Unidad de Medida Inercial (IMU), la elección del tamaño de la ventana y el desplazamiento desempeña un papel crucial en el procesamiento y la extracción de características. En esta sección, se explicará en detalle la importancia de estos dos parámetros y las razones detrás de su selección en el marco de la investigación.

2.13.3.2.1.1 La Ventana: Capturando Información en Segmentos

En nuestro estudio, el tamaño de la ventana se selecciona con cuidado para determinar cuántas muestras se consideran a la vez en el cálculo de la transformada de Fourier. La elección del tamaño de la ventana influye en la capacidad de detectar componentes de frecuencia en los datos y en la resolución en el dominio de la frecuencia.

Un tamaño de ventana más pequeño permite una mayor resolución en frecuencia, lo que significa que podemos identificar componentes de frecuencia más finas en la señal. Esto es esencial cuando se buscan cambios en las frecuencias de eventos en las señales IMU. Por ejemplo, si estamos interesados en detectar movimientos rápidos o eventos de alta frecuencia, un tamaño de ventana más pequeño es necesario para capturar esos cambios.

Sin embargo, es importante destacar que un tamaño de ventana más pequeño disminuirá la resolución temporal. Esto significa que podríamos perder información sobre cambios temporales rápidos en la señal. La elección del tamaño de la ventana debe ser un equilibrio entre la resolución en frecuencia y la resolución temporal.

Otro punto relevante en la selección del tamaño de la ventana es evitar que sea tan grande que abarque todas las muestras del conjunto de datos. Nuestro conjunto de datos consta de 2500 muestras, por lo que una ventana de 150 muestras representa aproximadamente el 6% del conjunto de datos. Esto es esencial para calcular el porcentaje de cobertura de datos por ventana.

Además, una ventana que exceda 150 muestras en tiempo equivaldría a más de 1.5 segundos de muestreo, considerando una frecuencia de muestreo de 100 Hz. Es fundamental evitar que las ventanas sean demasiado grandes, ya que esto podría diluir la capacidad de detectar eventos y comportamientos específicos en el tiempo.

Este enfoque en la selección del tamaño de la ventana se realiza para optimizar la resolución en frecuencia y temporal, garantizando un adecuado porcentaje de cobertura de los datos.

Además, hemos establecido una ventana mínima de 50 muestras para asegurarnos de que la FFT tenga un rango significativo de frecuencias para su cálculo. Esto nos permite definir un rango de ventanas desde 50 hasta 150 muestras.

2.13.3.2.1.2 El Desplazamiento: Controlando la Superposición

El desplazamiento es esencial para controlar la superposición entre las ventanas. Un paso más pequeño proporciona una mayor superposición, lo que puede mejorar la resolución temporal, pero aumentará el costo computacional.

Un paso pequeño permite capturar cambios temporales rápidos en la señal. Si hay eventos de corta duración que son de interés, un paso menor asegura que no se pasen por alto. Por otro lado, un paso más grande puede reducir la cantidad de datos procesados y, en algunos casos, puede ser más eficiente computacionalmente.

Por tanto, es común considerar pasos con un punto inicial en torno al 10% del tamaño de la ventana y evitar exceder pasos que representen más del 30% de la ventana. Esta elección no solo influye en la resolución temporal, sino también en la cantidad de datos en el conjunto resultante.

La cantidad de datos del conjunto resultante es un factor importante, ya que afecta directamente la capacidad de entrenar modelos de manera efectiva. Tener una cantidad adecuada de datos es esencial para garantizar la representatividad de las características extraídas de las ventanas. En este sentido, pasos demasiado grandes pueden reducir la cantidad de datos en el conjunto, lo que podría limitar la capacidad de entrenar modelos robustos.

Además, un paso excesivamente pequeño podría dar como resultado ventanas con características idénticas entre sí, lo que podría introducir problemas en el entrenamiento de modelos. La variabilidad en las características extraídas de las ventanas es esencial para capturar los patrones y eventos de interés en los datos.

En resumen, la elección cuidadosa del paso es esencial para equilibrar la resolución temporal, la cantidad de datos en el conjunto resultante y la variabilidad en las características. Esto asegura que nuestro análisis sea efectivo y que los modelos entrenados sean representativos y precisos.

2.13.4 Modelado

Esta etapa consistió en la construcción, entrenamiento e implementación de diferentes modelos de machine learning utilizando las variables definidas, con variaciones y optimizaciones necesarias de los parámetros correspondientes. Se trabajó en el dominio del tiempo y en el dominio de la frecuencia, al igual que para la sección 2.13.3.

2.13.4.1 Modelado de la caracterización en el dominio del tiempo

Para el análisis en el dominio del tiempo, se agruparon las variables en combinaciones de tres, considerando todas las posibles combinaciones de las seis variables existentes. Luego, el conjunto de datos se dividió en entrenamiento y prueba en una relación de 70-30, y se ingresaron en diferentes modelos de aprendizaje automático para evaluar su rendimiento y seleccionar el mejor modelo basado en estas evaluaciones. Además, se realizaron agrupaciones de 2 y 4 variables.

Los modelos que se entrenaron fueron: KNN (o vecinos cercanos), Naive Bayes Gaussiano, SVM, Random Forest, MLP y Red Neuronal. Estos modelos se configuraron usando los parámetros por defecto.

El proceso descrito, se realizó 2 veces más: una vez aplicando estandarización MinMax (para realizar la transformación de escala en un rango entre 0 y 1) y otra usando StandarScaler (para realizar la transformación de las características, para que los valores tengan una media de 0 y una desviación estándar de 1).

2.13.4.2 Modelado de la caracterización en el dominio de la frecuencia

En la Figura 5 se ilustra de manera detallada el procedimiento completo de transformación y modelado en el dominio de la frecuencia. Con las nuevas variables derivadas de la transformación mencionada en la sección 2.13.3.2, en ventanas de 50, 70, 100, 120 y 150 muestras, con desplazamientos variables entre el 10% y el 30% del tamaño de la ventana. Se dividen los datos en conjuntos de entrenamiento y prueba, asignándoles proporciones del 80% y 20%, respectivamente.

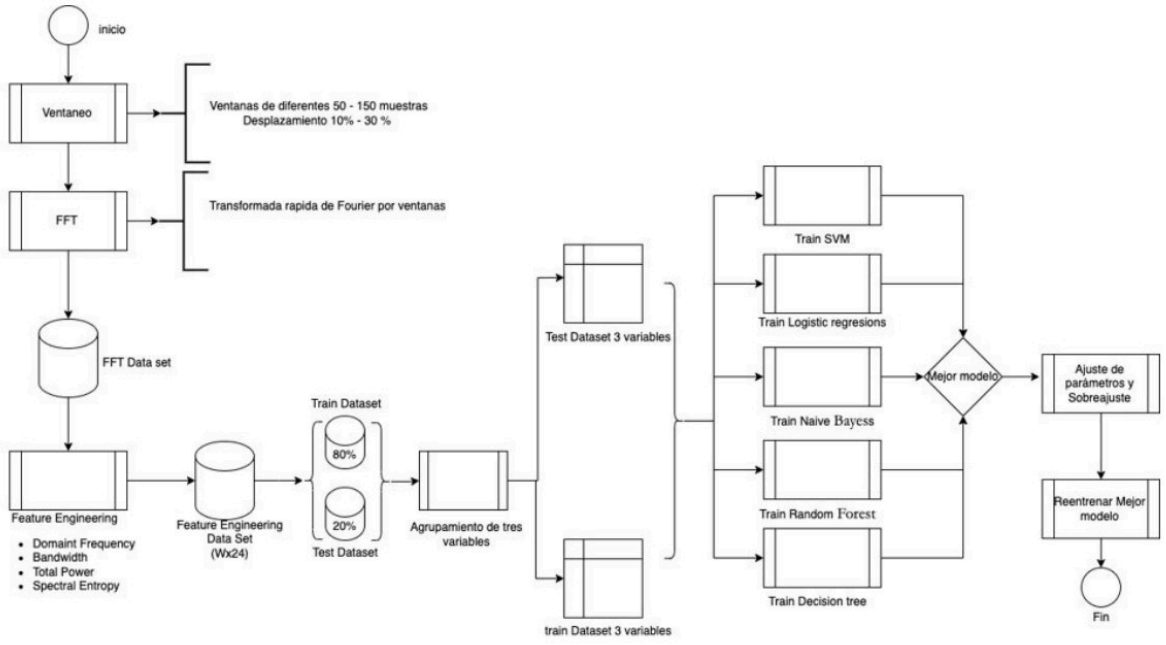


Figura 5. Diagrama análisis y procesamiento el dominio de la frecuencia.

Fuente: Elaboración propia

Los conjuntos de datos con las nuevas variables se agrupan en conjuntos de tres variables, sin repetición, formando nuevos conjuntos de datos. Estos conjuntos se someten a modelos de aprendizaje automático, incluyendo Support Vector Machines (SVM), Logistic Regression, Naive Bayes, Random Forest y Decision Trees.

El mejor modelo identificado en esta fase se somete a ajustes de parámetros para optimizar su rendimiento y asegurar una mayor precisión en la predicción de los datos. Este proceso garantiza una selección cuidadosa y refinada de la configuración del modelo, contribuyendo a la robustez y eficacia de este.

2.13.5 Interpretación y evaluación

En esta etapa, se establecieron las métricas para evaluar la asertividad del modelo – en este caso, el *accuracy score*-, y determinar que caracterización, modelo, agrupación y cantidad de variables y ventana utilizada tuvo el mejor rendimiento.
Adicionalmente, se realizó una propuesta de implementación del modelo en lenguaje C.

Para entender de forma visual todo el proceso metodológico que se realizó, para el análisis en el dominio del tiempo, se elaboró el diagrama que se observa en la Figura 6.

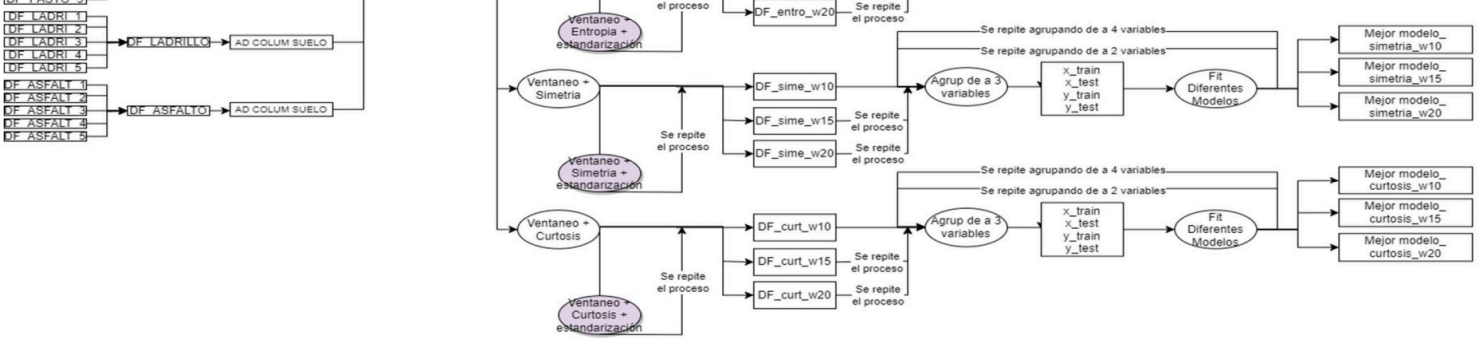


Figura 6. Diagrama análisis y procesamiento Datos.
Fuente. Elaboración Propia

3 Resultados

3.1 Caracterización por medias

Como se mencionó en el proceso metodológico, los datos fueron trabajados con un proceso de segmentación en ventanas, y en cada una de ellas se calculó la media, con el fin de obtener un dataframe de variables estadísticas que posteriormente se incorporó en los modelos de machine learning. El proceso de ventaneo se realizó para cada conjunto de datos correspondiente a los diferentes tipos de superficies. Además, se realizó un análisis variando el tamaño de la ventana, evaluando ventanas de 20, 15 y 10 unidades de medida.

Se analizaron visualmente a través de gráficos de dispersión, las características de los dataframes resultantes por tipo de superficie para inicialmente obtener una idea de que características presentaban una marcada variabilidad entre superficies, y, por ende, presentaban el potencial de ser valiosas para su inclusión en un modelo de predicción. Todos los gráficos de dispersión se realizaron para las posibles combinaciones de las 6 características, agrupando de a 3 de ellas. La Figura 7 y la Figura 8 son los gráficos de dispersión que arrojaron una diferenciación más representativa, siendo potenciales de selección las variables, en agrupación:

- Mean_Acel_y, Mean_Giro_x, Mean_Giro_y
- Mean_Acel_y, Mean_Acel_z, Mean_Giro_x

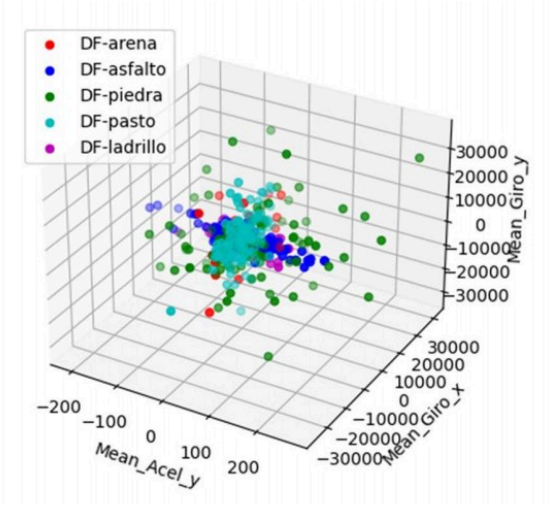


Figura 7. Gráfico de Dispersión Ay, Gx, Gy
Fuente. Elaboración propia.

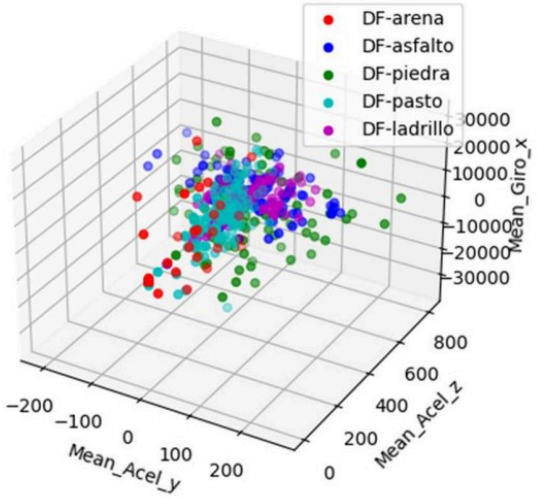


Figura 8. Gráfico de Dispersión Ay, Az, Gx
Fuente. Elaboración propia.

Tras el análisis, se entrenaron los modelos haciendo uso de los dataframes filtrados por agrupaciones de características. Las agrupaciones consistieron en todas las posibles combinaciones de 2, 3 y 4 características. Los modelos que se entrenaron fueron: KNN (o vecinos cercanos), Naive Bayes Gaussiano, SVM, Random Forest, MLP y Red Neuronal. Los modelos se configuraron usando los parámetros por defecto.
Para la división de datos de entrenamiento y prueba, se manejó un *test_size* = 0.3. Además, este proceso se realizó considerando variaciones en el tamaño de las ventanas, que incluyeron valores de 20, 15 y 10.

La Tabla 1 detalla los resultados del mejor modelo, para cada ventana y combinación de cantidad de variables, así como la combinación de variables que conforman dicho modelo y su *accuracy score*, medida de precisión que se usó para evaluar los modelos, y determinar el mejor.

Tabla 1. Resultados modelos dataframes medias.

Cantidad Variables	Mejor agrupación de variables	Mejor Modelo	Ventana	Accuracy Score
2	Mean_Acel_y, Mean_Giro_y	Random Forest	20	61.17 %
2	Mean_Acel_y, Mean_Giro_x	Random Forest	15	63.74 %
2	Mean_Acel_y, Mean_Giro_x	Random Forest	10	50.67 %
3	Mean_Acel_y, Mean_Giro_x, Mean_Giro_y	Random Forest	20	68.61 %
3	Mean_Acel_y, Mean_Giro_x, Mean_Giro_z	Random Forest	15	64.94 %
3	Mean_Acel_y, Mean_Giro_y, Mean_Giro_z	Random Forest	10	52.00 %
4	Mean_Acel_x, Mean_Acel_y, Mean_Giro_y, Mean_Giro_z	Random Forest	20	70.21 %
4	Mean_Acel_y, Mean_Acel_z, Mean_Giro_x, Mean_Giro_z	Random Forest	15	64.95 %
4	Mean_Acel_y, Mean_Acel_z, Mean_Giro_x, Mean_Giro_y	Random Forest	10	60.53 %

Fuente. Elaboración propia.

Los resultados revelaron que, para las combinaciones de 3 y 4 variables, el tamaño de ventana que presentó la mayor precisión en sus modelos fue de 20, mientras que, para una agrupación de 2 variables, fue de 15. Adicionalmente, entre los 6 modelos evaluados, independientemente de la ventana, la cantidad de variables y su agrupación, el modelo más sobresaliente fue el modelo Random Forest. También, se puede destacar que, en las combinaciones ganadoras, que incluyen ventanas de 20, 15 y 10, siempre estuvieron presentes la característica Mean_Acel_y y Mean_Giro_x.

3.1.1 Búsqueda Mejores Parámetros

Después de completar el análisis de las diversas caracterizaciones de la señal y de entrenar y evaluar varios modelos de machine learning, se seleccionó la combinación óptima de variables para cada ventana (20, 10 y 15), y se definieron diferentes funciones para encontrar la mejor combinación de parámetros de los modelos con el objetivo de mejorar el *accuracy score*. Se examinaron otros modelos adicionales, no limitándose exclusivamente al modelo ganador de esta combinación.
El resultado de esa iteración y búsqueda de los mejores parámetros, teniendo en cuenta la agrupación de variables ganadoras, dió diferentes resultados que se muestran en el apartado 3.2.1.1, 3.2.1.2 y 3.2.1.3.

3.1.1.1 Agrupación de 3 variables

La combinación más exitosa para la agrupación de 3 variables, como se observa en la Tabla 1, fue una ventana de tamaño 20, con las variables *Mean_Acel_y, Mean_Giro_x, Mean_Giro_y.* La Tabla 2 muestra los parámetros que deben ser configurados en el entrenamiento de diferentes modelos para mejorar el *accuracy score* inicial.

Tabla 2. Resultados mejores parámetros de entrenamiento de modelos para 3 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 100 max_depth: 10	69.68 %	68.61%
SVM	Kernel: rbf, C:1	44.68%	44.68 %
KNN	n_neighbors: 7 weights: 'distance'	51.06%	48.40 %
Naive Bayes	GaussianNB	47.34 %	47.34 %
MLP	Activation: 'relu' Alpha: 0.001 hidden_layer_sizes: (100,) max_iter: 200	43.08%	15.96 %

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo Random Forest, subiendo en 1.07% su precisión.

3.1.1.2 Agrupación de 2 variables

Para la agrupación de 2 variables, según los resultados reflejados en la Tabla 1, se destacó el modelo con una ventana de tamaño 15 y con las variables *Mean_Acel_y, Mean_Giro_x.* La Tabla 3, proporciona los parámetros que deben ser utilizados en el entrenamiento de diferentes modelos con el fin de mejorar el *accuracy score* inicial.

Tabla 3. Resultados mejores parámetros de entrenamiento de modelos para 2 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 100 max_depth: None	61.17%	61.17%
SVM	Kernel: rbf, C:1	39.89%	39.89%
KNN	n_neighbors: 7 weights: 'distance'	50.00%	47.87%
Naive Bayes	GaussianNB	56.91%	56.91%

MLP	Activation: 'logistic' Alpha: 0.001 hidden_layer_sizes: (100,) max_iter: 300	38.83%	15.96%

Fuente. Elaboración propia.

La búsqueda de mejores parámetros evidenció, que los valores por *default* del modelo ganador Random Forest, permiten obtener el mayor accuracy en esta combinación.

3.1.1.3 Agrupación de 4 variables

La combinación ganadora para la agrupación de 4 variables, según lo reflejado en la Tabla 1, al igual que para una combinación de 3 variables, fue una ventana de tamaño 20, con las variables *Mean_Acel_x, Mean_Acel_y, Mean_Giro_y, Mean_Giro_z*. La Tabla 4 muestra los parámetros que se deben configurar en el entrenamiento de diferentes modelos, para lograr un *accuracy score* mayor.

Tabla 4. Resultados mejores parámetros de entrenamiento de modelos para 4 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 100 max_depth: 10	73.40%	68.61%
SVM	Kernel: rbf, C:1	45.21%	45.21%
KNN	n_neighbors: 7 weights: 'distance'	54.78%	55.31%
Naive Bayes	GaussianNB	48.40%	48.40%
MLP	Activation: 'logistic' Alpha: 0.001 hidden_layer_sizes: (50,50) max_iter: 200	36.17%	15.96%

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo Random Forest, subiendo en 4.79% su precisión con los nuevos valores de n_estimators y max_depth.

3.2 Caracterización por Varianza

Los datos se sometieron a un proceso de segmentación en ventanas, calculando la varianza en cada una de ellas. De este proceso se generó un dataframe de variables estadísticas que posteriormente se incorporó en los modelos de machine learning. El proceso de ventaneo se aplicó a cada conjunto de datos correspondiente a los diferentes tipos de superficie. Además,

este análisis se realizó variando el tamaño de la ventana, considerando ventanas de 20, 15 y 10 unidades de medida.

Para identificar las características con marcada variabilidad entre superficies y con potencial para ser seleccionadas como las variables a ingresar en los modelos de predicción, se realizó un análisis visual a través de gráficos de dispersión. Todos los gráficos de dispersión se realizaron para las posibles combinaciones de las 6 características, agrupando de a 3 de ellas. Las Figura 9, Figura 10 y Figura 11, muestran los gráficos de dispersión que arrojaron una diferenciación más representativa, sugiriendo posibles selecciones de variables, en agrupaciones específicas:

- Var_Acel_x, Var_Acel_y, Var_Acel_z
- Var_Acel_x, Var_Giro_y, Var_Giro_z
- Var_Acel_z, Var_Giro_y, Var_Giro_z

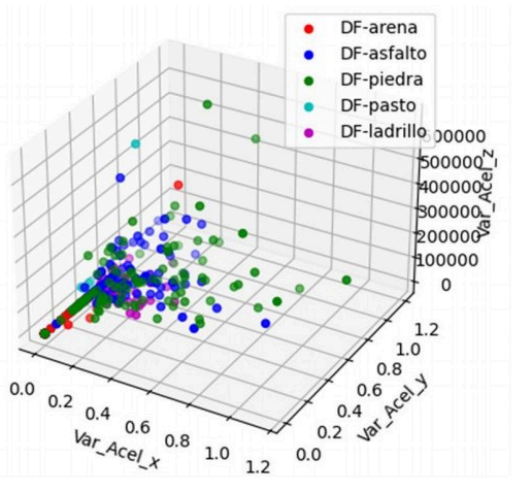


Figura 9. Gráfico de Dispersión Ax, Ay, Az
. Fuente.: Elaboración propia.

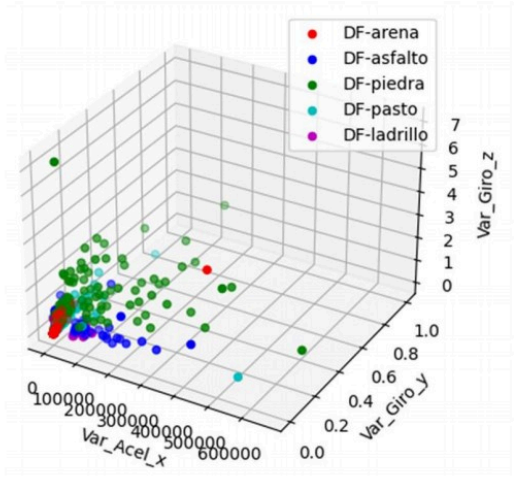


Figura 10. Gráfico de Dispersión Ax, Gy, Gz
Fuente. Elaboración propia.

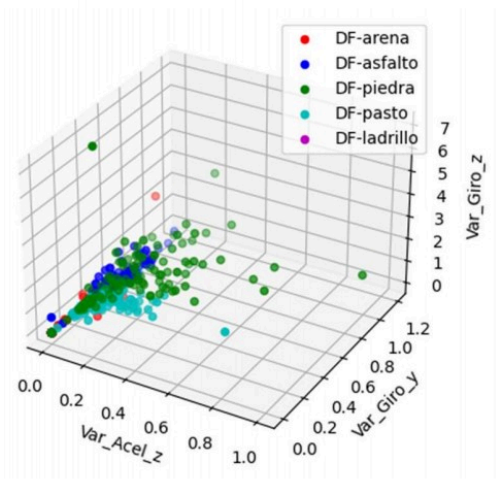


Figura 11. Gráfico de Dispersión Az, Gy, Gz
Fuente. Elaboración propia.

Después de realizar el análisis, se procedió a entrenar los modelos utilizando los dataframes filtrados según agrupaciones de características. Estas agrupaciones consideraron todas las posibles combinaciones de 2, 3 y 4 características, al igual que en el caso de la media. Los modelos que se entrenaron incluyeron: KNN (k-vecinos más cercanos), Naive Bayes Gaussiano, SVM (máquinas de soporte vectorial), Random Forest, MLP (perceptrón multicapa) y Red Neuronal.

En cuanto a la configuración de los modelos, se utilizaron los parámetros por defecto, siguiendo el mismo enfoque. Para la división de datos de entrenamiento y prueba, se mantuvo el *test_size* = 0.3, al igual que en el caso anterior. Cabe destacar que este proceso también se realizó considerando variaciones en el tamaño de las ventanas, que incluyeron valores de 20, 15 y 10.

Tabla 5. Resultados modelos dataframes varianzas.

Cantidad Variables	**Mejor agrupación de variables**	**Mejor Modelo**	**Ventana**	**Accuracy Score**
2	Var_Acel_y, Var_Giro_x	Random Forest	20	68.68 %
2	Var_Acel_y, Var_Giro_x	Random Forest	15	66.93 %
2	Var_Acel_y, Var_Giro_z	Random Forest	10	60.27 %
3	Var_Acel_x, Var_Giro_y, Var_Giro_z	Random Forest	20	72.34 %
3	Var_Acel_y, Var_Giro_y, Var_Giro_z	Random Forest	15	72.11 %
3	Var_Acel_x, Var_Giro_y, Var_Giro_z	Random Forest	10	66.93 %
4	Var_Acel_x, Var_Acel_z, Var_Giro_y, Var_Giro_z	Random Forest	20	73.40 %

4	Var_Acel_y, Var_Acel_z, Var_Giro_x, Var_Giro_z	Random Forest	15	73.70 %
4	Var_Acel_y, Var_Acel_z, Var_Giro_x, Var_Giro_z	Random Forest	10	70.67 %

Fuente. Elaboración propia.

La Tabla 5 muestra los resultados del mejor modelo, para cada ventana y combinación de cantidad de variables, así como la combinación de variables que conforman dicho modelo y su *accuracy score,* medida de precisión que se usó para evaluar los modelos, y determinar el mejor.

Los resultados reflejaron que, para las combinaciones de 2 y 3 variables, el tamaño de ventana que presentó la mayor precisión en sus modelos fue de 20, mientras que, para una agrupación de 4 variables, fue de tamaño 15. Adicionalmente, entre los 6 modelos evaluados, independientemente de la ventana, la cantidad de variables y su agrupación, el modelo más sobresaliente fue el modelo Random Forest, al igual que para el análisis de la caracterización por media. Cabe destacar que, en las combinaciones ganadoras, para las ventanas de 20 y 15, se mantuvieron las características Var_Giro_x y Var_Giro_y constantes. La característica Var_Giro_x, también se mantuvo para la combinación de dos variables, en ventanas de 20 y 15.

3.2.1 Búsqueda de Mejores Parámetros

Tras completar el análisis de la caracterización de la señal y entrenar y evaluar varios modelos de machine learning, se seleccionó la mejor combinación de variables para cada ventana (20, 10 y 15). Luego, se realizó la búsqueda de los mejores parámetros de cada modelo, con el objetivo de mejorar el *accuracy score*, teniendo como insumo las mejores combinaciones de variables mencionadas. Se evaluaron otros modelos adicionales, al igual que en el caso de la media, sin limitarse al modelo ganador de esta combinación.
El resultado del proceso de iteración y búsqueda de los mejores parámetros condujo a los resultados plasmados en las secciones siguientes.

3.2.1.1 Agrupación de 3 variables

La combinación más exitosa para la agrupación de 3 variables, como se observa en la Tabla 5, fue una ventana de tamaño 20, con las variables *Var_Acel_y, Mean_Giro_y, Mean_Giro_z.* La Tabla 6 muestra los parámetros que deben ser usados en el entrenamiento de diferentes modelos para mejorar el *accuracy score.*

Tabla 6. Resultados mejores parámetros de entrenamiento de modelos para 3 variables.

Modelo	Mejores Parámetros	Accuracy Score – Mejores Parámetros	Accuracy Score Inicial
Random Forest	n_estimators: 50 max_depth: 10	75.00 %	72.34%
SVM	Kernel: rbf, C:10	57.97%	51.06 %
KNN	n_neighbors: 9 weights: 'uniform'	60.64%	57.98 %
Naive Bayes	GaussianNB	47.34 %	47.34 %
MLP	Activation: 'tanh' Alpha: 0.001 hidden_layer_sizes: (100,100) max_iter: 100	38.82%	31.91 %

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo Random Forest, subiendo en 2.66% su precisión, con la variación de parámetros.

3.2.1.2 Agrupación de 2 variables

Para la agrupación de 2 variables, según los resultados reflejados en la Tabla 5, al igual que para la agrupación de 3 variables, se destacó el modelo con una ventana de tamaño 20 y con las variables *Mean_Acel_y, Mean_Giro_x*. La Tabla 7, proporciona los parámetros que deben ser usados en el entrenamiento de diferentes modelos con el fin de mejorar el *accuracy score*.

Tabla 7. Resultados mejores parámetros de entrenamiento de modelos para 2 variables.

Modelo	Mejores Parámetros	Accuracy Score – Mejores Parámetros	Accuracy Score Inicial
Random Forest	n_estimators: 100 max_depth: 10	71.81%	68.68%
SVM	Kernel: rbf, C:10	45.21%	42.02%
KNN	n_neighbors: 7 weights: 'uniform	52.65%	52.65%
Naive Bayes	GaussianNB	38.83%	38.83%
MLP	Activation: 'relu' Alpha: 0.01 hidden_layer_sizes: (50,50) max_iter: 100	34.04%	18.08%

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo Random Forest, subiendo en 3.13% su precisión, con la variación de parámetros de max_depth de 'None' a 10.

3.2.1.3 Agrupación de 4 variables

La combinación ganadora para la agrupación de 4 variables, según lo reflejado en la Tabla 5, fue una ventana de tamaño 15, con las variables *Var_Acel_y, Var_Acel_z, Var_Giro_x, Var_Giro_z*. La Tabla 8, muestra los parámetros que se deben configurar en el entrenamiento de diferentes modelos, para lograr un *accuracy score* mayor.

Tabla 8. Resultados mejores parámetros de entrenamiento de modelos para 4 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 100 max_depth: 10	74.50%	73.70%
SVM	Kernel: rbf, C:10	57.77%	49.00%
KNN	n_neighbors: 9 weights: 'uniform'	53.78%	52.59%
Naive Bayes	GaussianNB	65.33%	65.33%
MLP	Activation: 'tanh' Alpha: 0.01 hidden_layer_sizes: (50,50) max_iter: 100	32.67%	21.52%

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo Random Forest, subiendo en 0.8% su precisión con los nuevos valores de n_estimators y max_depth.

3.3 Caracterización por Simetría

Los datos se sometieron a un proceso de segmentación en ventanas, calculando la simetría en cada una de ellas. De este proceso se generó un dataframe de variables estadísticas que fue el input en los modelos de machine learning. El proceso de ventaneo se aplicó a cada conjunto de datos correspondiente a los 5 tipos de superficie. El análisis se realizó variando el tamaño de la ventana, considerando ventanas de 20, 15 y 10 unidades de medida.

Para identificar las características con marcada variabilidad entre superficies y con potencial para ser seleccionadas como las variables a ingresar en los modelos de predicción, se realizó un análisis visual a través de gráficos de dispersión. Se generaron gráficos de dispersión para todas las posibles combinaciones de las 6 características, agrupándolas de a 3 en 3. Las Figura 12, Figura 13 y Figura 14 , muestran los gráficos de dispersión que presentaron una diferenciación más notoria, sugiriendo como posibles variables a seleccionar:

- Simetria_Acel_x, Simetria_Acel_z, Simetria_Giro_x
- Simetria_Acel_z, Simetria_Acel_y, Simetria_Giro_z
- Simetria_Acel_x, Simetria_Acel_z, Simetria_Acel_y

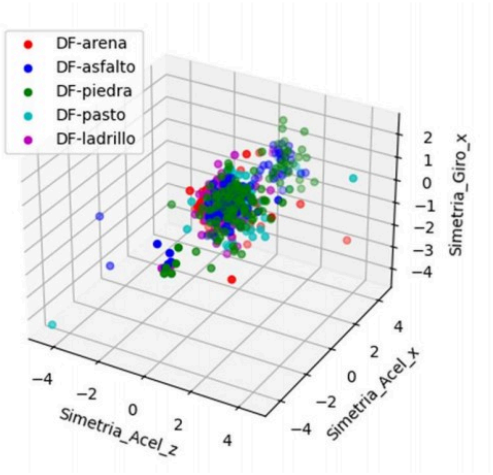


Figura 12. Gráfico de Dispersión Az, Ax, Gx
Fuente. Elaboración propia.

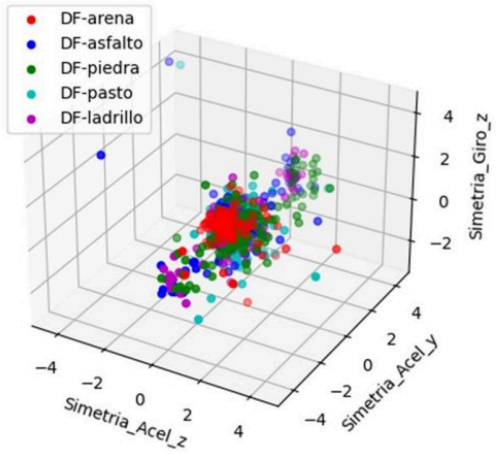


Figura 13. Gráfico de Dispersión Az Ay, Gz
Fuente. Elaboración propia.

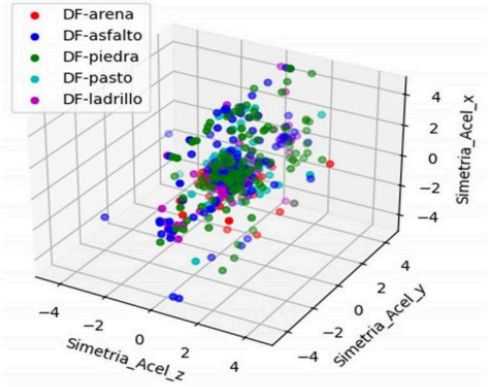


Figura 14. Gráfico de Dispersión Ax, Ay, Az
Fuente. Elaboración propia.

Al igual que para el enfoque de la media y la varianza, posterior al análisis, se procedió a entrenar los modelos utilizando los dataframes filtrados según agrupaciones de características. Estas agrupaciones consideraron todas las posibles combinaciones de 2, 3 y 4 características. Los modelos que se entrenaron incluyeron: KNN (k-vecinos más cercanos), Naive Bayes Gaussiano, SVM (máquinas de soporte vectorial), Random Forest, MLP (perceptrón multicapa) y Red Neuronal, replicando la elección de modelos.

En cuanto a la configuración de los modelos, se utilizaron los parámetros por defecto, manteniendo la coherencia con el enfoque previamente establecido. Para la división de datos de entrenamiento y prueba, se mantuvo el *test_size* = 0.3. Cabe destacar que este proceso también se realizó considerando variaciones en el tamaño de las ventanas, que incluyeron valores de 20, 15 y 10.

Tabla 9. Resultados modelos dataframes simetrías.

Cantidad Variables	**Mejor agrupación de variables**	**Mejor Modelo**	**Ventana**	**Accuracy Score**
2	Simetria_Acel_x, Simetria_Acel_z	SVM	20	38.30 %
2	Simetria_Acel_x, Simetria_Acel_y	SVM	15	37.45 %
2	Simetria_Acel_z, Simetria_Giro_x	Random Forest	10	34.67 %
3	Simetria_Acel_y, Simetria_Acel_z, Simetria_Giro_x	SVM	20	39.89 %
3	Simetria_Acel_x, Simetria_Acel_y, Simetria_Acel_z	SVM	15	43.42 %
3	Simetria_Acel_y, Simetria_Acel_z, Simetria_Giro_x	Random Forest	10	34.93 %
4	Simetria_Acel_x, Simetria_Acel_y, Simetria_Acel_z, Simetria_Giro_z	KNN	20	42.55 %
4	Simetria_Acel_x, Simetria_Acel_y, Simetria_Acel_z, Simetria_Giro_x	SVM	15	44.22 %
4	Simetria_Acel_y, Simetria_Acel_z, Simetria_Giro_x, Simetria_Giro_z	Random Forest	10	37,86 %

Fuente. Elaboración propia.

La Tabla 9 muestra los resultados del mejor modelo, para cada ventana y combinación de cantidad de variables, así como la combinación de variables que conforman dicho modelo y su *accuracy score,* medida de precisión que se usó para evaluar los modelos, y determinar el mejor.

Los resultados reflejaron que, para las combinaciones de 3 y 4 variables, el tamaño de ventana que presentó la mayor precisión en sus modelos fue de 15, mientras que, para una agrupación

de 2 variables, fue de tamaño 20. Adicionalmente, entre los 6 modelos evaluados, independientemente de la ventana, la cantidad de variables y su agrupación, el modelo más sobresaliente fue el modelo SVM. Cabe destacar que, en las combinaciones ganadoras, para las ventanas de 15, se mantuvieron las características Simetria_Acel_y y Simetria_Acel_x constantes. La característica Simetria_Acel_x, también se mantuvo para la combinación de dos variables, en la ventana de 20 y 15.

3.3.1 Búsqueda Mejores Parámetros

Posterior al análisis de la caracterización de la señal y entrenar y evaluar varios modelos de machine learning, se seleccionó la mejor combinación de variables para cada ventana (20, 10 y 15) para poder realizar la búsqueda de los mejores parámetros de cada modelo, con el objetivo de mejorar el *accuracy score*, teniendo como insumo las mejores combinaciones de variables mencionadas. Se evaluaron otros modelos adicionales, al igual que en el caso de la media y la varianza, sin limitarse al modelo ganador de esta combinación.
Los resultados obtenidos del proceso de iteración y búsqueda de los mejores parámetros se presentan a continuación, organizados según la cantidad de variables.

3.3.1.1 Agrupación de 3 variables

La combinación más exitosa para la agrupación de 3 variables, como se observa en la Tabla 9, fue una ventana de tamaño 15, con las variables *Simetria_Acel_x, Simetria_Acel_y, Simetria_Giro_z*. La Tabla 10 muestra los parámetros que deben ser usados en el entrenamiento de diferentes modelos para mejorar el *accuracy score*.

Tabla 10. Resultados mejores parámetros de entrenamiento de modelos para 3 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 50 max_depth: 10	41.43 %	37.85%
SVM	Kernel: rbf, C:1	43.42%	43.42 %
KNN	n_neighbors: 9 weights: 'uniform'	38.24%	34.27 %
Naive Bayes	GaussianNB	33.07 %	33.07 %

Fuente. Elaboración propia.

La búsqueda de mejores parámetros evidenció que, los valores por *default* del modelo ganador SVM, es decir, haciendo uso de un kernel: 'rbf' y un C:1, permiten obtener el mayor accuracy en esta combinación.

3.3.1.2 Agrupación de 2 variables

Para la agrupación de 2 variables, según los resultados reflejados en la Tabla 9, se destacó el modelo con una ventana de tamaño 20 y con las variables *Simetria_Acel_y, Simetria_Giro_z.* La Tabla 11 , proporciona los parámetros que deben ser usados en el entrenamiento de diferentes modelos con el fin de mejorar el *accuracy score.*

Tabla 11. Resultados mejores parámetros de entrenamiento de modelos para 2 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 150 max_depth: 10	36.17%	35.11%
SVM	Kernel: rbf, C:1	38.30%	38.30%
KNN	n_neighbors: 9 weights: 'distance'	35.64%	30.32%
Naive Bayes	GaussianNB	31.38%	31.38%

Fuente. Elaboración propia.

Al igual que para la agrupación de 3 variables, la búsqueda de mejores parámetros mostró que los valores por *default* del modelo ganador SVM –kernel:'rbf', C:1- permiten obtener el mayor accuracy.

3.3.1.3 Agrupación de 4 variables

La combinación ganadora para la agrupación de 4 variables, según lo reflejado en la Tabla 9, fue una ventana de tamaño 15, al igual que para la agrupación de 3 variables, con las variables *Simetria_Acel_x, Simetria_Acel_z, Simetria_Giro_x, Simetria_Giro_z.* La Tabla 12, muestra los parámetros que se deben configurar en el entrenamiento de diferentes modelos, para lograr un *accuracy score* mayor.

Tabla 12. Resultados mejores parámetros de entrenamiento de modelos para 4 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 100 max_depth: 10	40.24%	38.25%
SVM	Kernel: rbf, C:1	44.22%	44.22%
KNN	n_neighbors: 5 weights: 'distance'	39.84%	37.45%
Naive Bayes	GaussianNB	39.44%	39.44%

Fuente. Elaboración propia.

De manera similar a la agrupación de 2 y 3 variables, la búsqueda de mejores parámetros evidenció, que los valores por *default* del modelo ganador SVM, permiten obtener el mayor

accuracy en esta combinación.

3.4 Caracterización por Curtosis

Siguiendo el mismo enfoque de las 3 caracterizaciones anteriores, los datos fueron sometidos a un proceso de segmentación en ventanas, calculando la curtosis en cada una de ellas. De este proceso resultó un dataframe de variables estadísticas que fue la entrada para los modelos de machine learning. El proceso de ventaneo se aplicó a cada conjunto de datos correspondiente a los 5 tipos de superficie. El análisis se realizó variando el tamaño de la ventana, considerando ventanas de 20, 15 y 10 unidades de medida.

Para identificar las características una variabilidad notoria entre superficies y con potencial para ser seleccionadas como las variables a ingresar en los modelos de predicción, se realizó un análisis visual a través de gráficos de dispersión. Se generaron gráficos de dispersión para todas las posibles combinaciones de las 6 características, en agrupaciones de a 3 en 3. Las Figura 15, Figura 16 y Figura 17 , muestran los gráficos de dispersión que presentaron una diferenciación más notoria, sugiriendo como posibles variables a seleccionar:

- Curtosis_Acel_z, Curtosis_Acel_x, Curtosis_Acel_y
- Curtosis_Acel_z, Curtosis_Acel_y, Curtosis_Giro_x
- Curtosis_Acel_x, Curtosis_Acel_y, Curtosis_Giro_z

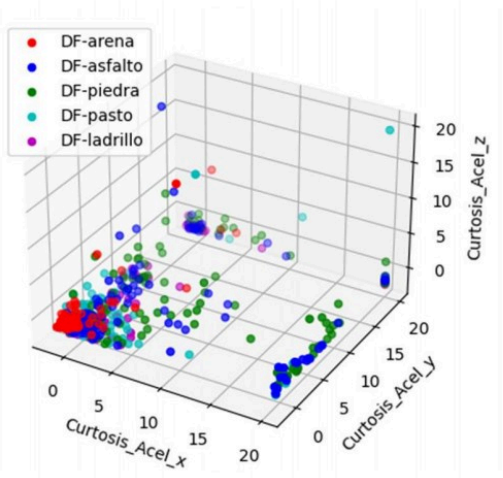


Figura 15. Gráfico de Dispersión Az, Ax, Ay
Fuente. Elaboración propia.

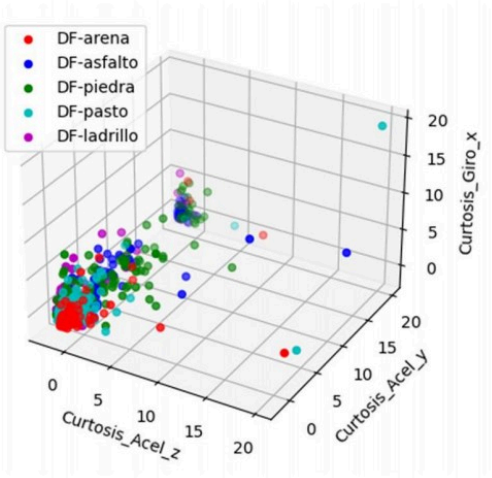


Figura 16. Gráfico de Dispersión Az, Ay, Gx
Fuente. Elaboración propia.

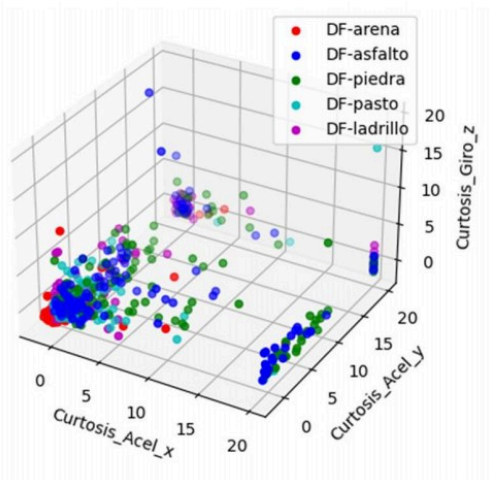


Figura 17. Gráfico de Dispersión Ax, Ay, Gz
Fuente. Elaboración propia.

Al igual que para los anteriores enfoques, posterior al análisis se procedió a entrenar los modelos utilizando los dataframes filtrados según agrupaciones de características. Estas agrupaciones consideraron todas las posibles combinaciones de 2, 3 y 4 características. Los modelos que se entrenaron incluyeron: KNN (k-vecinos más cercanos), Naive Bayes Gaussiano, SVM (máquinas

de soporte vectorial), Random Forest, MLP (perceptrón multicapa) y Red Neuronal, replicando la elección de modelos.

En cuanto a la configuración de los modelos, se utilizaron los parámetros por defecto, manteniendo la coherencia con el enfoque previamente establecido. Para la división de datos de entrenamiento y prueba, se mantuvo un *test_size* = 0.3. Cabe destacar que este proceso también se realizó considerando variaciones en el tamaño de las ventanas, que incluyeron valores de 20, 15 y 10.

La Tabla 13 muestra los resultados del mejor modelo, para cada ventana y combinación de cantidad de variables, así como la combinación de variables que conforman dicho modelo y su *accuracy score*, medida de precisión que se usó para evaluar los modelos, y determinar el mejor.

Los resultados reflejaron que, para las combinaciones de 2, 3 y 4 variables, el tamaño de ventana que presentó la mayor precisión en sus modelos fue de 20. Adicionalmente, entre los 6 modelos evaluados, para las ventanas de 20, en la combinación de 2 y 3 variables, el modelo más sobresaliente fue el modelo SVM, a diferencia, de la combinación de 4 variables, dónde el modelo KNN obtuvo una mejor precisión. La característica Curtosis_Giro_y, se mantuvo para la combinación de 2 y 3 variables.

Tabla 13. Resultados modelos dataframes curtosis.

Cantidad Variables	**Mejor agrupación de variables**	**Mejor Modelo**	**Ventana**	**Accuracy Score**
2	Curtosis_Acel_z, Curtosis_Giro_y	SVM	20	45.21 %
2	Curtosis_Acel_x, Curtosis_Acel_y	KNN	15	30.04 %
2	Curtosis_Acel_x, Curtosis_Acel_z	Random Forest	10	41.07 %
3	Curtosis_Acel_y, Curtosis_Giro_y, Curtosis_Giro_z	SVM	20	46.28 %
3	Curtosis_Acel_x, Curtosis_Acel_y, Curtosis_Acel_z	KNN	15	43.02 %
3	Curtosis_Acel_x, Curtosis_Acel_z, Curtosis_Giro_y	Random Forest	10	41.07 %
4	Curtosis_Acel_x, Curtosis_Acel_y, Curtosis_Acel_z, Curtosis_Giro_x	KNN	20	48.94 %
4	Curtosis_Acel_x, Curtosis_Acel_y, Curtosis_Acel_z, Curtosis_Giro_x	SVM	15	42.63 %
4	Curtosis_Acel_y, Curtosis_Acel_z, Curtosis_Giro_x, Curtosis_Giro_z	Random Forest	10	40.53 %

Fuente. Elaboración propia.

3.4.1 Búsqueda Mejores Parámetros

Tras completar el análisis de la caracterización de la señal y entrenar y evaluar varios modelos de machine learning, se seleccionó la mejor combinación de variables para cada ventana (20, 10 y 15). Luego, se realizó la búsqueda de los mejores parámetros de cada modelo, con el objetivo de mejorar el *accuracy score*, teniendo como insumo las mejores combinaciones de variables mencionadas. Se evaluaron otros modelos adicionales, al igual que en el caso de la media, sin limitarse al modelo ganador de esta combinación.
El resultado del proceso de iteración y búsqueda de los mejores parámetros se plasmó en las secciones siguientes.

3.4.1.1 Agrupación de 3 variables

La combinación más exitosa para la agrupación de 3 variables, como se observa en la Tabla 13, fue una ventana de tamaño 15, con las variables *Curtosis_Acel_z*, *Curtosis_Giro_y*, *Curtosis_Giro_z*. La Tabla 14 muestra los parámetros que deben ser usados en el entrenamiento de diferentes modelos para mejorar el *accuracy score*.

Tabla 14. Resultados mejores parámetros de entrenamiento de modelos para 3 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 50 max_depth: 10	42.02 %	39.36%
SVM	Kernel: rbf, C:1	46.28%	46.28%
KNN	n_neighbors: 3 weights: 'distance'	42.55%	35.10%
Naive Bayes	GaussianNB	37.23 %	33.23 %

Fuente. Elaboración propia.

Al igual que para la simetría, la búsqueda de mejores parámetros mostró que los valores por *default* del modelo ganador SVM –kernel:'rbf', C:1- permiten obtener el mayor accuracy.

3.4.1.2 Agrupación de 2 variables

Para la agrupación de 2 variables, según los resultados reflejados en la Tabla 13, se destacó el modelo con una ventana de tamaño 20 y con las variables *Curtosis_Acel_z*, *Curtosis_Giro_y*. La Tabla 15, proporciona los parámetros que deben ser usados en el entrenamiento de diferentes modelos con el fin de mejorar el *accuracy score*.

Tabla 15. Resultados mejores parámetros de entrenamiento de modelos para 2 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**

Random Forest	n_estimators: 100 max_depth: None	36.70%	36.70%
SVM	Kernel: rbf, C:1	45.21%	45.21%
KNN	n_neighbors: 9 weights: 'uniform'	40.96%	39.46%
Naive Bayes	GaussianNB	31.38%	31.38%

Fuente. Elaboración propia.

Al igual que para la agrupación de 3 variables, la búsqueda de mejores parámetros mostró que los valores por *default* del modelo ganador SVM –kernel:'rbf', C:1- permiten obtener el mayor accuracy.

3.4.1.3 Agrupación de 4 variables

La combinación ganadora para la agrupación de 4 variables, según lo reflejado en la Tabla 13, fue una ventana de tamaño 20, con las variables *Curtosis_Acel_x, Curtosis_Acel_y, Curtosis_Acel_z, Curtosis_Giro_x*. La Tabla 16, muestra los parámetros que se deben configurar en el entrenamiento de diferentes modelos, para lograr un *accuracy score* mayor.

Tabla 16. Resultados mejores parámetros de entrenamiento de modelos para 4 variables.

Modelo	**Mejores Parámetros**	**Accuracy Score – Mejores Parámetros**	**Accuracy Score Inicial**
Random Forest	n_estimators: 50 max_depth: None	46.27%	44.15%
SVM	Kernel: rbf, C:10	44.14%	40.42%
KNN	n_neighbors: 7 weights: 'distance'	50.00%	48.94%
Naive Bayes	GaussianNB	35.11%	35.11%

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo KNN, subiendo en 1.06% su precisión variando el parámetro weights de 'uniform' -por default-, a 'distance'.

3.5 Caracterización por Entropía

Para la última caracterización, y siguiendo el enfoque de las anteriores caracterizaciones, los datos fueron sometidos a un proceso de segmentación en ventanas, calculando la curtosis en cada una de ellas. De este proceso resultó un dataframe de variables estadísticas que fue la entrada para los modelos de machine learning. El proceso de ventaneo se aplicó a cada conjunto de datos

correspondiente a los 5 tipos de superficie. El análisis se realizó variando el tamaño de la ventana, considerando ventanas de 20, 15 y 10 unidades de medida.

Se realizó un análisis visual a través de gráficos de dispersión, con el fin de identificar las características con una variabilidad notoria entre superficies y con potencial para ser seleccionadas como las variables a ingresar en los modelos de predicción. Se generaron gráficos de dispersión para todas las posibles combinaciones de las 6 características, en agrupaciones de a 3 en 3. La Figura 18 muestra el gráfico de dispersión de la combinación de variables más representativa:

- Entropia_Acel_x, Entropia_Acel_y, Entropia_Acel_z

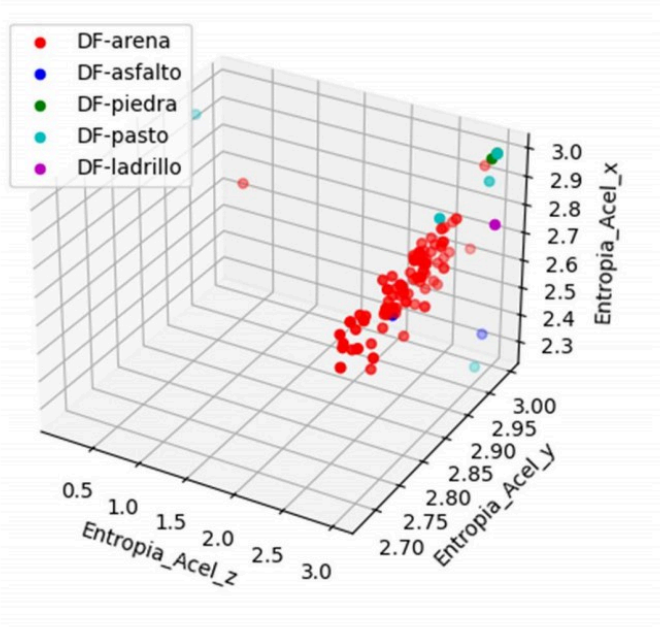


Figura 18. Gráfico de Dispersión Ax, Ay, Az
Fuente. Elaboración propia.

La Tabla 17 muestra los resultados del mejor modelo, para cada ventana y combinación de cantidad de variables, así como la combinación de variables que conforman dicho modelo y su *accuracy score*, medida de precisión que se usó para evaluar los modelos, y determinar el mejor.

Tabla 17. Resultados modelos dataframes entropía.

Cantidad Variables	**Mejor agrupación de variables**	**Mejor Modelo**	**Ventana**	**Accuracy Score**
2	Entropia_Acel_z, Entropia_Giro_y	KNN	20	52.13%
2	Entropia_Acel_x, Entropia_Acel_z	KNN	15	40.22%
2	Entropia_Acel_x, Entropia_Acel_z	KNN	10	50.93 %

3	Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_y	KNN	20	52.13 %
3	Entropia_Acel_x, Entropia_Acel_z, Entropia_Acel_y	KNN	15	44.62%
3	Entropia_Acel_x, Entropia_Acel_y, Entropia_Acel_z	KNN	10	50.67%
4	Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_x, Entropia_Giro_y	KNN	20	52.13%
4	Entropia_Acel_x, Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_x	KNN	15	44.22%
4	Entropia_Acel_x, Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_x	KNN	10	50.67 %

Fuente. Elaboración propia.

Los resultados reflejaron que, para las combinaciones de 2, 3 y 4 variables, el tamaño de ventana que presentó la mayor precisión en sus modelos fue de 20. Adicionalmente, entre los 6 modelos evaluados, independientemente de la ventana, la cantidad de variables y su agrupación, el modelo más sobresaliente fue el modelo KNN. Las características Entropia_Acel_x, Entropia_Acel_y, Entropia_Acel_z, se mantuvieron para las combinaciones 3 y 4 variables.

3.5.1 Búsqueda Mejores Parámetros

Tras completar el análisis de la caracterización de la señal y entrenar y evaluar varios modelos de machine learning, se seleccionó la mejor combinación de variables para cada ventana (20, 10 y 15). Luego, se realizó la búsqueda de los mejores parámetros de cada modelo, con el objetivo de mejorar el *accuracy score*, teniendo como insumo las mejores combinaciones de variables mencionadas. Se evaluaron otros modelos adicionales, al igual que en el caso de la media, sin limitarse al modelo ganador de esta combinación.

El resultado del proceso de iteración y búsqueda de los mejores parámetros se expone en las secciones siguientes 3.6.1.1, 3.6.1.2 y 3.6.1.3.

3.5.1.1 Agrupación de 3 variables

La combinación más exitosa para la agrupación de 3 variables, como se observa en la Tabla 17, fue una ventana de tamaño 20, con las variables *Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_y.*

La Tabla 18 muestra los parámetros que deben ser usados en el entrenamiento de diferentes modelos para mejorar el *accuracy score*.

Tabla 18. Resultados mejores parámetros de entrenamiento de modelos para 3 variables.

Modelo	Mejores Parámetros	Accuracy Score – Mejores Parámetros	Accuracy Score Inicial
Random Forest	n_estimators: 100 max_depth: 10	44.13 %	42.02%
SVM	Kernel: rbf, C:1	42.02%	42.02%
KNN	n_neighbors: 7 weights: 'unifomr'	52.13%	52.13%
Naive Bayes	GaussianNB	24.47 %	24.47 %

Fuente. Elaboración propia.

Con la mejora de parámetros, el modelo ganador siguió siendo KNN, mostrando que los parámetros por default que entrenaron el modelo permiten obtener la mayor precisión.

3.5.1.2 Agrupación de 2 variables

Para la agrupación de 2 variables, según los resultados reflejados en la Tabla 17, se destacó el modelo con una ventana de tamaño 20 y con las variables Entropia_*Acel_z, Entropia_Giro_y*. La Tabla 19, proporciona los parámetros que deben ser usados en el entrenamiento de diferentes modelos con el fin de mejorar el *accuracy score*.

Tabla 19. Resultados mejores parámetros de entrenamiento de modelos para 2 variables.

Modelo	Mejores Parámetros	Accuracy Score – Mejores Parámetros	Accuracy Score Inicial
Random Forest	n_estimators: 100 max_depth: None	42.55%	41.49%
SVM	Kernel: rbf, C:1	42.02%	42.02%
KNN	n_neighbors: 9 weights: 'uniform'	52.13%	52.13%
Naive Bayes	GaussianNB	26.07%	26.07%

Fuente. Elaboración propia.

Con la mejora de parámetros, al igual que para la agrupación de 3 variables, el modelo ganador siguió siendo KNN, mostrando que los parámetros por default que entrenaron el modelo permiten obtener la mayor precisión.

3.5.1.3 Agrupación de 4 variables

La combinación ganadora para la agrupación de 4 variables, según lo reflejado en la Tabla 17, fue una ventana de tamaño 20, con las variables *Entropia_Acel_y, Entropia_Acel_z, Entropia_Giro_x, Entropia_Giro_y*. La Tabla 20, muestra los parámetros que se deben configurar en el entrenamiento de diferentes modelos, para lograr un *accuracy score* mayor.

Tabla 20. Resultados mejores parámetros de entrenamiento de modelos para 4 variables.

Modelo	Mejores Parámetros	Accuracy Score – Mejores Parámetros	Accuracy Score Inicial
Random Forest	n_estimators: 50 max_depth: None	42.55%	42.02%
SVM	Kernel: rbf, C:10	40.96%	40.96%
KNN	n_neighbors: 7 weights: 'uniform'	52.13%	52.13%
Naive Bayes	GaussianNB	22.34%	22.34%

Fuente. Elaboración propia.

Con la mejora de parámetros, y al igual que para la agrupación de 2 y 3 variables, el modelo ganador siguió siendo KNN, mostrando que los parámetros por default que entrenaron el modelo permiten obtener la mayor precisión.

3.6 Análisis de componentes principales

En la fase de preprocesamiento de los datos, se aplicó el Análisis de Componentes Principales (PCA) al conjunto de datos con el fin de reducir la dimensionalidad y mejorar la eficiencia del proceso de clasificación. Se seleccionaron las cuatro componentes principales más significativas, lo que permitió mantener al menos el 95% de la varianza de los datos originales. Esta reducción de dimensionalidad se realizó con el objetivo de simplificar los datos sin perder información crítica, lo que facilita el proceso de modelado y clasificación. Las 4 componentes principales resultantes se utilizaron como características para los modelos de clasificación.

Además, como parte del proceso de análisis y con el propósito de una comprensión visual más profunda de las relaciones en los datos, se procedió a la representación gráfica de las componentes principales, para una mejor ilustración gráfica solo se toman las 3 componentes principales, de modo que se puede representar en un diagrama de tres dimensiones. Estas representaciones gráficas permitieron explorar de manera visual la distribución de las clases y las relaciones entre las características en un espacio tridimensional. Este enfoque de análisis visual proporcionó información valiosa sobre la estructura de los datos y su capacidad para diferenciar los tipos de superficies como se muestra en la Figura 19.

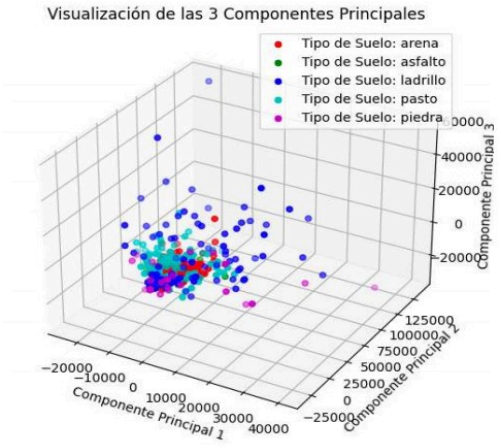


Figura 19. Gráfica de tres componentes principales.
Fuente. Elaboración propia.

La Tabla 21 presenta los resultados de clasificación para cuatro modelos de aprendizaje automático a los cuales se les aplicaron las 4 componentes principales resultantes del Análisis de Componentes Principales (PCA). Estos modelos incluyen Naive Bayes, SVM (Máquinas de Soporte Vectorial), Árbol de Decisión y Random Forest. Cada modelo fue evaluado en función de diversas métricas de rendimiento, que incluyen Accuracy (exactitud), Precision (precisión), Recall (recuperación) y F1 Score (puntuación F1). Estas métricas proporcionan una medida integral del rendimiento de los modelos en términos de su capacidad para clasificar las superficies de un rover utilizando las características reducidas por PCA. Los resultados muestran las tasas de precisión y recuperación de cada modelo, lo que permite identificar cuál de ellos obtuvo el mejor rendimiento en la clasificación de las superficies.

El mejor modelo, basado en los resultados proporcionados en la tabla, es el "Árbol de Decisión," ya que obtuvo la mayor precisión (Accuracy) de 0.624. Esto significa que el modelo de Árbol de Decisión tuvo el mejor rendimiento en la clasificación de las superficies de un rover utilizando las 4 componentes principales resultantes del PCA en comparación con los otros modelos evaluados.

Tabla 21. Resultado modelos con PCA con 4 componentes principales.

Modelo	**Precision**	**Recall**
Naive Bayes	0.605562	0.600
SVM	0.608967	0.600
Árbol de Decisión	0.621327	0.624

Random Forest	0.614721	0.592

Fuente. Elaboración propia.

Como se muestra en la Tabla 21, los valores de medición de métricas con los diferentes modelos no tienen resultados muy significativos, por lo anterior no se continua el análisis de clasificación usando componentes principales.

3.7 Caracterización usando transformada de Fourier

Como se menciona en la sección de metodología para el análisis en el dominio de la frecuencia, 2.13.3.2 y 2.13.4.2 el rango de ventas se determinó entre 50 a 150 muestras, primeramente, se usaron las ventanas de 50, 70, 100, 120 y 150, con desplazamientos entre el 10% y el 30% del valor de la ventana. En la Tabla 22 se muestra una visión de los resultados obtenidos para una ventana de 50 muestras y diferentes desplazamientos, con los 3 mejores resultados obtenidos, puntuados por la métrica de precisión y con la combinación de variables.

Tabla 22. Mejores modelos con ventanas de 50 muestras.

Configuración	Mejor Modelo	Mejor Puntaje	Combinación de Variables
Ventana de 50 muestras con desplazamiento de 5 muestras	Random Forest	0.86	Ax_Total_Power, Gy_Total_Power, Gz_Total_Power
	Random Forest	0.85	Ax_Total_Power, Ay_Total_Power, Gx_Total_Power
	Random Forest	0.82	Ax_Bandwidth, Ax_Total_Power, Gx_Total_Power
Ventana de 50 muestras con desplazamiento de 13 muestras	Random Forest	0.79	Ay_Total_Power, Gy_Total_Power, Gz_Total_Power
	Random Forest	0.78	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.77	Ax_Total_Power, Ay_Total_Power, Gx_Total_Power
Ventana de 50 muestras con desplazamiento de 30 muestras	Random Forest	0.83	Ay_Total_Power, Gx_Total_Power, Gy_Spectral_Entropy
	Random Forest	0.82	Ax_Spectral_Entropy, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.80	Ax_Total_Power, Gx_Total_Power, Gz_Total_Power

Fuente. Elaboración propia.

La Tabla 23, continúa con el formato de la tabla anterior pero empleado ventadas de 70 muestras, Es importante destacar que, en todas las configuraciones evaluadas, el modelo de Random Forest emerge como el mejor modelo en términos de rendimiento.

Tabla 23. Mejores modelos con ventanas de 70 muestras.

Configuración	Mejor Modelo	Mejor Puntaje	Combinación de Variables
Ventana de 70 muestras con desplazamiento de 7 muestras	Random Forest	0.86	Ax_Total_Power, Ay_Total_Power, Gx_Total_Power
	Random Forest	0.85	Ax_Total_Power, Ax_Spectral_Entropy, Gz_Total_Power
	Random Forest	0.82	Ax_Dominant_Frequency, Ax_Total_Power, Gz_Total_Power
Ventana de 70 muestras con desplazamiento de 13 muestras	Random Forest	0.90	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.89	Ax_Total_Power, Ay_Total_Power, Gx_Total_Power
	Random Forest	0.86	Ax_Total_Power, Ax_Spectral_Entropy, Gz_Total_Power
Ventana de 70 muestras con desplazamiento de 21 muestras	Random Forest	0.85	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.84	Ax_Bandwidth, Ax_Total_Power, Gz_Total_Power
	Random Forest	0.83	Ax_Bandwidth, Ax_Total_Power, Gx_Total_Power

Fuente. Elaboración propia.

En la Tabla 24, se muestran los resultados para ventanas de 120 muestras. Observamos un patrón distintivo en cuanto a las variables más efectivas para la clasificación. Se destaca que, en múltiples configuraciones, las combinaciones más prometedoras incluyen las variables Ax_Total_Power y Ay_Total_Power. Esto sugiere que estas dos variables desempeñan un papel fundamental en la precisión de las predicciones.

Tabla 24. Mejores modelos con ventanas de 120 muestras.

Configuración	Mejor Modelo	Mejor Puntaje	Combinación de Variables
Ventana de 120 muestras con desplazamiento de 12 muestras	Random Forest	0.91	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.89	Ax_Total_Power, Ay_Dominant_Frequency, Gy_Total_Power

	Random Forest	0.87	Ax_Dominant_Frequency, Ay_Total_Power, Gz_Total_Power
Ventana de 120 muestras con desplazamiento de 24 muestras	Random Forest	0.91	Ay_Total_Power, Gy_Spectral_Entropy, Gz_Total_Power
	Random Forest	0.90	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.89	Ax_Bandwidth, Ax_Total_Power, Gx_Bandwidth
Ventana de 120 muestras con desplazamiento de 36 muestras	Random Forest	0.90	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.88	Ax_Bandwidth, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.84	Ax_Bandwidth, Ay_Total_Power, Gx_Total_Power

Fuente. Elaboración propia.

La Tabla 25 mostró los resultados obtenidos con ventanas de 150 muestras. En esta configuración, se mantuvo el patrón anterior con respecto a las variables Ax_Total_Power y Ay_Total_Power, que continuaron siendo factores clave para la clasificación efectiva. Sin embargo, se observó que el aumento en la precisión no fue significativo en comparación con las configuraciones anteriores.

El análisis de ventanas de 150 muestras destacó la importancia de encontrar un equilibrio entre la resolución temporal y la capacidad de detectar características relevantes en la señal. Esta iteración tuvo como objetivo explorar si ventanas más pequeñas podrían mejorar aún más el rendimiento del modelo.

Tabla 25. Mejores modelos con ventanas de 150 muestras.

Configuración	**Mejor Modelo**	**Mejor Puntaje**	**Combinación de Variables**
Ventana de 150 muestras con desplazamiento de 15 muestras	Random Forest	0.92	Ax_Total_Power, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.92	Ax_Bandwidth, Ay_Total_Power, Gy_Total_Power
	Random Forest	0.9	Ax_Bandwidth, Ax_Total_Power, Gx_Total_Power
Ventana de 150 muestras con desplazamiento de 30 muestras	Random Forest	0.9	Ay_Total_Power, Gx_Total_Power, Gy_Total-Power
	Random Forest	0.89	Ax_Total_Power, Gx_Total_Power, Gy_Spectral_Entropy
	Random Forest	0.87	Ax_Dominant_Frequency, Ay_Total_Power, Gy_Total-Power

Ventana de 150 muestras con desplazamiento de 45 muestras	Random Forest	0.91	Ax_Dominant_Frequency, Gy_Bandwidth, Gz_Total-Power
	Random Forest	0.87	Ax_Dominant_Frequency, Gx_Bandwidth, Gz_Total-Power
	Random Forest	0.85	Ax_Dominant_Frequency, Ay_Total_Power, Gx_Total-Power

Fuente. Elaboración propia.

Dado que una ventana de 150 muestras se consideraba relativamente grande y podría abarcar una gran parte de la señal, se decidió realizar una iteración adicional. En esta iteración, se utilizaron ventanas de 100 muestras con un desplazamiento de 10 muestras. Las variables utilizadas para el modelo fueron 'Ax_Total_Power', 'Ay_Total_Power' y 'Gy_Total_Power'. El modelo resultante logró una precisión del 0.91 en las predicciones empleando Random Forest como modelo.

El resultado del modelo obtenido tiene puntaje sospechosamente satisfactorias, es importante destacar que los modelos basados en árboles, como Random Forest, son propensos a presentar sobreajuste, lo que significa que pueden ajustarse demasiado a los datos de entrenamiento y no generalizar bien a nuevos datos. (Hao & Guo, 2019)

Como de observó en el modelo que utilizaba una ventana de 70 muestras con un paso de 21 muestras se empleaba un clasificador Random Forest, junto con las variables Ax_Total_Power, Ay_Total_Power y Gy_Total_Power, se evidenció sobreajuste, así como para otros modelos. Esto indicaba que el modelo no generalizaba adecuadamente, como se ilustra en la Figura 20.

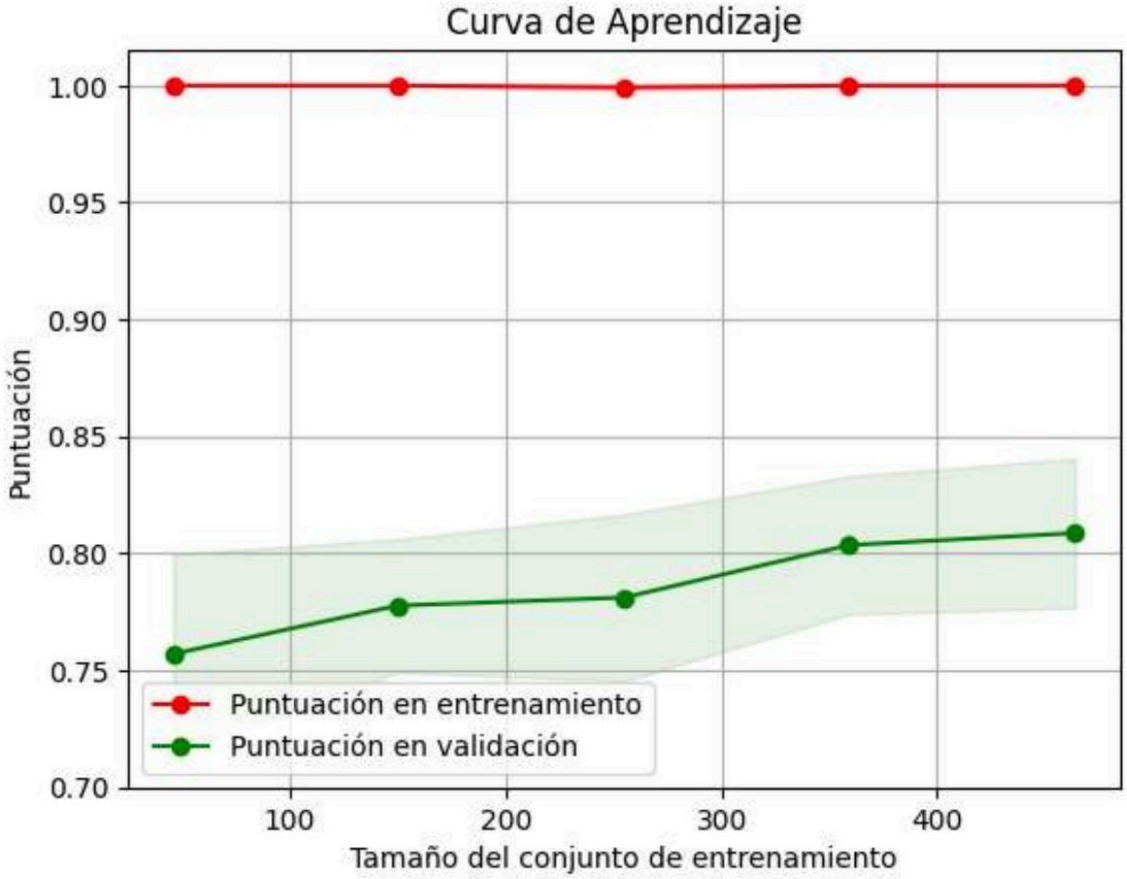


Figura 20. Curva de aprendizaje Modelo con ventana de 70 muestras y desplazamiento de 21.
Fuente. Elaboración propia.

En el estudio, se pudo notar una diferencia marcada entre el modelo con una ventana de 70 muestras y el modelo con una ventana de 100 muestras. Al realizar la misma gráfica que la Figura 20 para el modelo con ventana de 100 muestras, se observó que no presentaba la misma tendencia. Específicamente, se notó que la tendencia de la curva en la Figura 20 no era tan amplia como en la Figura 21. Esta comparación sugiere que el modelo con ventana de 100 muestras no experimenta el mismo sobreajuste, ya que la distancia entre las curvas de puntuación de entrenamiento y validación no mostraba una brecha tan significativa.

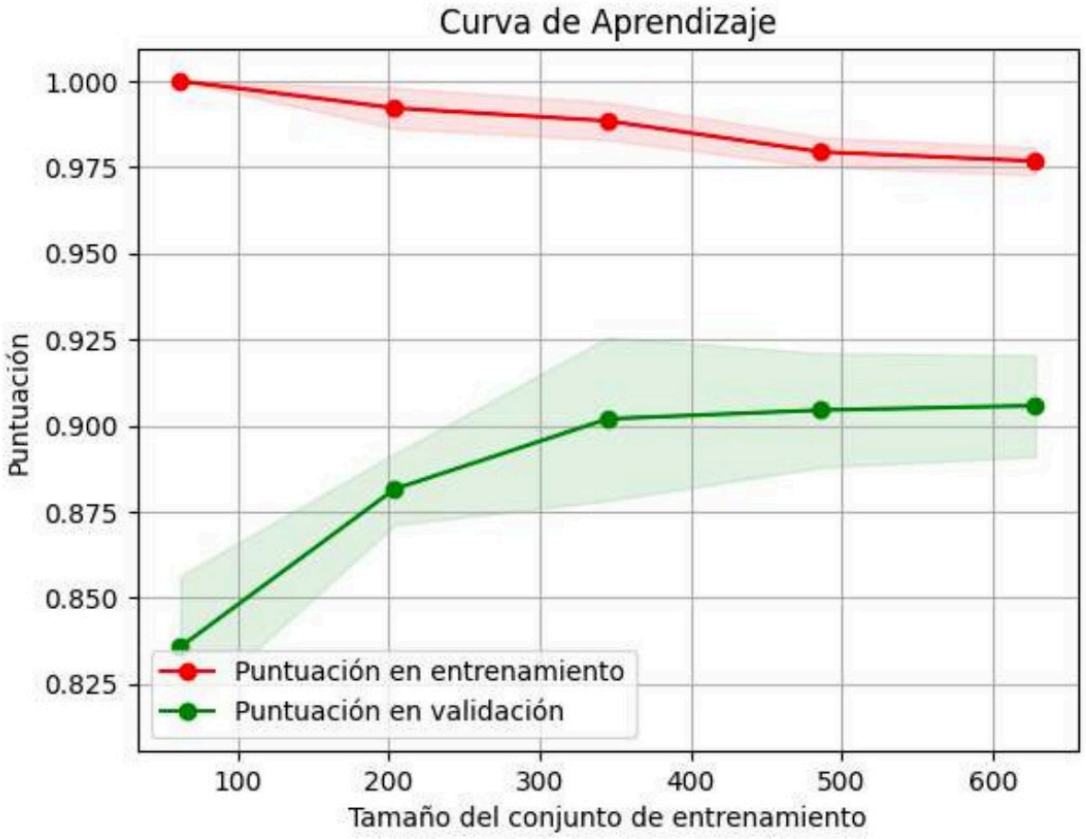


Figura 21. Curva de aprendizaje modelo con ventana de 100 muestras y desplazamiento de 10.
Fuente. Elaboración propia.

Para mejorar la capacidad del modelo de Random Forest y mitigar el sobreajuste observado, se empleó la técnica de ajuste de Parámetros. Se realizaron varias iteraciones, explorando diferentes valores para parámetros clave, como "n_estimators," "max_depth," y "min_samples_split." Tras un exhaustivo proceso de búsqueda, se lograron obtener resultados más equilibrados y una mayor capacidad de generalización.

En particular, se observó que los siguientes parámetros ofrecieron un mejor desempeño:

- n_estimators: 100
- max_depth: 5
- min_samples_split: 2

Esta nueva configuración permitió que el modelo presentara una disminución en el fenómeno de sobreajuste, como se ilustra en la Figura 22 Aunque la precisión del modelo experimentó una

ligera reducción con una precisión de 89% en comparación con la precisión del 91% que tenía el modelo antes de ajustar los parámetros, este ajuste fue esencial para obtener un equilibrio entre el rendimiento en los datos de entrenamiento y la capacidad de generalización en los datos de validación.

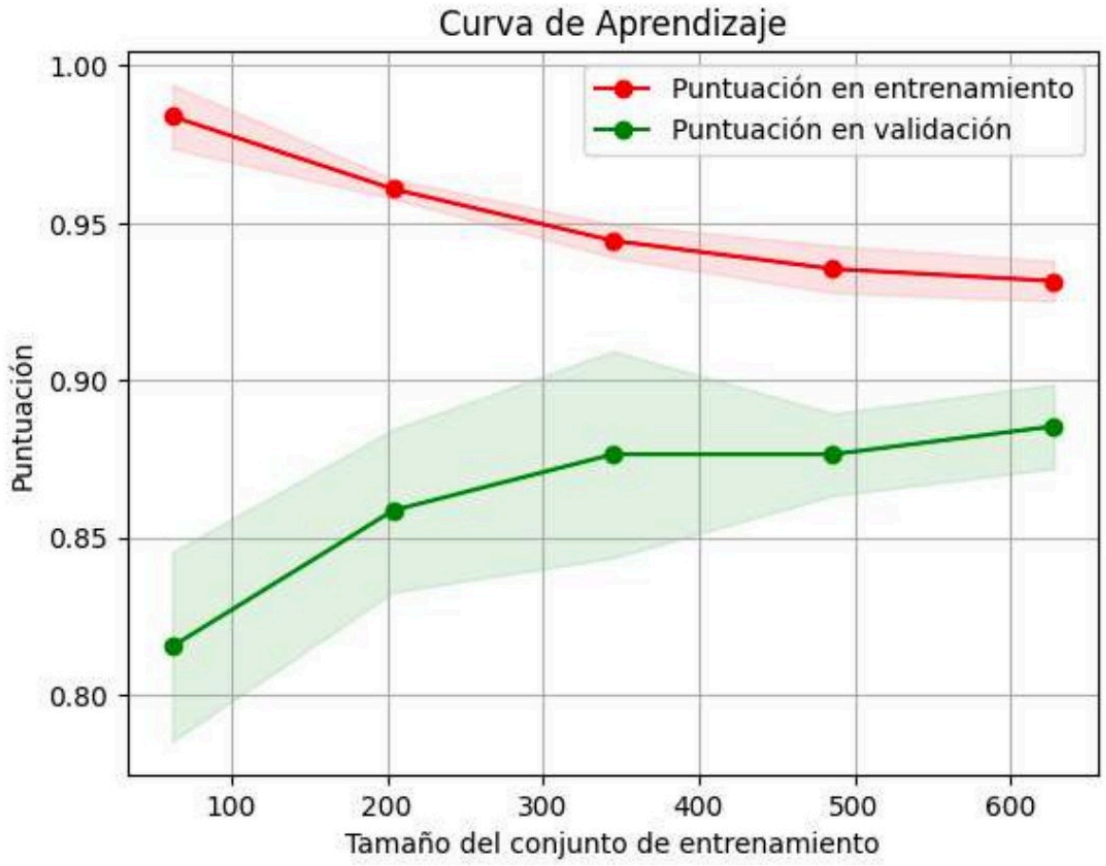


Figura 22. Curva de aprendizaje modelo con ventana de 100 muestras y desplazamiento de 10 con sobreajuste mejorado.
Fuente. Elaboración propia.

La Figura 22 representa la curva de aprendizaje del modelo, donde se puede apreciar que la brecha entre la puntuación de entrenamiento y la puntuación de validación se ha reducido considerablemente, indicando una mejora en la capacidad del modelo para generalizar patrones en nuevos datos. Este resultado refleja una mayor robustez del modelo en lugar de un sobreajuste a los datos de entrenamiento. Para resumir el modelo con mayor precisión obtenido al final del ejercicio, se resume en la Tabla 26. Con mejor precisión que los modelos obtenidos con

anterioridad al realizar el análisis en el dominio del tiempo. Modelo que fue mejor que los obtenidos en el dominio del tiempo.

Tabla 26. Modelo con mejor precisión.

Modelo	Random Forest
Variables	['Ax_Total_Power', 'Ay_Total_Power', 'Gy_Total_Power']
Parámetros	n_estimators: 100
	max_depth: 5
	min_samples_split: 2
Tamaño de la ventana	100 muestras
Desplazamiento	10 muestras

Fuente. Elaboración propia.

3.8 Propuesta de Implementación en C

En esta sección se aborda el desafío de implementar el modelo con mejor precisión obtenido en un Arduino Nano 33 BLE Sense, dispositivo que cuenta con sensor IMU y procesamiento, de modo que permitirá tener predicciones del terreno al tiempo que el rovet está en recorrido, el código se muestra en el Anexo 1.

En esta sección se cubre los principales aspectos del código que son:

- Adquisición de datos: Cómo capturar y registrar mediciones en tiempo real de sensores como acelerómetros y giroscopios a través del Arduino Nano 33 BLE Sense.
- Procesamiento de datos: Cómo analizar y preprocesar los datos recolectados para extraer las métricas relevantes que alimentan el modelo, como la potencia de señales en distintas direcciones.
- Conversión del modelo: Etapa de convertir un modelo de Random Forest previamente entrenado en Python a un formato compatible con el Arduino Nano 33 BLE Sense.

3.8.1 Adquisición de datos

En la etapa de adquisición de datos, es de vital importancia mantener una frecuencia de muestreo constante de 100 Hz. Esto se logra mediante el empleo de interrupciones de dispositivo y la configuración de un temporizador específico. La frecuencia de muestreo constante es esencial para garantizar mediciones precisas y coherentes de sensores, como los acelerómetros y giroscopios presentes en el Arduino Nano 33 BLE Sense.

El uso de interrupciones permite al dispositivo "parar" sus tareas regulares a intervalos regulares y priorizar la adquisición de datos en tiempo real. Además, el ajuste del temporizador garantiza que las mediciones se realicen a la frecuencia requerida, lo que es crucial para el posterior procesamiento y análisis de los datos.

Cuando ocurre una interrupción del sistema, se activa la lectura de la IMU, y se extraen específicamente las variables de acelerómetro en el eje X y Y, junto con la medición de giroscopio en el eje Y. Esta selección se realiza con el fin de optimizar el análisis y el uso de recursos, descartando las señales innecesarias para el propósito de este proyecto.

3.8.2 Procesamiento de los datos

Arduino cuenta con una librería incorporada que permite la realización eficiente de la FFT. La función de FFT se emplea para transformar las mediciones de acelerómetro en los ejes X y Y, así como las mediciones de giroscopio en el eje Y, de su dominio de tiempo original al dominio de frecuencia.

Además, para el cálculo de la potencia en las señales transformadas, se ha diseñado una función específica denominada “calcularPotenciaTotal”. Esta función se adapta de manera precisa a las necesidades de análisis de este proyecto y permite la determinación de la potencia de las señales, un aspecto fundamental para el modelo de Random Forest.

3.8.3 Conversión del modelo

La conversión del modelo de Random Forest previamente entrenado es un paso crítico en la implementación en el Arduino Nano 33 BLE Sense. Para facilitar este proceso, se emplea la biblioteca Micromlgen, una herramienta que simplifica la conversión de modelos de aprendizaje automático a un formato de fácil implementación en dispositivos con recursos limitados, como el Arduino. (Eloquent Arduino, 2023)

El fragmento de código necesario para realizar esta conversión se muestra en la Figura 23. Una vez completada la conversión, se genera un archivo con extensión “.h”. Este archivo contiene el modelo de Random Forest en un formato que el Arduino puede entender y cargar sin problemas.

```
from micromlgen import port
classmap = {
        0: 'arena',
        1: 'asfalto',
        2: 'ladrillo',
        3: 'pasto',
        4: 'piedra'
    }

c_code = port(final_model, classmap)

with open('classifier.h', 'w') as file:
    file.write(c_code)
```

Figura 23. Conversión del modelo de python a C. Fuente. Elaboración propia.

Fuente. Elaboración propia.

4 Conclusiones y recomendaciones

- De las 5 caracterizaciones que se trabajaron (Medias, Varianzas, Simetría, Curtosis y Entropía) las que arrojaron mejores resultados fueron los DataFrames de Medias y Varianzas, cuyos *accuracy scores* al entrenar y predecir, estuvieron por encima del 65%, mientras que los tres restantes, no lograron superar el umbral del 50%.
- Los mejores resultados se dieron trabajando con la caracterización de Varianzas, y se observaron las mejores predicciones en agrupaciones de 3 y 4 variables, por lo que se puede elegir trabajar con cualquiera de estas dos, ya que la diferencia de los resultados entre ambas se mantiene en un rango de 1 a 3 %, en su *accuracy score.*
- Los mejores resultados, con análisis en el dominio del tiempo, incluso en las caracterizaciones que no lograron superar el umbral del 50%, se dieron con una ventana de 20.
- En el dominio del tiempo el mejor modelo con caracterización de Varianza fue Random Forest, en una combinación de 3 variables (Aceleración en x, Giroscopio en Y, Giroscopio en Z) (72.34%), que al buscar y modificar sus parámetros a n_estimators: 50 y max_depth:10, logró subir al 75%.
- Los modelos obtenidos a través de la ingeniería de características en el dominio de la frecuencia han demostrado ser significativamente más efectivos que aquellos derivados de la ingeniería de características en el dominio del tiempo. Este hallazgo subraya la importancia de considerar y explorar en profundidad las características de la señal en el dominio de la frecuencia al abordar problemas de clasificación, particularmente en el contexto de clasificación de terrenos para un Rover utilizando sensores IMU. La aplicación de transformadas de Fourier y técnicas de procesamiento de señales en el dominio de la frecuencia permitió obtener modelos más precisos y confiables, lo que respalda la eficacia de esta metodología para la resolución de problemas similares en el futuro.
- La optimización de parámetros desempeñó un papel fundamental en la mitigación del sobreajuste, lo que resultó en un modelo de Random Forest más equilibrado y con una capacidad de generalización mejorada, aunque con una ligera reducción en la precisión.
- El modelo ganador se ha identificado como un modelo Random Forest que utiliza un conjunto de características compuesto por 'Ax_Total_Power,' 'Ay_Total_Power' y 'Gy_Total_Power.' Obtenidas del transformar las señales al dominio de la frecuencia usando la transformada rápida de Fourier. Para mejorar la capacidad de generalización y reducir el sobreajuste, se llevaron a cabo ajustes en los parámetros del modelo. Los valores óptimos se encontraron en n_estimators: 100, max_depth: 5 y min_samples_split: 2. Este modelo se evaluó utilizando una ventana de 100 muestras y un desplazamiento de 10 muestras. Aunque la precisión se redujo ligeramente al 89%, se observó una significativa reducción en el sobreajuste.

5 Bibliografía

Eloquent Arduino. (01 de 07 de 2023). *Micromlgen*. Obtenido de https://eloquentarduino.com/libraries/micromlgen/

Aponte Vivas, S. (2018). *Diseño e implementación de un sistema basado en aprendizaje automático que facilite la percepción robótica del entorno por medio de sensores laser.* Santiago de Cali.

Bishop, C. (2006). *Pattern Recognition and Machine Learning.*

Breiman, L. (2001). *Random Forests.*

Bulmer, M. (2012). *Principles of Statistics.*

Chang, J., Mingliang, M., Yuling, L., Zerui, L., Xiaochuan, L., & Wenjun, L. (2019). Comparative Study of Different Methods in Vibration-Based Terrain Classification for Wheeled Robot with Shock Absorbers.

Christian, M., Uyanik, C., Erdemir, E., Kaplanoglu, E., Bhattacharya, S., Bailey, R., . . . Hargrove, S. (2019). *Application of Deep Learning to IMU sensor motion.* USA.

Cortes, C., & Vapnik, V. (1995). *Support-Vector Networks.*

Downey, A. B. (2016). *Think DSP.* O'Reilly Media, Inc.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The kdd Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*.

Foil, G. y. (2013). 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. En *Probabilistic surface classification for rover instrument targeting* (págs. 775-782).

Géron, A. (2022). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd Edition.* O'Reilly Media, Inc.

Ghosh, B., Xi, N., & Tarn, T. (1999). *Control in Robotics and Automation Sensor-Based Integration.* Academic Press.

Hao, Z., & Guo, G. (2019). Survey of Machine Learning Random Forest Algorithms.

Hinestroza Ramirez, D. (2018). *El Machine Learning a través de los tiempos y los aporte a la humanidad.* Pereira.

Huang, J., Yu Huang, Z., & Chen, K. (2017). *Combining Low-Cost Inertial Measurement Unit (IMU) and Deep Learning Algorithm for Prediction Vehicle Attitude.* Taiwan.

Luceno, A., & González, F. J. (2006). *Métodos Estadísticos para medir, describir y controlar la variabilidad.*

Mackay, D. J. (2003). *Information Theory, Inference and Learning Algorithms.* Cambridge University Press.

Moine, J. M., Haedo, A. S., & Gordillo, S. (s.f.). *Estudio comparativo de metodologías para minería de datos.* Buenos Aires.

Murphy, K. P. (2012). *Machine Learning A Probabilistic Perspective.* London, England: The MIT Press.

Ojeda, L., Borenstein, J., Witus, G., & Karisen, R. (2006). Terrain characterization and classification with a mobile robot. Journal of Field Robotics.

Pérez, R. (1986). *Nociones Básicas de Estadística.* Universidad de Oviedo.

Robert W. Heath, J. (2017). *Introduction to Wireless Digital Communication a Signal Processing Perspective.* person.

Ross, S. (2014). *A First Course in Probability.* Pearson Education.

Scikit-learn. (s.f.). Obtenido de Scikit-learn: https://scikit-learn.org/stable/index.html

United Nations. (2022). *World Population Prospects 2022*. New York: United Nations.

Warden, P., & Situnayake, D. (2019). *TinyML*. O'Reilly Media, Inc.

William H. Press, S. A. (2007). Numerical Recipes: The Art of Scientific Computing. En *Numerical Recipes: The Art of Scientific Computing* (págs. 600-631). Cambridge.

Witte, R. S., & Witte, J. S. (2015). *Statistics.* Wiley.

6 Anexos

6.1 Código en lenguaje C para implementar el modelo

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <Arduino_LSM9DS1.h>
#include <arduinoFFT.h>
#include "classifier.h"

const int frecuenciaMuestreoHz = 100; // Frecuencia de muestreo en Hz
const int ventanaSize = 100; // Tamaño de la ventana para la transformada
const int desplazamiento = 10; // Desplazamiento entre ventanas

float x, y, z;
float gyroY;
int ventanaIndex = 0;
float ventanaBufferX[ventanaSize]; // Búfer para la señal del acelerómetro en X
float ventanaBufferY[ventanaSize]; // Búfer para la señal del acelerómetro en Y
float ventanaBufferGyroY[ventanaSize]; // Búfer para la señal del giroscopio en Y
arduinoFFT FFT = arduinoFFT();
Eloquent::ML::Port::RandomForest clf;

void setup() {
Serial.begin(9600);
while (!Serial);
Serial.println("Started");

if (!IMU.begin()) {
Serial.println("Failed to initialize IMU!");
while (1);
}

Serial.print("Accelerometer sample rate = ");
Serial.print(IMU.accelerationSampleRate());
Serial.println("Hz");

unsigned int timerValue = 65536 - (F_CPU / (1024 * frecuenciaMuestreoHz)) + 1;
TCCR1A = 0;
TCCR1B = (1 << WGM12) | (1 << CS10) | (1 << CS12);
OCR1A = timerValue;
TIMSK1 = (1 << OCIE1A);
sei();
```

```
FFT.begin(ventanaSize);
}

ISR(TIMER1_COMPA_vect) {
if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
IMU.readAcceleration(x, y, z);
IMU.readGyroscope(x, gyroY, z);



ventanaBufferX[ventanaIndex] = x;
ventanaBufferY[ventanaIndex] = y;
ventanaBufferGyroY[ventanaIndex] = gyroY;


ventanaIndex++;
if (ventanaIndex >= ventanaSize) {
ventanaIndex = 0;
}


if (ventanaIndex % desplazamiento == 0) {
float potenciaX = calcularPotenciaTotal(ventanaBufferX, ventanaSize);
float potenciaY = calcularPotenciaTotal(ventanaBufferY, ventanaSize);
float potenciaGyroY = calcularPotenciaTotal(ventanaBufferGyroY, ventanaSize);


float fftMetrics[3] = {potenciaX, potenciaY, potenciaGyroY};
int predictedLabel = clf.predictLabel(fftMetrics);


// Procesa la etiqueta predicha según tus necesidades
Serial.print("Etiqueta predicha: ");
Serial.println(predictedLabel);
}
}
}


// Función para calcular la potencia total de una señal en una ventana FFT
float calcularPotenciaTotal(float ventanaBuffer[], int ventanaSize) {
float potenciaTotal = 0.0;
for (int i = 0; i < ventanaSize / 2; i++) {
float valorAbsoluto = sqrt(sq(ventanaBuffer[i]));
```

```
potenciaTotal += sq(valorAbsoluto);
}
return potenciaTotal;
}


void loop() {
}
```