Universidad Zaragoza

Trabajo Fin de Grado

# Sistema de teclado completamente virtual y de alta movilidad alternativo al `QWERTY` basado en visión artificial.


Autor

Ángela Rojo Sediles

Director

Albert Redó Sánchez

ESCUELA DE INGENIERÍA Y ARQUITECTURA


2023

# AGRADECIMIENTOS

Agradezco a Marc Lara la oportunidad de desarrollar su idea original de la lógica del teclado TYPYN y sus comentarios y soporte durante el desarrollo del proyecto. Agradezco el soporte del grupo del Graphics & Imaging Lab, liderado por el profesor Diego Gutiérrez, y en particular a mi director, Albert Redó Sánchez, por el guía y discusiones acaecidas durante el desarrollo del proyecto. Finalmente quiero agradecer el soporte incondicional de mi familia y amigos durante este enriquecedor camino.

# Sistema de teclado completamente virtual y de alta movilidad alternativo al QWERTY basado en visión artificial.

El diseño de teclado físico más extendido es la distribución conocida como QWERTY. El principal problema que presenta el diseño QWERTY es que fue diseñado con el objetivo de reducir la velocidad de pulsación en máquinas de escribir mecánicas para evitar que los tipos interfirieran y se atascasen entre sí y, en consecuencia, no es una distribución pensada para facilitar la rapidez de pulsaciones. La proliferación de dispositivos móviles ha impuesto el uso de **teclados virtuales** en sustitución de los teclados físicos. Los teclados virtuales permiten la interacción usando los dedos, un ratón o un joystick y se encuentran en las pantallas de teléfonos, tablets, televisiones e incluso en algunas páginas web o disponibles como una proyección láser sobre una mesa o superficie plana. Los teclados virtuales proporcionan al usuario mayor **libertad**, **adaptabilidad** y **poder de decisión** según **sus preferencias y las condiciones de trabajo**. De todos modos, la distribución de teclas preferente para la mayoría de los usuarios de los teclados virtuales sigue siendo la distribución QWERTY. Existe cierto consenso en la comunidad científica de que la distribución QWERTY no es la más óptima para la interacción en escenarios virtuales en aplicaciones de realidad virtual y realidad aumentada y en escenarios en los cuales el usuario tiene que moverse para realizar ciertas tareas. La distribución QWERTY limita al usuario a permanecer en posiciones relativamente estáticas y fijas. La consecuencia de que el usuario tenga que estar relativamente estático tiene como consecuencia negativa la proliferación de hábitos sedentarios. El proyecto **TYPYN** nace con la motivación y el objetivo de proporcionar al usuario una forma extremadamente intuitiva y fácil de aprender para interactuar con dispositivos virtuales en cualquier lugar y realizando cualquier actividad física. El diseño de teclado propuesto en el proyecto **TYPYN** consiste en la entrada de datos mediante gestos simples que permiten al usuario introducir las letras del alfabeto. TYPYN implementa tecnologías de visión por computador y sensórica *wearable* para registrar de forma no intrusiva los gestos de la mano y dedos del usuario y transformarlos en las letras del alfabeto mediante una **lógica intuitiva y fácil de aprender** para el usuario. El objetivo principal del proyecto ha sido diseñar, implementar y validar un prototipo que permita demostrar que el dispositivo planteado para este proyecto, así como su lógica de teclado, es viable como dispositivo de entrada de datos.

# Índice

# Capítulo 1

# Introducción y objetivos

## 1.1. Motivación

Los teclados físicos son los dispositivos más usados habitualmente para la interacción con computadoras y otros dispositivos [1]. Sin embargo, recientemente, los teclados virtuales presentes en **smartphones** y **tablets** se han popularizado debido a que permiten mayor versatilidad y libertad a los **usuarios** [2, 3].

Dentro de los **teclados físicos**, el diseño de teclas más extendido es la distribución conocida como `QWERTY`. La distribución `QWERTY` fue diseñada y patentada por Christopher Sholes en 1868 [4], quién vendió la patente a Remington en 1873. Esta distribución de teclado no era la única existente para las máquinas de escribir de la época, sin embargo `QWERTY` fue el diseño que ganó mayor popularidad y pasó a conocerse como el *"diseño universal"* en la década de 1880 [5].

El principal problema que presenta el diseño `QWERTY` en la actualidad es que fue diseñado con el objetivo de reducir la velocidad de pulsación en máquinas de escribir mecánicas para evitar que los tipos interfirieran y se atascasen entre sí. El diseño `QWERTY` no fue diseñado para teclados no mecánicos como los actuales [6, 7, 8, 9]. Por ello, a lo largo de las últimas décadas, han ido apareciendo una serie de disposiciones de teclado distintas que pretendían adaptarse mejor a los nuevos dispositivos del momento. Una de esas disposiciones de teclado fue el teclado **Dvorak**, inventado en 1936 por August Dvorak [10]. El objetivo de este diseño de teclado fue precisamente mejorar la velocidad de escritura, minimizar los errores, y reducir al mínimo el movimiento de los dedos. La adopción del teclado Dvorak ha sido escasa debido a la **elevada barrera de entrada que supone su aprendizaje una vez el usuario ya ha aprendido** `QWERTY`.

Por otro lado, debido a la proliferación de dispositivos móviles [11, 12], el uso de **teclados virtuales** se ha extendido enormemente en los últimos años en sustitución de los teclados físicos [13, 14]. Este tipo de teclados permiten la interacción usando los dedos, un ratón o un joystick y se encuentran en las pantallas de teléfonos [15], ta-

blets, televisiones e incluso en algunas páginas web o disponibles como una proyección láser sobre una mesa o superficie plana. Algunas ventajas destacables de los teclados virtuales frente a los teclados físicos son: i) la disposición de las teclas puede ser personalizable al gusto, a necesidades del usuario o a otros modelos existentes, como el modelo Dvorak mencionado anteriormente; ii) ocupan menos espacio y son más portables que los teclados físicos, lo que facilita la tarea de escribir fuera de la oficina o del hogar; y iii) que al estar integrados en las pantallas de otros dispositivos, abaratan costes, ya que se necesita un dispositivo menos.

En resumen, las características de los teclados virtuales proporcionan al usuario mayor **libertad**, **adaptabilidad** y **poder de decisión** según **sus preferencias y las condiciones de trabajo**. De todos modos, la distribución de teclas preferente para la mayoría de los usuarios de los teclados virtuales sigue siendo la distribución `QWERTY` [16].

Más recientemente, la **realidad virtual** [17] y la realidad aumentada están adquiriendo mucho protagonismo y existe cierto consenso en la comunidad científica de que la distribución `QWERTY` no es la más óptima para la interacción en escenarios virtuales debido a que limita al usuario a permanecer en posiciones relativamente estáticas y fijas. La integración y diseminación de estas tecnologías hace **plantear una nueva forma de interacción** entre los **dispositivos de entrada** y el **usuario**, el cuál requiere una mejor experiencia y facilidad de uso. Esta nueva forma de interacción es especialmente importante en escenarios en los que el usuario se encuentra en movimiento y fuera de puestos de trabajo fijos, en lo cuales el uso de un teclado físico o virtual con distribución `QWERTY` entorpece enormemente la capacidad del usuario para la introducción de datos.

La consecuencia de que el usuario tenga que estar relativamente estático y en puesto fijo de trabajo para el uso de dispositivo virtuales tiene como consecuencia negativa la proliferación de hábitos sedentarios. La Organización Mundial de la Salud estima que el sedentarismo y falta de actividad física debido a permanecer sentados o estáticos en la oficina es la causa de la epidemia de obesidad, estrés, y afecciones musculoesqueléticas. Esta demostrado que la inactividad causa 5 millones de muertes por año y que la actividad física mejora la salud significativamente. [18]

El proyecto **TYPYN** nace con la motivación y el objetivo de proporcionar al usuario una forma de interactuar con dispositivos virtuales en cualquier lugar y realizando cualquier actividad física (e.g., caminar, correr, pasear, etc.), y a la vez que sea un forma de interactuar intuitiva y fácil de aprender. Este proyecto se basa en la patente [19]. La adopción del teclado propuesto en TYPYN no sólo proporciona una forma **más rápida, intuitiva y eficiente de interactuar con dispositivos virtuales y**

**móviles**, sino que permite al usuario **mejorar su salud y bienestar** al permitirle trabajar mientras realiza cualquier actividad física de su gusto.

## 1.2. Proyecto TYPYN

El diseño de teclado propuesto en el proyecto **TYPYN** consiste en la entrada de datos mediante gestos simples que permiten al usuario introducir las letras del alfabeto. TYPYN implementa tecnologías de visión por computador para registrar de forma no intrusiva e interpretar los gestos de la mano y dedos del usuario y transformarlos en las letras del alfabeto mediante una **lógica intuitiva y fácil de aprender** para el usuario.

### 1.2.1. Clasificación de los gestos y disposición del teclado

Los gestos con las manos son una forma de comunicación no verbal. Se puede decir que un gesto con la mano es una composición formada por el centro de la palma, la posición de los dedos y la forma construida por la mano. Este tipo de comunicación presenta muchas ventajas. La principal es que facilita la comunicación, debido a que es un medio natural [20] y puede ser usado en diferentes campos como, por ejemplo, lenguaje para personas sordomudas, control de robots, interacción humano-computador (ordenadores y tablets, juegos, entornos virtuales), domótica, aplicaciones médicas, etc. Los gestos se pueden **clasificar** de diferentes formas: i) postura y gesto, siendo la *postura* lo que se refiere a la forma de la mano, mientras que el *gesto* se refiere al movimiento de la mano; ii) estático y dinámico, siendo el *estático* lo que se refiere a la pose estable de la mano, mientras que *dinámico* se refiere al movimiento de la mano, por ejemplo, saludar, e iii) híbrido, que combina los anteriores criterios.

La **disposición y lógica del teclado** propuesta por TYPYN es la siguiente:

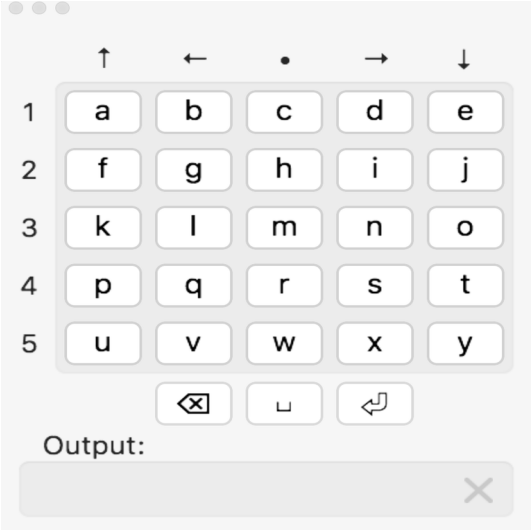



Figura 1.1: Lógica del teclado TYPYN. Las filas indican el número de dedos involucrados en el movimiento mientras que las columnas indican el movimiento: arriba, izquierda, centro, derecha y abajo. Gestos especiales de selección, borrado y avance pueden implementarse con movimiento de pinzamiento o giros completos de la mano para arriba o para abajo.

En este teclado cada letra del alfabeto lleva asociada una postura de la mano y un gesto. La *postura de la mano* consiste en el número de dedos estirados que tenga el usuario al realizar una letra del alfabeto. Siendo posibles las posturas que involucren 1, 2, 3, 4 o 5 dedos. Mientras que, el *gesto* es el movimiento de la mano asociado a la letra. Siendo posibles los movimientos derecha, izquierda, arriba, abajo y centro. Por ejemplo: para realizar la letra 'a': la postura debe ser 1 dedo y el gesto "arriba".

Por otro lado, según la otra clasificación, estático sería un fotograma capturado por la cámara en un momento concreto durante la realización del movimiento y dinámico sería la composición del gesto con todos los fotogramas capturados por la cámara.

Por último, cabe destacar que existe una gran variedad de movimientos de la mano en un gesto, ya que, por ejemplo, un apretón de manos varía en función de la persona, el momento, el lugar, etc. En nuestro proyecto ocurre lo mismo, por ejemplo, el gesto asociado a la letra 'a' es con 1 dedo y el movimiento es hacia arriba. En este caso, este movimiento aunque siempre sea realizado con el mismo dedo y orientación del movimiento, variará en la forma de realizarlo en función de la persona y el momento, por lo que este movimiento no será realizado exactamente de la misma forma, pero debería ser clasificado siempre como el mismo gesto. Además de existir esta dificultad, en nuestro proyecto, este alfabeto solo restringe el número de dedos, por lo que la persona puede realizarlo con cualquier dedo de la mano y debe de ser detectado y reconocido también como su gesto válido.

### 1.2.2. Mejoras frente a otros teclados

Algunas de las **ventajas** anticipadas de TYPUN frente a los teclados actuales, tanto virtuales como físicos, son:

- TYPYN está enfocado a **tecnologías en auge** como, por ejemplo, RV y RA.
- Los gestos son un medio natural de comunicación, lo que facilita el aprendizaje y ejecución por parte del usuario.
- El modelo de gestos es **simple y fácil de aprender** para el usuario, en contraposición con el `QWERTY`, cuyo diseño fue pensando principalmente para teclados mecánicos.
- El dispositivo planteado **no exige al usuario llevar un dispositivo pesado**, molesto o poco común como, por ejemplo, guantes sensorizados usados en RV. Simplemente requiere una cámara que podría estar integrada en un reloj inteligente, lo cuál es menos intrusivo y cómodo para el usuario.
- Permite al usuario **escribir en movimiento o realizando otras tareas físicas o deportivas** como, por ejemplo, andar y correr, ya que no fuerza al usuario a estar atado a un dispositivo fijo. Esto permite al usuario una **mayor libertad**.
- **Mejora de la salud y el bienestar**, ya que permite al usuario llevar un estilo de vida más activo, en oposición a un estilo de vida sedentario que provoca obesidad, problemas de espalda y otros indisposiciones físicas debidas a la falta de actividad física.
- Puede **mejorar y facilitar la experiencia de comunicación en personas con alguna discapacidad** como, por ejemplo, invidentes o personas con poca movilidad.

### 1.2.3. Acogida del producto

Esta claro que la acogida de un nuevo producto siempre requiere de una fase de aprendizaje y adaptación, sobre todo para los más mayores. Sin embargo, debido a la simplicidad de los gestos, la acogida del producto podría ser relativamente sencilla.

En cuanto a los **adolescentes (12-18 años)** y los **jóvenes (18-25 años)**, el aprendizaje y uso del teclado, a priori, podría ser algo bastante fácil, sobre todo en la etapa de la adolescencia. Esto es así ya que es una época clave para el aprendizaje. Además, este teclado hace uso de gestos muy básicos y fáciles de aprender, los cuáles pueden resultar mucho más simples que algunas de las posturas que hay que realizar

con las manos para escribir en un teclado `QWERTY`. Por último, el aprender este teclado puede resultar algo entretenido, ya que además permite y fomenta la actividad entre los jóvenes.

En cuanto a los **adultos (25-65 años)**, dicha adaptación probablemente sea más complicada al principio, debido al uso habitual del teclado `QWERTY` en oficinas y empresas. Sin embargo, para personas que trabajen a distancia o en otros ámbitos del día a día, podrían irse integrando de manera sencilla. Para ello, estos usuarios deben tener una clara ambición o predisposición para ello, motivados por una vida más sana o realizar algún ejercicio. Por último, si los usuarios son padres, pueden tener otras motivaciones como enseñar a sus hijos este nuevo modelo de teclado, el cuál puede ser una mejora para su salud, o si éstos ya lo usan, pueden estar impulsados por ver o entretenerse con sus hijos al usar el dispositivo.

Finalmente, en el rango de la **tercera edad (> 65 años)**, el aprendizaje seguramente sea muy costoso debido a su menor conocimiento de las nuevas tecnologías, por lo que, en caso de que lo requieran, probablemente tendrán una difícil adaptación.

## 1.3. Objetivos

El objetivo principal del proyecto es elaborar un prototipo que permita demostrar que el dispositivo planteado para este proyecto, así como su lógica de teclado, es **útil y viable como dispositivo de entrada de datos**. Los objetivos concretos que se plantean para el proyecto son los siguientes:

- **Identificación del problema a resolver** y división de éste en una serie de tareas o bloques a realizar durante el proyecto, así como la **elaboración de un plan de acción**. *(Diagrama de Gantt: Anexo C)*
- Investigación de la **historia de los teclados** *(Introducción: Capitulo 1, sección 1.1)*, así como de teclados actuales existentes similares al planteado en este proyecto. *(Estado del arte: Capítulo 2, sección* **??***)*
- Investigación sobre la **obtención, análisis y transformación de los datos** en el contexto de un proyecto de **aprendizaje máquina** y en el contexto concreto de nuestro proyecto que es la extracción de **datos de la mano**. Esto incluye la obtención de datos, análisis de los mismos, limpieza, extracción y selección de características y muestras, normalización, etc. con el objetivo de obtener un buen conjunto de datos. *(Estado del arte: Capítulo 2, secciones 2.1, 2.2, 2.3, 2.5, 2.4)*
- Investigación sobre los **algoritmos de aprendizaje** existentes para problemas de **clasificación**, así como **medidas de evaluación** de modelos de aprendizaje

para la posterior comparación entre los modelos desarrollados. *(Estado del arte: Capítulo 2, secciones 2.6, 2.7, 2.8)*

- Planteamiento de un **diseño hardware** que permita desarrollar el prototipo del proyecto. *(Diseño y desarrollo: Capítulo 3, sección 3.2)*
- **Planteamiento del problema a nivel de software**: elaboración de diferentes diagramas (arquitecturales, etc.) con el objetivo de simplificar el problema descomponiéndolo en diferentes módulos. *(Diseño y desarrollo: Capítulo 3, secciones 3.1, 3.6)*
- **Desarrollo de los diferentes módulos** en base a lo investigado, planteado, aprendido y observado durante el avance del proyecto. *(Diseño y desarrollo: Capítulo 3)*
- **Elaboración de pruebas, obtención y análisis de resultados** con el objetivo de extraer conclusiones que nos permitan seleccionar el modelo más adecuado para el proyecto. *(Pruebas y resultados: Capítulo 4)*
- Obtención del **modelo final** del proyecto TYPYN, así como la realización de un análisis de lo desarrollado durante el proyecto, incluyendo las **limitaciones y mejoras futuras** del proyecto. *(Conclusiones y trabajos futuros: Capítulo 5)*
- **Elaboración de una memoria** que represente con fidelidad el trabajo desarrollado durante el proyecto.

# Capítulo 2

# Estado del arte

Para la detección de los gestos de las manos hay que abordar diversas cuestiones ya que existen numerosos métodos de reconocimiento que, dependiendo de las circunstancias, presentan ciertos beneficios y limitaciones respecto a otros. **El rendimiento de estos métodos depende de las técnicas usadas (puntos de similitud y diferencia, técnicas de segmentación de la mano), algoritmos de clasificación, la cantidad y el tipo de gestos a detectar, el conjunto de datos utilizado, el rango de detección, el tipo de cámara, etc.** Principalmente, existen diversos enfoques para el reconocimiento de los gestos de la mano. Los dos enfoques principales son: i) el uso de **guantes portátiles**, que utiliza tecnología de sensores instrumentados [21], o ii) el uso de **cámaras** junto a técnicas de **visión por computador** [22, 23]. Este último enfoque y las distintas técnicas que lo componen se explican en la sección 2.1.

## 2.1. Segmentación de manos

En esta sección se exponen algunas de las principales técnicas de segmentación basadas en visión por computador. Esta sección esta basada en [22].

### 2.1.1. Reconocimiento basado en colores

Dentro del reconocimiento basado en color, existen principalmente dos aproximaciones: el reconocimiento puede realizarse mediante el uso de un guante de colores o técnicas de detección del color de la piel.

Usar un **guante con marcas de diferentes colores** permite que la cámara pueda seguir y detectar la localización de la palma y dedos y obtener el modelo geométrico 3D de la mano. Esto permite operaciones como zoom, mover, dibujar y escribir usando un teclado virtual.

Una de las ventajas principales es la simplicidad de uso, mayor comodidad y bajo

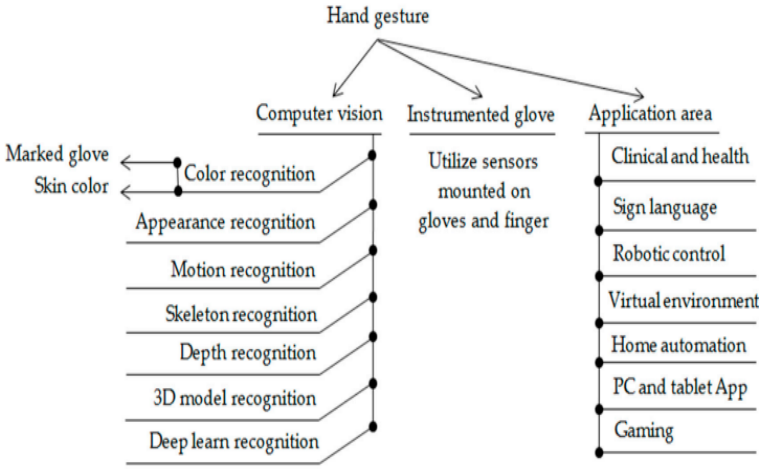



Figura 2.1: Clasificación de los métodos de segmentación de manos. Fuente de la imagen: [22]

precio respecto al guante instrumentado, ya que este guante no hace uso de sensores. Por otro lado, la desventaja principal es que sigue requiriendo el uso de guantes, lo que limita la interacción natural *Human-Computer-Interaction* (HCI).

Por otro lado, dentro de la **detección del color de la piel** se pueden considerar dos técnicas principalmente. La primera consiste en la *detección de la piel basada en píxeles*: cada píxel de la imagen es clasificada en piel o no, individualmente de su vecino. La segunda consiste en la *detección de la piel de la región*: los píxeles de la piel se procesan espacialmente en función de información como intensidad y textura.

Permite su uso en numerosas aplicaciones como clasificación de objetos, recuperación de fotografías degradadas, seguimiento del movimiento de personas, observación de vídeos, aplicaciones HCI, reconocimiento facial, segmentación de manos e identificación de gestos. Para esto, se pueden tener en cuenta diferentes espacios de color en función de la aplicación.

Este método presenta varios desafíos como variaciones de la iluminación, problemas de fondo, el fondo no debe contener ningún elemento que coincida con el color de piel u otros tipos de ruido.

### 2.1.2. Reconocimiento basado en apariencia

Este método consiste en extraer características de la imagen, en este caso, de las manos, con el objetivo de modelar la apariencia visual sin un proceso de segmentación previo y comparar dichas características con las extraídas de las imagen de entrada. El método es ejecutado en tiempo real debido a la facilidad de extracción de características de imágenes 2D y es considerado más fácil de implementar que el método del modelo

3D. Este método permite detectar varios tonos de piel.

El algoritmo `AdaBoost` mantiene características fijas como puntos clave para una parte de una mano. Es utilizado para detección de objetos, caracterización, modelado de movimiento y reconocimiento de patrones.

El algoritmo `AdaBoost` puede resolver el problema de oclusión y permite separar en 2 modelos: un modelo de movimiento y un modelo estático 2D.

### 2.1.3. Reconocimiento basado en movimiento

Este método permite extraer un objeto a través de una serie de fotogramas de imagen. El algoritmo `AdaBoost` es necesario para reconocer el gesto.

El principal problema de este método ocurre cuando más de un gesto esta activo en el proceso de reconocimiento y cuando el fondo es dinámico. La pérdida del gesto puede ser causado por oclusión entre los gestos de la mano rastreados o error en la extracción de la región del gesto rastreado y el efecto de larga distancia en la apariencia de la región.

### 2.1.4. Reconocimiento basado en esqueleto

El reconocimiento basado en esqueleto especifica parámetros del modelo que pueden mejorar la detección de características complejas. Estas características son usadas para la representación del esqueleto que es usado para la clasificación.

Las diversas representaciones del esqueleto describen atributos geométricos y restricciones y traducen características y correlaciones de datos, para centrarse en características geométricas y estadísticas.

Las características utilizadas más comúnmente son: la orientación, ubicación, trayectorias y curvatura de las articulaciones, así como el espacio y grado de ángulo entre las articulaciones.

### 2.1.5. Reconocimiento basado en profundidad

Este enfoque se basa en el uso de una cámara de profundidad. Este tipo de cámara proporciona información geométrica 3D sobre un objeto. Previamente, se utilizan las dos aproximaciones principales: preceptos *time-of-flight* y codificación ligera. Los datos 3D de una cámara de profundidad reflejan directamente el campo de profundidad si se comparan con una imagen en color que contiene solo una proyección. Este enfoque, permite que la iluminación, la sombra y el color no afecte a la imagen resultante. Sin embargo, el costo, el tamaño y la disponibilidad de la cámara de profundidad limitan su uso.

### 2.1.6. Reconocimiento basado en modelos 3D

El modelo 3D depende principalmente del modelo de mano cinemática 3D, el cuál tiene un alto grado de libertad. En este modelo, la estimación de los parámetros de la mano se obtiene comparando la imagen de entrada con la apariencia bidimensional proyectada por el modelo de mano tridimensional. Este modelo presenta las características de la mano como una "estimación de la pose" cuando forma un modelo volumétrico o esquelético que es idéntico al de la mano del usuario. Donde el parámetro del modelo 3D se actualiza a través del proceso de coincidencia y el de profundidad se agrega al modelo para aumentar la precisión. Hay algunas limitaciones: requiere un gran conjunto de datos de imágenes para formular las formas características de la mano en caso de vista múltiple, el proceso implica cálculos costosos y tiene menor capacidad para tratar vistas poco claras.

### 2.1.7. Reconocimiento basado en *Deep Learning*

Es una técnica buena y confiable que es utilizada en una gran variedad de aplicaciones. El *Deep Learning* usa multicapas para aprender datos y proporciona buenos resultados de predicción. El mayor desafío que enfrenta esta técnica es el conjunto de datos requerido para aprender el algoritmo, lo que puede afectar al tiempo de procesamiento [24].

## 2.2. Modelos para la descripción de manos

Esta sección se centra principalmente en un paquete llamado `Mediapipe Hands`, el cuál aporta un modelo de descripción de manos, el cuál es interesante para el desarrollo proyecto.

`Mediapipe` [25, 26] es una solución que permite detectar **21 puntos de referencia** de las manos en una imagen o vídeo. Para ello, hace uso del módulo *Hand Landmarker* que usa un modelo de aprendizaje automático (*Machine Learning* - ML) que opera sobre los datos de imagen.

El modelo de aprendizaje que implementa se entrenó en aproximadamente 30000 imágenes del mundo real, así como en varios modelos de manos sintéticas renderizados impuestos sobre varios fondos. Este modelo consta de **dos modelos empaquetados**:

- *Modelo de detección de palma*: ubica las manos dentro de la imagen de entrada, es computacionalmente costoso.
- *Modelo de detección de puntos de referencia de mano*: identifica puntos de referencia específicos de la mano en la imagen recortada de la mano definida por el

modelo de detección de la palma.

El módulo consta de **dos modos de ejecución**:

– *Imagen*: opera sobre los datos de forma estática. Hace uso de los dos modelos explicados.

– *Vídeo o transmisión en vivo*: opera sobre los datos como un flujo continuo. En este caso, *Hand Landmarker* usa el cuadro delimitador definido por el *modelo de puntos de referencia de manos* en un marco para localizar la región de las manos para usarla en los marcos posteriores. *Hand Landmarker* solo vuelve a activar el *modelo de detección de palmas* si el *modelo de puntos de referencia de manos* ya no identifica la presencia de manos o no puede rastrear las manos dentro del marco. Esto reduce la cantidad de veces que *Hand Landmarker* activa el *modelo de detección de palma*, el cuál es más costoso.

El módulo representa los puntos de referencia de las manos detectadas de distintas formas: i) coordenadas de imagen, o ii) coordenadas mundiales. El **modelo de manos** que sigue `Mediapipe` es el siguiente:

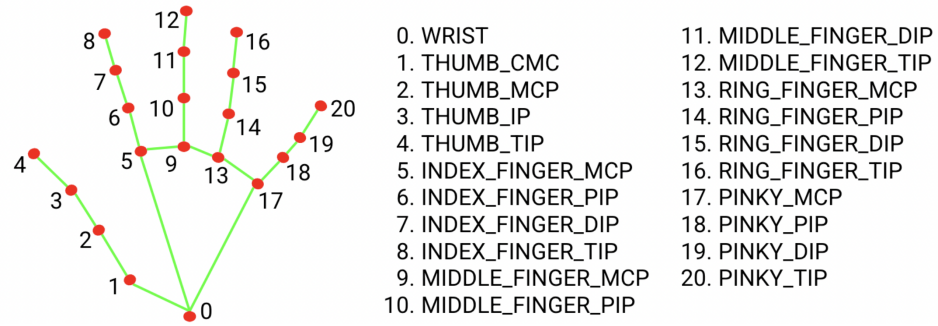



Figura 2.2: Modelo de descripción de manos de Mediapipe, que se basa en 21 puntos de referencia, 4 por cada dedo y 1 para la muñeca. [26]

Este modelo consta de 21 coordenadas, 4 coordenadas por dedo y una referente a la muñeca. Las coordenadas de cada dedo se corresponden a la yema, las flexiones y la base del dedo.

## 2.3. Pre-procesamiento de datos

El **pre-procesamiento de datos** es una etapa muy importante cuyo objetivo es obtener **conjuntos de datos finales** que puedan considerarse **correctos** en base a la tarea que se quiera realizar. La evaluación del pre-procesamiento se basa en gran parte en análisis estadístico. En el Anexo A se describen algunos los conceptos estadísticos usados en este trabajo. Esta etapa incluye los siguientes aspectos:

– Imputación de valores faltantes.

– Filtrado de ruido.

– Discretización de valores.

– Tratamiento de datos para el preprocesamiento desequilibrado.

– Reducción de instancias o reducción horizontal mediante técnicas de selección y generación.

– Reducción de la dimensionalidad o reducción vertical mediante técnicas de selección de características o transformaciones en el espacio.

La etapa de preprocesado de datos es importante ya que los resultados finales del modelo dependerán en gran medida de los datos usados. Por ejemplo, una combinación de tamaño de muestra pequeño y alta dimensionalidad puede dificultar el aprendizaje debido a la dificultad que implica formar conjunciones sobre un alto grado de características teniendo un número de muestras limitadas. Si el espacio de muestra es suficientemente grande, se puede definir un conjunto de reglas inductivas generales, aunque complejas, para el espacio de datos. Sin embargo, cuando las muestras son limitadas, las reglas formadas pueden volverse demasiado específicas, lo que lleva a un sobre-ajuste [24]. En esta sección se describirán algunos de estos aspectos, los cuáles pueden ser interesantes para el proyecto.

### 2.3.1. Tratamiento de conjuntos de datos desequilibrados

Un **conjunto de datos desbalanceado o desequilibrado** es un conjunto donde el número de observaciones pertenecientes a una clase es significativamente mayor que las pertenecientes a las otras clases [27, 28, 29, 30].

Este tipo de conjunto de datos compromete significativamente el rendimiento de la mayoría de algoritmos de aprendizaje estándar, ya que la mayoría de ellos espera distribuciones de clase equilibradas o costos de clasificación errónea iguales. La mayoría de algoritmos genera reglas de inducción que describen a las clases. Al tener un desequilibrio de clases, las reglas de la clase minoritaria suelen ser menores en cantidad y más débiles que las de la clase mayoritaria, ya que es superada en número de muestras y sobre-presentada [24].

Como resultado, este tipo de conjunto de datos provoca un problema de métricas en la medición del rendimiento de los algoritmos. Al tener un conjunto de datos desequilibrado, un algoritmo puede predecir todos los casos como pertenecientes a la clase

mayoritaria. La precisión del algoritmo será muy alta si este desequilibrio es muy grande, pero realmente no será así. Es una precisión "engañosa" ya que la clase minoritaria es totalmente ignorada y esto resulta en problemas de clasificación. Como solución a este problema existen dos enfoques:

- Cambiar la forma de evaluar el modelo: usar métricas pensadas para este tipo de datos, en vez de usar métricas como la precisión que no reflejan fielmente la calidad del modelo ante este tipo de conjuntos de datos.
- Balancear las clases: modificar el conjunto de datos original con el objetivo de generar conjuntos de datos que tengan un equilibrio entre las distintas clases que lo componen. Para ello, se usan técnicas de sobre-muestreo y sub-muestreo. Estas técnicas y conceptos son explicados en detalle en posteriores secciones de este capítulo. *(Procesamiento de datos desequilibrados: sección 2.4)*

### 2.3.2. Reducción de la dimensionalidad

Algunos motivos por los que realizar una selección de características o variables antes de realizar un entrenamiento son:

- **Reduce los tiempos de entrenamiento** al tener menos datos.
- **Requiere menos almacenamiento**, al poder eliminar características de los datos en la base de datos.
- **Reduce la complejidad del modelo**, se obtienen modelos más simples y fáciles de entender.
- **Reduce el "overfitting"**: el clasificador generaliza mejor al no tener que ajustarse a un modelo complejo de datos.
- **Aumenta la capacidad de generalización del modelo** al no tener tantas características a las que ajustarse, obteniendo modelos más precisos.

Dentro de la selección de características, existen diversos métodos para ello. Los métodos pueden clasificarse de dos formas:

- Según la **forma en la que seleccionan** las características pueden clasificarse en *métodos de filtro, métodos envolventes y híbridos.*
- Según **cómo analizan las variables** pueden clasificarse en *métodos de análisis uni-variable o métodos de análisis multivariable.*

Estos métodos se explican de forma detallada en posteriores secciones. *(Selección de características, sección 2.5).*

## 2.4. Procesamiento de datos desequilibrados

En esta sección se explican diferentes métodos que utilizan el método denominado "muestreo" para solucionar los problemas que presentan estos tipos de conjuntos de datos. Dichos problemas fueron explicados en una sub-sección de la sección 2.3. Esta sección basa su contenido principalmente en el paper [30].

El "muestreo" consiste en modificar el conjunto de datos original con el objetivo de generar conjuntos de datos que tengan un equilibrio entre las distintas clases que lo componen. La idea principal del **muestreo** es aumentar las muestras de la clase minoritaria o disminuir las muestras de la clase mayoritaria con el objetivo de obtener un equilibrio. Existen distintos tipos de muestreo:

- **Sub-muestreo (*undersampling*)**: se centra en la eliminación de instancias de la clase mayoritaria con el objetivo de equilibrar las clases.
- **Sobre-muestreo (*oversampling*)**: se centra en la agregación de instancias de la clase minoritaria con el objetivo de equilibrar las clases.
- **Método híbrido**: combina ambas dos técnicas.

Las técnicas más básicas de muestreo son RUS y ROS para sub-muestreo y sobre-muestreo, respectivamente. Existen otras técnicas más avanzadas de muestreo que han intentado abordar los inconvenientes de las técnicas más básicas como *NearMiss* para sub-muestreo y SMOTE y ADASYN para sobre-muestreo. Todas estas técnicas son explicadas a continuación. Para describir los métodos se va a usar la siguiente notación:

- $S = (X, Y) = \{(x_i, y_i)\}, i = 1, .., m$: conjunto de datos de entrenamiento de $m$ muestras, donde:
  - $x_i = \{f_1, .., f_n\}, x_i \in X$: es una muestra en el espacio de características n-dimensional.
  - $y_i = \{1, .., C\}, y_i \in Y$: es la etiqueta de clase asociada a $x_i$, el valor objetivo.
- $S = S_{min} \cup S_{maj}$ siendo $S_{min} \cap S_{maj} = \emptyset$, donde:
  - $S_{min} \subseteq S$: subconjunto de datos de la clase minoritaria.
  - $S_{maj} \subseteq S$: subconjunto de datos de la clase mayoritaria.

Las técnicas básicas basan sus métodos de balanceo en la aleatoriedad. Por ello, son técnicas mucho más rápidas que otras más avanzadas, aunque también pueden ser menos eficientes.

### 2.4.1. Sub-muestreo aleatorio - *random undersampling*

El método *random undersampling* (RUS) obtiene la proporción deseada entre clases mediante la **eliminación aleatoria de instancias de la clase mayoritaria**. Como resultado, obtiene un conjunto de datos correspondiente al original menos el subconjunto de datos de la clase mayoritaria elegido de forma aleatoria. Esto es: $|S_{res}| = |S_{min}| + |S_{maj}| - |E_{maj}|$

Las principales ventajas de este método son **tiempos de computación y uso de memoria menores**, debido a la reducción de la cantidad de datos para entrenamiento. La principal desventaja es el criterio de reducción de muestras, ya que al ser aleatorio **no se controla que muestras elimina** pudiendo descartar información útil necesaria para el clasificador. Al eliminar muestras relevantes, puede generar una mala representación de la población en los datos lo que conduce a un menor rendimiento en los modelos. Por ello, este método puede ser **apropiado para conjuntos de datos poco desequilibrados**, pero no para conjuntos altamente desequilibrados y con bajo número de instancias.

### 2.4.2. Sobre-muestreo aleatorio - *random oversampling*

El método *random oversampling* (ROS) obtiene la proporción deseada de clases mediante la **replicación aleatoria de instancias de la clase minoritaria**. Como resultado, obtiene un conjunto de datos correspondiente al original más un subconjunto de datos replicado correspondiente a la clase minoritaria. Esto es: $|S_{res}| = |S_{min}| + |S_{maj}| + |E_{min}|$

En caso de tener un gran desequilibrio en los datos se utiliza la **técnica de "reemplazo"** que permite seleccionar una muestra minoritaria más de una vez con el objetivo de alcanzar el equilibrio entre clases.

La principal ventaja de este método es que **no hay pérdida de información** ya que conserva el conjunto de datos original. Las principal desventaja es el **aumento de los tiempos de computación y uso de memoria**, ya que aumenta el conjunto de datos. Además, **aumenta la probabilidad de sobre-ajuste** en el entrenamiento, ya que se replican las muestras de la clase minoritaria, aumentando la probabilidad de que los clasificadores produzcan varias cláusulas en una regla para varias copias del mismo ejemplo. Esto hace que la regla se vuelva demasiado específica y por ende pueda haber sobre-ajuste. Este método es **eficiente en conjuntos de datos altamente desequilibrados**.

### 2.4.3. Sub-muestreo informado - *informed undersampling*

El sub-muestreo informado consiste en un conjunto de técnicas que seleccionan muestras de la clase mayoritaria descartando otras de una forma más avanzada que una elección aleatoria. Dentro del sub-muestreo informado podemos encontrar métodos como *EasyEnsemble* y *BalanceCascade* que intentan solventar el problema de pérdida de información; *NearMiss* que selecciona las muestras haciendo uso del clasificador *k-nearest neighbour* (KNN) y *one-side selection* (OSS) que selecciona usando técnicas de limpieza de datos.

#### *EasyEnsemble*

Este método implementa un **aprendizaje de conjunto** (*ensemble*) que tiene un enfoque de aprendizaje no supervisado. Obtiene múltiples subconjuntos de la clase mayoritaria obtenidos por muestreo aleatorio independiente con reemplazo y desarrolla múltiples clasificadores basados en la combinación de cada subconjunto de la clase mayoritaria con los datos de la clase minoritaria.

#### *BalanceCascade*

Este método tiene un enfoque supervisado. Es un método iterativo que **desarrolla múltiples clasificadores para seleccionar sistemáticamente que instancias de la clase mayoritaria sub-muestrear**.

Este método establece la primera hipótesis del conjunto H(1) considerando un conjunto de muestras $N = E \cup Smin$ siendo $E$ un subconjunto seleccionado del conjunto mayoritario de forma que $|E| = |S_{min}|$. Observa los resultados de esta hipótesis H(1) e identifica un subconjunto $N_{maj}$ correspondiente a los $x_i \in N$ que han sido correctamente clasificados como pertenecientes a $S_{maj}$. Ya que este conjunto $N_maj$ se ha usado para el entrenamiento de la hipótesis H(1) y ha sido correctamente clasificado, se asume que este subconjunto ha sido correctamente aprendido por lo que se descarta del conjunto $S_{maj}$ para la siguiente iteración y establecimiento de la siguiente hipótesis del algoritmo H(2). Se itera hasta un criterio de parada en cuyo punto se utiliza un esquema de combinación en cascada para formar una hipótesis final.

#### Técnicas *NearMiss*-v

Este método hace uso del clasificador KNN y existen cuatro versiones diferentes:

- ***NearMiss*-1:** selecciona las muestras de la clase mayoritaria cuya distancia promedio a las tres muestras de la clase minoritaria más cercanas es la menor.

– ***NearMiss*-2:** selecciona las muestras de la clase mayoritaria cuya distancia promedio a las tres muestras de la clase minoritaria más lejanas es la menor.

– ***NearMiss*-3:** para cada muestra de la clase minoritaria selecciona las n muestras más cercanas de la clase mayoritaria, obteniendo un conjunto que garantiza que cada muestra de la clase minoritaria está rodeada por el mismo número n de muestras mayoritarias.

– **MostDistant-method:** selecciona las muestras de la clase mayoritaria cuya distancia promedio a las tres muestras de la clase minoritaria más cercanas es la mayor.

La versión 2 es la que mejores resultados ha ofrecido en conjuntos de datos altamente desequilibrados.

#### Selección unilateral - *one-side Selection*

El método *one-side selection* (OSS) selecciona un subconjunto representativo de la clase mayoritaria $E$ y lo combina con el conjunto de la clase minoritaria para formar un conjunto $N = E \cup S_{min}$. Este conjunto se refina usando una técnica de limpieza de datos.

Este algoritmo inicialmente selecciona un subconjunto $N = S_{min} \cup x_i$ siendo $x_i \in S_{maj}$ y elegida al azar. Después, clasifica las instancias restantes de la clase mayoritaria $S^*_{maj} = S_{maj} - x_i$ usando el algoritmo KNN con $k = 1$ y $N$ como conjunto de entrenamiento. Como resultado, las muestras mayoritarias bien clasificadas de $S^*_{maj}$ se descartan ya que se consideran redundantes y las mal clasificadas se añaden a $N$ ya que no lo son. Por último, se usa T-link para limpieza de datos que elimina el ruido y las muestras limítrofes de la clase mayoritaria. Está recomendado su uso en conjuntos de datos poco desequilibrados. Para conjuntos de datos altamente desequilibrados se puede usar en combinación con otro algoritmo como SMOTE. La principal ventaja es que elimina las muestras de la clase mayoritaria que se consideran redundantes.

### 2.4.4. Sub-muestreo con técnicas de limpieza de datos

Este tipo de sub-muestreo consiste en la **eliminación de muestras evitando el solapamiento entre clases**. No tienen como objetivo devolver un conjunto de datos equilibrado, pero puede ser usado como método de sub-muestreo cuando se usan para eliminar instancias de la clase mayoritaria.

Dentro del sub-muestreo con técnicas de limpieza de datos podemos encontrar algunos algoritmos como los enlaces de Tomek y *edited nearest neighbor rule* (ENN).

Un "enlace Tomek", "Tomek link" o "T-link" es un par de instancias $(x_i, x_j)$ que pertenecen a distintas clases y son el par de vecinos más cercanos (mínimamente distanciados) de forma recíproca. Es decir, no existe una instancia $x_k$ que haga cumplir lo siguiente: $d(xi, xk) < d(xi, xj)$ o $d(xj, xk) < d(xi, xj)$. Si un par se considera enlace Tomek, una de esas instancias es ruido o ambas están cerca de un borde por lo que se eliminan del conjunto de datos.

Este algoritmo puede usarse para limpiar la superposición no deseada entre clases generada después de aplicar un muestreo sintético.

La principal ventaja de este método es que elimina la superposición entre clases, por lo que se obtiene como resultado grupos de clases bien definidos lo que ayuda en la generación de reglas de clasificación y mejora el rendimiento de los modelos.

### 2.4.5. Sobre-muestreo sintético

El sobre-muestreo sintético consiste en la **generación de muestras sintéticas de la clase minoritaria** con el objetivo de equilibrar el número de muestras de las clases. Dentro del sobre-muestreo sintético podemos encontrar algunos algoritmos como `SMOTE`, `Borderline-SMOTE` y `ADASYN`. Estos métodos se diferencian por la forma en la que generan las muestras, perteneciendo estos dos últimos a una variante del sobre-muestreo sintético llamada *sobre-muestreo sintético adaptativo* que intenta solventar la principal desventaja del primero.

#### *Synthetic minority over-sampling technique* - SMOTE

Este método **genera muestras sintéticas de la clase minoritaria del conjunto de datos interpolando las ya existentes**. El algoritmo calcula los K-vecinos de la clase minoritaria más cercanos a cada muestra $x_i \in S_{min}$ usando como criterio de selección la distancia euclidiana menor a lo largo de las $n$-dimensiones del espacio de características. Después, para crear una muestra sintética $x_{sint}$, selecciona aleatoriamente uno de los K vecinos más cercanos $x_k$ de una muestra $x_i$ y multiplica la diferencia de sus vectores de características correspondientes por un número aleatorio $r \in [0, 1]$ añadiendo este vector a $x_i$. Esto es: $x_{sint} = x_i + (x_k - x_i) * r$. De esta forma, la muestra sintética resultante es un punto a lo largo del segmento que une $x_i$ con $x_k$. Este proceso se repite hasta alcanzar un equilibrio entre las clases.

La principal ventaja frente a las técnicas más básicas de sobre-muestreo es que aborda el problema del sobreajuste haciendo más general el límite de decisión de la clase minoritaria. La principal desventaja es que `SMOTE` genera el mismo número de muestras sintéticas para cada muestra minoritaria original sin tener en cuenta las instancias que lo rodean, lo que provoca una generalización excesiva y varianza debido a solapamientos

entre clases. Para solventar estas debilidades existe el muestreo sintético adaptativo que incluye algoritmos como `Borderline-SMOTE` y `ADASYN`.

***Borderline-SMOTE***

Este método **genera muestras sintéticas a partir de instancias de la clase minoritaria cercanas al borde**. Para ello, calcula el conjunto de vecinos más cercanos para cada muestra $x_i \in S_{min}$ llamado $S_{i:m-NN}$ y selecciona aquellos $x_i$ que poseen más vecinos pertenecientes a la clase mayoritaria que a la minoritaria. Estos son aquellos que cumplen: $m/2 \leq |S_{i:m-NN} \cap Smaj| < m$. Estas muestras $x_i$ se introducen en un conjunto nombrado "danger". Este conjunto representa las muestras de la clase minoritaria en el borde del conjunto de datos que son aquellas muestras que tienen más posibilidades de ser mal clasificadas. Las muestras $x_i$ que están únicamente rodeadas por instancias de la clase mayoritaria se consideran como ruido y no como instancias en el borde por lo que no son añadidas a este conjunto "danger". Después de caracterizar este conjunto "danger", aplica el algoritmo original `SMOTE` sobre este conjunto "danger" para generar las muestras minoritarias sintéticas en la vecindad de los bordes.

***Adaptative synthetic sampling* - ADASYN**

Este método genera de forma sistemática y adaptada diferentes cantidades de muestras sintéticas para cada muestra minoritaria de acuerdo a su distribución.

Primero, calcula la cantidad de muestras sintéticas minoritarias a generar de la siguiente forma: $G = (|S_{maj}| - |S_{min}|) \times B$, donde $B \in [0, 1]$ es un parámetro que especifica el nivel de balanceo deseado. Después, calcula el conjunto de k vecinos más cercanos para cada muestra $x_i \in Smin$ y calcula la relación $l_i$ de la siguiente forma: $li = \frac{\nabla_i/k}{Z}, i = 1, .., |Smin|$, donde $\nabla_i$ es el número de muestras de los k vecinos más cercanos de $x_i$ que pertenecen a $S_maj$ y $Z$ es una constante de normalización tal que $l_i$ es una función de distribución, $\sum l_i = 1$. Luego, determina el número de muestras sintéticas que han de ser generadas para cada $x_i$ de la siguiente forma: $gi = li \times G$. Finalmente, para cada $x_i$ se generan $g_i$ muestras sintéticas usando el algoritmo original `SMOTE`.

## 2.5. Selección de características

### 2.5.1. Métodos de análisis uni-variable

Estos métodos analizan únicamente la **dependencia que existe entre cada una de las variables una a una y la variable objetivo**. Este tipo de métodos **no**

**analiza las relaciones entre variables**, por lo que puede seleccionar características muy parecidas o redundantes.

### 2.5.2. Métodos de análisis multivariable

Estos métodos **analizan la dependencia que existe entre un conjunto de variables y la variable objetivo**. Este tipo de métodos analiza las relaciones entre variables, por lo que permite descartar variables redundantes o irrelevantes.

### 2.5.3. Métodos de filtro

Los **métodos de filtro** son utilizados en una etapa de pre-procesamiento y son independientes del algoritmo de Machine-Learning que se use. Estos métodos evalúan cada una de las características en base a criterios estadísticos puntuándolas por su correlación con la variable de salida objetivo.

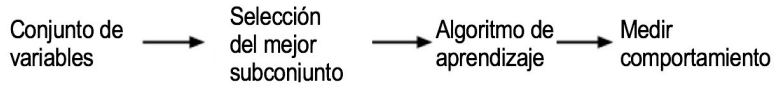



Figura 2.3: Métodos de filtro [31]

Las principales ventajas son que es independiente del algoritmo, el tiempo de computación es inferior a otros métodos, y que reduce el *overfitting*.

El principal inconveniente es que pueden fallar buscando el mejor subconjunto de características, ya que se basan en métodos estadísticos y no tienen en cuenta el algoritmo.

La biblioteca `scikit-learn` [32, 33] ofrece algunos métodos de filtro y análisis univariante para selección de características a los cuales denominan "Univariate feature selection". Estos son: "F-ANOVA" y "mutualInfo" y son descritos a continuación [34].

#### F-test, f-score o valor f-ANOVA

Estima el grado de dependencia lineal entre cada una de las variables una a una contra la variable objetivo. Funciona bien con pocas muestras.

- Fórmula:

$$F = \frac{\frac{SCE}{gl_E}}{\frac{SCD}{gl_D}} = \frac{\frac{\sum_{i}^{m} n(\overline{X_i} - \overline{X})^2}{m-1}}{\frac{\sum_{i}^{m}\sum_{j}^{n}(X_{ij} - \overline{X_i})^2}{m(n-1)}} \tag{2.1}$$

donde:

- SCE: suma de cuadrados entre los grupos. Explica la variación entre los grupos, esto es la suma de los cuadrados de las diferencias de cada media grupal con la media total, multiplicado por el número de muestras y dividido entre sus grados de libertad.
- SCD: suma de cuadrados dentro de los grupos. Explica la variación dentro de los grupos, esto es la suma de los cuadrados de las diferencias de cada muestra respecto a su media grupal, dividido por sus grados de libertad.
- m: número de grupos.
- n: número de muestras de cada grupo.

– Hipótesis:

- $H_0$: $\overline{X_1} = ... = \overline{X_m}$, es decir, la característica no marca una diferencia para los distintos grupos, no es relevante.
- $H_1$: no se cumple $H_0$, es decir, la característica marca una diferencia para los distintos grupos, es relevante.

– Interpretación valor F:

- Valor alto de F ($SCE \gg SCD$): la variación total viene en su mayoría de la variación existente entre los grupos, es difícil que se cumpla $H_0$, por lo que la característica marca una diferencia para los grupos y podría ser una **característica relevante**.

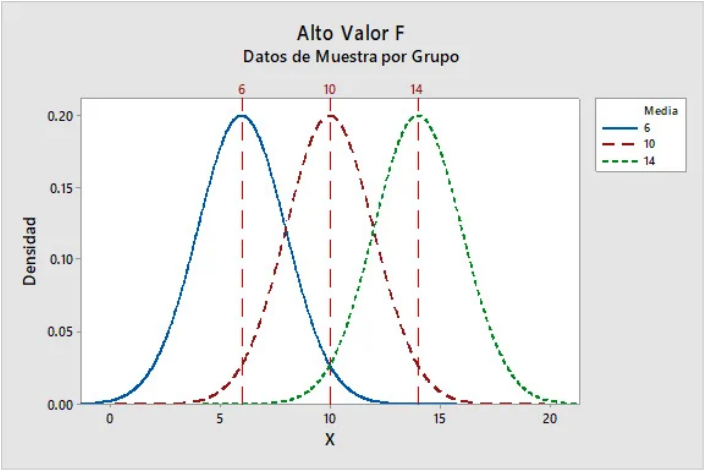



Figura 2.4: Valor alto de F [35]

- Valor bajo de F ($SCD \gg SCE$): la variación total viene en su mayoría de la variación dentro de los grupos, es fácil que se cumpla $H_0$, por lo que la característica no marca una diferencia para los grupos y se podría considerar una **característica menos relevante**.

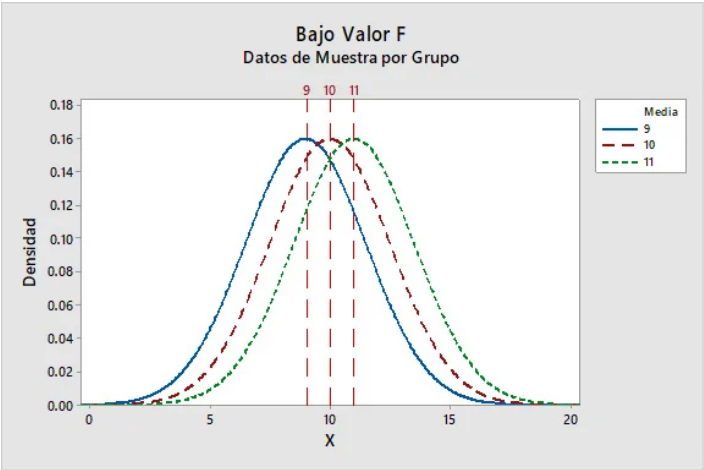



Figura 2.5: Valor bajo de F [35]

- Valor crítico de F ($F_{m-1,m(n-1)}$): es el valor que permite aceptar o rechazar la $H_0$, viene dado por la distribución F de Fisher con $m-1$ gl para el numerador y $m(n-1)$ gl para el denominador dado un nivel de significancia $\alpha$.

– Aceptar o rechazar la hipótesis dado un nivel de significancia $\alpha$:

- Valor F:
  - $F < F_{m-1,m(n-1)}$: cumple la $H_0$, no es una característica relevante.

- $F > F_{m-1,m(n-1)}$: rechaza la $H_0$, es una característica relevante.

- Valor p: el p-valor es la probabilidad de que el valor estadístico F calculado sea posible dado que la hipótesis $H_0$ sea cierta.
  - $p - valor > \alpha$: cumple la $H_0$, no es una característica relevante.
  - $p - valor < \alpha$: rechaza la $H_0$, es una característica relevante.

**Información mutua - *Mutual-information***

Permite estimar cualquier dependencia (no como F-test), aunque necesita un número de muestras mayor para funcionar de forma correcta.

– Información mutua (MI) entre dos variables aleatorias: valor no negativo que mide la dependencia mutua entre las variables. Cuantifica la "cantidad de información" obtenida sobre una variable aleatoria al observar la otra variable aleatoria. Este concepto está ligado al de entropía, por ello, es necesario definir primero este concepto.

- Entropía de una variable aleatoria H(X): cuantifica la "cantidad de información" esperada contenida en una variable aleatoria. Se entiende como una medida de incertidumbre: cuanto más grande su valor, más incertidumbre sobre el valor que tomará; cuanto menos valor, más información sobre dicho valor. La fórmula es la siguiente:

$$H(X) = -\sum_{x \in X} p(x) log(p(x)) \tag{2.2}$$

  donde $p(x)$ es la función de densidad de probabilidad marginal de la variable aleatoria X, y *log* puede ser el logaritmo natural, en base 2 o en base 10, produciendo correspondientemente unidades de nats, bits o Hartleys.

- Entropía conjunta de dos variables aleatorias X y Y H(X,Y):

$$H(X,Y) = -\sum_{x \in X} \sum_{y \in Y} p(x,y) log(p(x,y)) \tag{2.3}$$

- Entropía condicional $H(Y|X)$: incertidumbre que se tiene sobre Y cuando se conoce el valor de X.

$$H(Y|X) = \sum_{x \in X} p(x) H(Y|X = x) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) log(p(y|x)) \tag{2.4}$$

– La información mutua MI determina que tan diferente es la distribución conjunta de los datos $P_{(X,Y)}$ de la distribución que éstos tendrían si X e Y fueran independientes, esto es el producto de las distribuciones marginales ($P_{(X,Y)} = P_X * P_Y$) en caso de independencia lineal. Dicha medida es conocida como la divergencia Kullback-Leibler, la fórmula es la siguiente:

$$MI(X;Y) = D_{KL}(P_{(X,Y)} \parallel P_X \otimes P_Y) \tag{2.5}$$

siendo $D_{KL}(P \parallel Q)$ la divergencia Kullback-Leibler. Es un tipo de distancia estadística, una medida de cómo una distribución de probabilidad P es diferente de una segunda distribución de probabilidad de referencia Q. Su fórmula es la siguiente:

$$D_{KL}(P \parallel Q) = \sum_{x \in X} P(x) log \frac{P(x)}{Q(x))} \tag{2.6}$$

– Estimación de las funciones de densidad de probabilidad FDP p(x,y), p(x), p(y): son desconocidas a priori y necesarias para la estimación de la MI. Para solucionar el problema, los valores deben ser discretizados o sus densidades han de ser aproximadas mediante el uso de métodos paramétricos o no paramétricos. En el caso de variables discretas, al trabajar con sumatorios, las probabilidades pueden ser estimadas del conteo de frecuencia en los datos.

  - Métodos paramétricos: suponen que la FDP a estimar pertenece a una clase de funciones paramétricas como la distribución normal, exponencial, etc.
  - Métodos no paramétricos: no suponen nada, sino que dejan que los datos sean los que estimen la distribución. Algunos métodos son: el histograma, estimador Naive, estimador de núcleo, estimador del vecino más cercano, etc. En el caso de `sklearn`, usa métodos no paramétricos para estimar la entropía a partir de las distancias de k vecinos más cercanos.

– Interpretación:

  - $MI = 0$: la distribución conjunta coincide con el producto de las marginales, es decir, las variables son independientes. En este caso la información que nos aporta la característica X sobre la variable objetivo Y es 0 por lo que se considera que la **característica es no relevante**.

- $MI > 0$: mayor dependencia. En este caso la variable X si aporta información acerca de la variable objetivo Y, por lo que se podría considerar una **característica relevante**.

### Umbral de varianza

Elimina todas las características cuya varianza no alcanza algún umbral. Este método supone que las características con mayor varianza aportarán información más útil, pero no tiene en cuenta la relación de las características con la variable objetivo. Este método parece útil para eliminar las características que no aportan ninguna información, es decir, las de varianza 0. Este método parece menos útil que los anteriores para nuestro proyecto, ya que en nuestro caso calculamos nosotros las características y este caso no se da, por lo que no lo utilizaremos.

### Diferencia absoluta media - *mean absolute difference*

El *mean absolute difference* (MAD) calcula la media de las diferencias absolutas del valor medio. La principal diferencia con el anterior que usa la varianza es la ausencia del cuadrado en esta medida. Al igual que la anterior medida, asume que a mayor MAD, mayor cantidad de información aporta la característica. Esto es:

$$MAD(X) = \frac{1}{N} \sum_{x \in X} |x - \overline{X}| \tag{2.7}$$

### Relación de correlación y de dispersión

Estas relaciones se utilizan como un análisis bi-variable o multivariable de las características para eliminar características redundantes. La relación de correlación es la medida de relación lineal entre 2 o más variables. Si dos características están muy correlacionadas, implica que se puede predecir una en base a la otra, por lo que aportan información muy similar y el modelo no necesita las dos. Se puede utilizar para eliminar características muy similares o redundantes.

### Selección de atributos correlacionados - *correlated features selection*

El método *correlated features selection* (CFS) identifica un subconjunto de características muy correlacionados con la clase objetivo, sin estar éstos fuertemente correlacionados entre sí. Lo que hace esta técnica es buscar entre los subconjuntos de características posibles para obtener "el mejor" mediante un método de búsqueda concreto. La forma en la que evalúa estos subconjuntos es calculando la media de las

correlaciones de cada característica con la clase y las correlaciones por redundancia entre características. Es un método rápido, pero, aunque permite eliminar redundancia, también puede eliminar características que no están correlacionadas directamente con la clase pero sí con otras características.

Existen otros métodos en la literatura para selección de características como ReliefF y Branch and Bound (B&B)

### 2.5.4. Métodos envolventes - Wrapper

Los **métodos envolventes** entrenan un algoritmo de aprendizaje utilizando diferentes tamaños y combinaciones de subconjuntos de variables. Este tipo de métodos toma decisiones de añadir o eliminar variables del subconjunto en función del buen comportamiento de los modelos entrenados con el objetivo de mejorar el resultado y evalúan el subconjunto final con validación cruzada. Estos métodos convierten el problema de selección de características en un problema de búsqueda del subconjunto óptimo. La figura 2.6 muestra un diagrama de cómo funcionan dichos métodos.

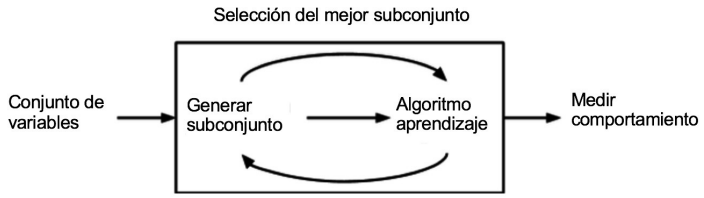



Figura 2.6: Métodos envolventes. [31]

Las principales ventajas son que reducen el *overfitting* y suelen acertar con el subconjunto óptimo, ya que tienen en cuenta las relaciones entre variables y el algoritmo.

Las principales inconvenientes son que son dependiente del algoritmo, que es más probable tener *overfitting* que usando métodos de filtro, y que el tiempo de computación es más elevado que en otros métodos ya que ha de entrenar los modelos para tomar decisiones.

Existen diferentes métodos para encontrar el subconjunto "óptimo" como: Selección hacia delante, Eliminación hacia atrás y Eliminación de características recursivas. Estos son descritos a continuación.

#### Selección hacia delante - *forward selection*

Comienza con un subconjunto vacío y de forma iterativa va agregando la característica que mejora más el modelo de aprendizaje. Finaliza cuando añadir una característica no ofrece ninguna mejora en el rendimiento del modelo.

**Eliminación hacia atrás - *backward elimination***

Comienza con el conjunto completo de todas las características y de forma iterativa va eliminando la característica menos significativa, es decir, aquella cuya eliminación mejora más los resultados del modelo. Finaliza cuando eliminar una característica no ofrece ninguna mejora en el rendimiento del modelo.

**Eliminación de características recursivas - *recursive feature elimination***

Es un algoritmo de optimización que busca encontrar el subconjunto de funciones con mejor rendimiento. Crea repetidamente modelos y deja de lado la mejor o la peor característica de rendimiento en cada iteración. Construye el siguiente modelo con las características de la izquierda hasta que se agotan todas las características, luego clasifica las características según el orden de su eliminación.

### 2.5.5. Métodos integrados o embebidos

Este tipo de métodos combina los dos anteriores, aprovechando las ventajas de cada uno de ellos. Generan subconjuntos basados en decisiones estadísticas y entrenamientos de algoritmos simultáneamente.

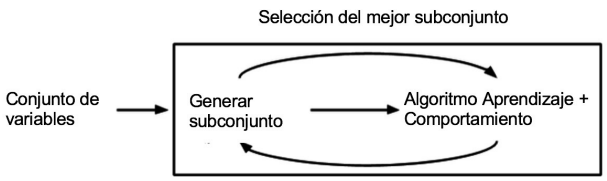



Figura 2.7: Métodos embebidos [31]

Algunos ejemplos de estos métodos son: árboles de clasificación (*Classification Trees*) o bosques aleatorios (*Random Forests*)

## 2.6. Aprendizaje automático

El *Machine Learning* (ML) o **aprendizaje automático** es una rama de la inteligencia artificial que, a través de algoritmos, dota a las máquinas de la capacidad de "aprender" mediante la identificación de patrones en datos masivos y elaborar predicciones, sin ser expresamente programadas para ello [36]. El objetivo es crear un modelo que nos permita resolver una tarea dada. En función de la tarea que se quiera aprender resulta muy importante disponer de un gran volumen de datos de calidad que se ajusten al objetivo a aprender, así como la elección del algoritmo más adecuado y su

posterior parametrización de forma adecuada al problema. Esto permitirá obtener un modelo el cuál al ser entrenado identificará patrones en los datos y será capaz de hacer predicciones correctas de la tarea dada.

Dentro de esta área se encuentran distintos tipos de aprendizaje. Las tres grandes corrientes dentro del ML son [37, 38]: aprendizaje no supervisado, aprendizaje supervisado, aprendizaje de refuerzo y *Deep Learning*.

### 2.6.1. Aprendizaje no supervisado

En el **aprendizaje no supervisado**, los algoritmos trabajan con datos de entrada no etiquetados. Ya que no disponen de "etiquetas" o datos de salida para el entrenamiento, estos algoritmos se centran en describir la estructura de los datos con el objetivo de encontrar algún tipo de organización que simplifique el análisis. Tienen carácter exploratorio. Este tipo de aprendizaje se suele usar en problemas de clustering, agrupamientos de co-ocurrencias y perfilado o *profiling*.

En el contexto de este trabajo, en el que los datos los tenemos etiquetados, se pueden usar para buscar agrupamientos basados en similitudes de los datos. Aunque el resultado de estos algoritmos no nos asegure un significado o utilidad, este tipo de aprendizaje nos permite explorar el conjunto de datos. En el caso de usarlos en el trabajo será con el objetivo de encontrar correlaciones útiles para el descubrimiento de nuevos parámetros significativos para la tarea objetivo, así como descartar otros parámetros que no ayuden significativamente en la tarea.

Los algoritmos más habituales de aprendizaje no supervisado son: algoritmos de clustering, análisis de componentes principales y descomposición de valores singulares.

### 2.6.2. Aprendizaje supervisado

En el **aprendizaje supervisado**, los algoritmos trabajan con datos etiquetados de los cuales intentan extraer una función que, en base a las variables de entrada, les asigne la etiqueta de salida correcta. De esta forma, dados unos datos de entrada, el algoritmo "aprende" a predecir la etiqueta de salida adecuada. Este tipo de aprendizaje se suele usar en problemas de clasificación y problemas de regresión.

Estos dos tipos de aprendizaje supervisado, clasificación y regresión, se distinguen por el tipo de variable objetivo. En los problemas de clasificación la variable objetivo es de tipo categórico, mientras que en los de regresión es de tipo numérico.

Los algoritmos más habituales que aplican para el aprendizaje supervisado son: árboles de decisión, clasificación de Naive Bayes, regresión por mínimos cuadrados, regresión logística, support vector machines (SVM), métodos "Ensemble".

### 2.6.3. Aprendizaje por refuerzo

El **aprendizaje por refuerzo**, basa su aprendizaje en un proceso de retroalimentación. El algoritmo tiene como entrada el *feedback* que obtiene del exterior como respuesta a sus acciones y mejora el modelo en base a la monitorización de dichas entradas. Este tipo de aprendizaje es a base de prueba-error, aprende en base a la experiencia.

### 2.6.4. *Deep Learning*

El *Deep Learning* está inspirado por la estructura y función del cerebro humano, se basado en redes neurales artificiales en vez de sólo conceptos estadísticos. Este tipo de aprendizaje se puede usar para ambos enfoques, supervisado y no supervisado.

## 2.7. Modelos para clasificación supervisada

Esta sección basa su contenido en las explicaciones aportadas por la biblioteca [33].

### 2.7.1. Modelos lineales

#### *Ridge Classifier*

Para tareas de clasificación, este modelo trata el problema como una tarea de regresión. La clase predicha corresponde al signo de la predicción del regresor. Para clasificación multiclase, el problema se trata como una regresión con múltiples salidas siendo la clase predicha la correspondiente a la salida con el valor más alto -1, 1.

La regresión *Rigde*, también denominada "regresión contraída" o "Tikhonov regularization", regulariza un modelo lineal imponiendo una penalización en el tamaño de los coeficientes de la relación lineal. En este caso, los coeficientes calculados minimizan una suma de cuadrados residual penalizada añadiendo el cuadrado de la norma L2. Matemáticamente, resuelve el siguiente problema:

$$min_w \|Xw - y\|_2^2 + \alpha \|w\|_2^2 \tag{2.8}$$

donde:

- El parámetro $\alpha$, o parámetro de complejidad, debe de ser $\geq 0$, controla el grado de penalización o la cantidad de contracción. Cuando $\alpha = 0$ es equivalente a la regresión lineal y cuanto mayor es este valor, los coeficientes son menores siendo más robustos a la colinealidad.

– Penalización: usa el cuadrado de la norma L2. La norma L2 es la distancia Euclídea del vector al centro de coordenadas, por lo que la penalización es proporcional a la suma de cuadrados de los coeficientes.

### 2.7.2. Máquinas de Vectores de Soporte - *support vector machines*

Máquinas de vectores de soporte, o *support vector machines* (SVM), es un algoritmo de aprendizaje que puede ser usado para problemas de clasificación y regresión. Este algoritmo traza cada observación como un punto en un espacio dimensional p, siendo p el número de características de las muestras. El algoritmo intenta encontrar un hiper-plano que permita separar bien las diferentes clases y sea el "óptimo", es decir, aquel hiper-plano que maximiza la distancia mínima a los puntos de las clases. Si el hiper-plano existe, se le denomina "hiper-plano de máximo margen"; al conjunto de puntos formado por el punto de cada clase más próximo al hiper-plano se les denomina "vectores de soporte", son los que determinan la posición del hiper-plano; y la distancia mínima del hiper-plano a los vectores de soporte, se le denomina "margen". En general, cuanto mayor es este margen, mayor es la capacidad de generalización del algoritmo.

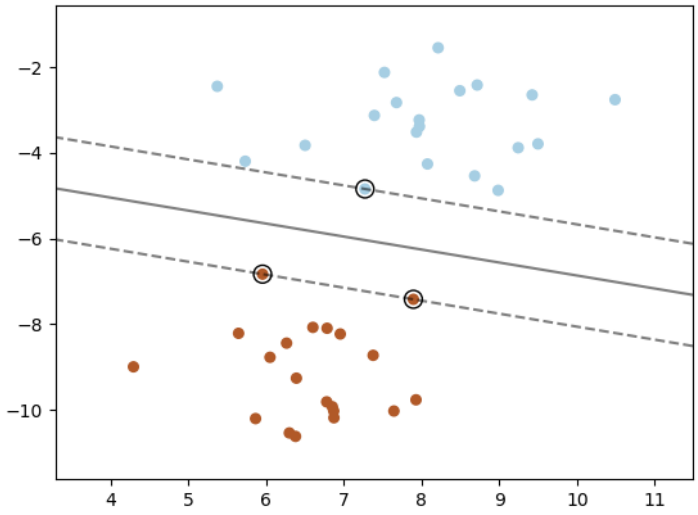



Figura 2.8: Esquema SVM [33]

Dado un conjunto de entrenamiento, donde $x_i \in \mathbb{R}^p, i = 1..n$ son los conjuntos de p características de las $n$ muestras, e $y_i \in$ {-1, 1} $^n, i = 1..n$ es el resultado objetivo de las $n$ muestras. Matemáticamente, un hiper-plano de separación, definido como una función lineal, que permite separar dicho conjunto en sus diferentes clases se define de la siguiente forma:

$$\begin{aligned} y_{pred}(x_i) &= (w_1 x_1 + .. + w_p x_p) + b \\ &= w^T x_i + b \end{aligned} \tag{2.9}$$

$$\begin{aligned} y_{pred}(x_i) \geq 0 \quad \text{si} \quad y_i = +1 \\ \leq 0 \quad \text{si} \quad y_i = -1 \end{aligned} \tag{2.10}$$

$$y_i y_{pred}(x_i) \geq 0, i = 1, ..., n \tag{2.11}$$

donde $w \in \mathbb{R}^p$ y $b \in \mathbb{R}$.

Dentro de los infinitos hiper-planos que pueden existir para separar las clases, el "óptimo" es aquel que maximiza el margen, siendo el margen la distancia mínima a los puntos de las clases. Por lo que encontrar el hiper-plano óptimo, consiste en encontrar el w que maximice el margen, es decir:

$$\begin{aligned} margen_{opt} &= max_{w \in \mathbb{R}^p}(min_{i=1..n}(d(x_i, y_{pred}(x)))) \\ &= max_{w \in \mathbb{R}^p}(min_{i=1..n}(\frac{|y_{pred}(x_i)|}{\|w\|_2})) \\ &= max_{w \in \mathbb{R}^p}(min_{i=1..n}(\frac{y_i y_{pred}(x_i)}{\|w\|_2})) \end{aligned} \tag{2.12}$$

Como existen infinitas soluciones, se fija lo siguiente:

$$margen \|w\|_2 = 1 \tag{2.13}$$

De la ecuación 2.13 se deduce que maximizar el margen implica minimizar w y sustituyendo en la ecuación 2.12 lo visto en 2.13 y verificando que se sigue cumpliendo lo de 2.11, el problema queda matemáticamente resuelto de la siguiente manera:

$$\begin{aligned} min \quad & \frac{1}{2}\|w\|_2^2 = \frac{1}{2} w^T w \\ sujetoa \quad & y_i y_{pred}(x) \geq 1 \\ & i = 1, .., n \end{aligned} \tag{2.14}$$

Lo ideal es que se cumpla lo de la ecuación 2.14, pero el problema no siempre es perfectamente separable en un hiper-plano, por lo que se permite que algunas muestras estén a una cierta distancia $C_i$ desde su límite de margen correcto. Modificándose de la siguiente manera:

$$\begin{aligned} min_{w,b,\text{Ç}} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{n} \text{Ç}_i \\ sujetoa \quad & y_i y_{pred}(x) \geq 1 - \text{Ç}_i \\ & \text{Ç}_i \geq 0, i = 1, .., n \end{aligned} \tag{2.15}$$

donde $C$ es un término de penalización que controla la intensidad de la penalización cuando una muestra está mal clasificada o dentro del límite del margen. Actúa como un parámetro de regularización inversa. Respecto al clasificador *Ridge*, explicado previamente, $C = \frac{1}{\alpha}$.

Este algoritmo es muy efectivo cuando el número de características es muy alto o si el número de características es mayor al número de muestras de datos. Ya que este algoritmo funciona con vectores, es muy importante normalizar los datos antes de usarlos.

### 2.7.3. Descenso de gradiente estocástico - *stochastic gradient descent*

El método *stochastic gradient descent* (SGD) es un método iterativo de optimización que se basa en minimizar el gradiente de una función objetivo. La razón de minimizar el gradiente es el hecho de que los puntos extremos de una función tiene derivada igual a 0. Debido a que, en general, una función puede tener diversos mínimos o máximos locales, el punto hacia el cual el gradiente pueda converger puede depender del punto de inicio de la iteración, que puede estar más cerca o lejos de un determinado punto mínimo o máximo. Para reducir esa posibilidad, el punto de inicio se toma al azar dentro del espacio de parámetros posibles de la función objetivo. Este proceso se repite de inicio y convergencia hacía un mínimo o máximo se repite una serie de veces para asegurar la consistencia de la solución.

### 2.7.4. Clasificación de vecinos más cercanos

La clasificación basada en vecinos predice la clase de un punto en función de las clases que tengan sus vecinos más cercanos. En su versión más básica utiliza ponderaciones uniformes, es decir, como clase predicha se asigna la mayoría simple de votos. En otras circunstancias, se puede ponderar a los vecinos de manera que los vecinos más cercanos contribuyan más al ajuste. Existen varios algoritmos en `sklearn` [39]:

- ***K-NeighborsClassifier*:** implementa el aprendizaje basado en los k vecinos más cercanos al punto de consulta.

- ***RadiusNeighborsClassifier*:** implementa el aprendizaje basado en el número de vecinos dentro de un radio fijo r al punto de consulta.

- ***NearestCentroid*:** representa cada clase con el centroide de los miembros que la forman (paso similar a la fase de actualización de etiquetas del algoritmo *k-means*).

### 2.7.5. Árboles de decisión

Los árboles de decisión dividen recursivamente el espacio de características de forma que generan un árbol cuyos nodos contienen condiciones que en función de su cumplimiento o no derivan a un nodo hijo u otro hasta llegar a un nodo hoja donde las muestras con mismas etiquetas se agrupan. Las predicciones se obtienen haciendo pasar a los nuevos puntos por dicho árbol y devolviendo la etiqueta del nodo hoja al que lleguen.

Cada nodo interno $m$ contiene un conjunto de datos $Q_m$ con $n_m$ muestras y realiza una división de candidatos $\theta = (j, t_m)$, siendo $j$ una característica y $t_m$ un umbral, que resulta en la división de sus datos en dos nodos hijos:

$$\begin{aligned} Q_m^{left}(\theta) &= \{(x, y) | x_j < t_m\} \\ Q_m^{right}(\theta) &= Q_m \setminus Q_m^{left}(\theta) \end{aligned} \tag{2.16}$$

La calidad de una división candidata $\theta$ del nodo $m$ se calcula de la siguiente manera:

$$G(Q_m, \theta) = \frac{n_m^{left}}{n_m} H(Q_m^{left}(\theta)) + \frac{n_m^{right}}{n_m} H(Q_m^{right}(\theta)) \tag{2.17}$$

donde $H$ es una función de impureza o pérdida. Por lo tanto, para cada división hay que seleccionar los parámetros que minimicen la impureza:

$$\theta^* = argmin_\theta G(Q_m, \theta) \tag{2.18}$$

Se realiza recursivamente hasta alcanzar la profundidad máxima permitida ($n_m < min_{samples}$ o $n_m = 1$.

Medidas comunes de impureza:

- Gini:

$$H(Q_m) = \sum_k p_{mk}(1 - p_{mk}) \tag{2.19}$$

- Pérdida de registro o entropía:

$$H(Q_m) = -\sum_k p_{mk} log(p_{mk}) \tag{2.20}$$

### 2.7.6. Métodos de conjunto - *ensemble methods*

Los métodos de conjunto agregan las predicciones de varios estimadores base creados con un algoritmo de aprendizaje determinado con el objetivo de mejorar la generalización/robustez sobre un solo estimador. Existen dos tipos de métodos de conjunto:

- **Métodos de promediación:** se construyen varios estimadores de forma independiente y se promedian las predicciones de los clasificadores individuales. A este tipo pertenecen los métodos de embolsado, bosques de árboles aleatorios, etc.
- **Métodos de impulso (*"boosting"*):** se construyen varios estimadores de forma secuencial intentando reducir el sesgo del estimador combinado. A este tipo pertenece `AdaBoost`, mejora del árbol de gradiente, etc.

#### Métodos de embolsado - *bagging methods*

Los métodos de embolsado construyen varias instancias de un estimador que usan como datos de entrenamiento subconjuntos aleatorios del conjunto de entrenamiento original y agregan las predicciones individuales de cada estimador para formar una predicción final.

Los métodos de conjunto se diferencian principalmente por la forma en la que extraen subconjuntos aleatorios del conjunto de entrenamiento:

- Pegado ("pasting"): se extraen como subconjuntos aleatorios de las muestras.
- Embolsado ("bagging"): las muestras se extraen con reemplazo.
- Sub-espacios aleatorios ("random subspaces"): se extraen como subconjuntos aleatorios de las características.
- Parches aleatorios ("random patches"): se extraen como subconjuntos de muestras y características.

#### Bosques de arboles aleatorios

Crean un conjunto de arboles a los que introducen aleatoriedad en su construcción mediante técnicas de perturbación y combinación.

- ***RandomForest*:** cada árbol del conjunto se construye a partir de una muestra extraída con reemplazo del conjunto de entrenamiento y en cada nodo encuentra la mejor división para las características de entrada (todas o un subconjunto aleatorio).

– ***ExtraTrees*:** añade más aleatoriedad respecto *RandomForest*, ya que elige la mejor división entre un conjunto de umbrales aleatorios para cada característica.

#### Métodos de impulso - *"Boosting"*

`AdaBoost`: su objetivo es ajustar una secuencia de modelos débiles usando distintas versiones de los datos, modificadas en cada iteración. Para modificar los datos, se aplican pesos a las muestras. En cada iteración, se modifican las ponderaciones de las muestras individualmente y el algoritmo de aprendizaje se vuelve a aplicar a los datos re-ponderados. En cada iteración, los pesos de las muestras mal clasificadas aumentan, mientras que los de las clasificadas correctamente disminuyen. De esta forma, cada modelo débil posterior se centra en las muestras más difíciles de predecir, que los modelos previos en la secuencia no pudieron clasificar correctamente. Finalmente, las predicciones de estos modelos se combinan mediante una mayoría ponderada de votos para generar la predicción final.

#### Árboles de decisión impulsados por gradiente - *gradient boosted decision trees*

Los *gradient boosted decision trees* (GBDT) es una evolución de los bosques aleatorios en la que la búsqueda de las ramas se realiza mediante la minimización de una función objetivo que intenta predecir la posible secuencia de la rama.

### 2.7.7. Modelos que implementan redes neuronales

#### Perceptrón multicapa - MLP

Es un algoritmo de aprendizaje que aprende una función $f(\dot{}) : \mathbb{R}^p - > \mathbb{R}^o$ mediante el entrenamiento de un conjunto de datos, siendo $p$ el número de dimensiones en la entrada (características) y $o$ el número de dimensiones para la salida (valores objetivo).

Este modelo consta de varias capas:

– Capa de entrada: consta de un conjunto de neuronas $\{x_1, ..., x_p\}$ que representan las características de entrada.

– Capas ocultas: cada neurona de la capa transforma los valores de la capa anterior utilizando una suma lineal ponderada $w_1x_1 + ... + w_px_p$ seguido de una función de activación no lineal $g(\dot{}) : \mathbb{R} - > \mathbb{R}$

– Capa de salida: transforma los valores de la última capa oculta en los valores de salida.

Dados unos datos de entrenamiento, donde $x_i \in \mathbb{R}^p$ es el conjunto de características e $y_i \in \{0, 1\}$ es el conjunto objetivo, y un MLP con una capa oculta y una neurona oculta, el modelo aprende la función $f(x) = w_2 g(w_1^T x + b_1) + b_2$, donde $w_1 \in \mathbb{R}^p$ son los pesos de la capa de entrada, $w_2, b_1 \in \mathbb{R}, g(\dot{}) : \mathbb{R} - > \mathbb{R}$ son los pesos, el sesgo agregado y la función de activación de la capa oculta, respectivamente, y $b_2 \in \mathbb{R}$ es el sesgo agregado a la capa de salida.

Inicialmente, el algoritmo empieza con unos pesos aleatorios y progresivamente va actualizando esos pesos tratando de minimizar una función de pérdida. Al calcular la pérdida, la propaga desde la capa de salida a las capas anteriores (*backward-propagation*), proporcionando a cada peso un valor actualizado con el objetivo de disminuir la pérdida. Finalmente, el algoritmo se detiene cuando alcanza un número máximo preestablecido de iteraciones o cuando la mejora de la pérdida es pequeña, inferior a un valor determinado.

En descenso de gradiente, se calcula $w$ de la siguiente manera:

$$w^{i+1} = w^i - \in \nabla Loss_w^i \tag{2.21}$$

donde $i$ es el número de iteración, $\in$ es la tasa de aprendizaje, y $\nabla Loss_w^i$ es el gradiente de la pérdida con respecto a los pesos.

La función de pérdida para clasificación es la entropía cruzada promedio. En el caso de clasificación binaria viene dada por la siguiente fórmula:

$$Loss(y, y, w) = -\frac{1}{n}\sum_{i=0}^{n}(y_i ln y_i + (1 - y_i)) + \frac{\alpha}{2n}\|w\|_2^2 \tag{2.22}$$

La función de activación de salida varía en función del tipo de clasificación:

- Clasificación binaria: usa la función logística para obtener valores de salida en el rango {0, 1}.

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.23}$$

- Clasificación multiclase: usa la función *softmax* para obtener un vector de probabilidades, que indica la probabilidad de que la muestra x pertenezca a cada una de las clases.

$$softmax(z)_i = \frac{exp(z_i)}{\sum_{l=1}^{c} exp(z_l)} \tag{2.24}$$

  donde $z_i$ representa el i-ésimo elemento de la entrada a *softmax* que corresponde a la clase $i$, y $c$ es el número de clases.

## 2.8. Validación de modelos

Con el objetivo de elegir el modelo más adecuado para el problema planteado es importante considerar los siguientes apartados.

### 2.8.1. Capacidad de generalización del modelo

Como resultado de un entrenamiento podemos clasificar su aprendizaje de tres formas: sobre-generalizado (*underfit*), bien entrenado (*good fit*) o sobre-ajustado (*overfit*) [40, 41, 42].

Un modelo de aprendizaje está **sobre-generalizado** si no puede generalizar los datos de entrenamiento ni las nuevas observaciones (datos de test). Este caso se puede identificar cuando el rendimiento del modelo es muy bajo tanto en entrenamiento como test, ya que el modelo entrenado no predice bien ni los datos de entrenamiento reales.

Un modelo de aprendizaje está **sobre-ajustado** si ha aprendido demasiado bien el conjunto de datos de entrenamiento, incluido el ruido o las fluctuaciones aleatorias, lo que le impide "generalizar" y predecir bien las nuevas observaciones. Este caso se puede identificar cuando el rendimiento del modelo es muy alto en entrenamiento y bajo en test.

Un modelo de aprendizaje está **bien entrenado** si presenta un modelo que está entre *overfit* y *underfit*, siendo este el objetivo del algoritmo de aprendizaje. Se puede identificar por un rendimiento en entrenamiento y test similares, existiendo una variación mínima entre ellos. Esto se debe a que casi siempre va a existir una brecha entre ellos ("brecha de generalización"), ya que el modelo tenderá a tener mejor rendimiento para los datos con los que ha entrenado.

### 2.8.2. Aspectos importantes para la evaluación

Para la selección del "mejor modelo" es importante considerar los siguientes aspectos durante la evaluación del modelo:

- Que no haya fuga de datos en el modelo entrenado: en aprendizaje automático, la fuga de datos es el uso de información en el proceso de formación del modelo que no se esperaría que estuviera disponible en el momento de la predicción. Esto hace que las métricas aportadas durante la evaluación del modelo sobre-estimen la utilidad de éste.

- Que la forma de obtener el mejor modelo y evaluarlo sea robusta: elegir el mejor modelo sobre una partición de datos de entrenamiento y test (80-20) y tomar como medida de evaluación la obtenida por el mejor modelo, sin re-evaluar sobre

otro conjunto de datos no visto puede sobre-estimar la calidad del modelo, ya que se ha elegido el mejor modelo para esta partición de datos entrenamiento-prueba. Para ello, existen técnicas como la validación cruzada.

## 2.9. Métricas de evaluación

Para evaluar la precisión del modelo existen diversas métricas, dependiendo del problema a tratar. Algunas de las más comunes son:

- **Exactitud (*accuracy*):** es la proporción de predicciones correctas respecto al total de predicciones realizadas. Se calcula como el número de predicciones correctas dividido entre el número total de predicciones. Esto es: $(tp+tn)/(tp+fp+tn+fn)$.

- **Precisión (*precision*):** es la proporción de predicciones positivas que son verdaderas positivas respecto a todas las predicciones positivas realizadas. Se calcula como el número de verdaderos positivos dividido entre el número de verdaderos positivos más falsos positivos. Esto es: ...

- **Recuperación (*recall*):** es la proporción de verdaderos positivos que son detectados como positivos respecto a todos los verdaderos positivos que existen en el conjunto de datos. Se calcula como el número de verdaderos positivos dividido entre el número de verdaderos positivos más falsos negativos.

- **Puntuación F1 (F1 score):** es la media armónica entre precisión y recuperación. Se calcula como 2 veces el producto de precisión y recuperación dividido entre la suma de precisión y recuperación.

Para evaluar el aprendizaje del modelo, se puede usar el concepto **"curva de aprendizaje"**. La curva de aprendizaje es una herramienta que permite evaluar el rendimiento de un modelo comparando el aprendizaje (eje y) respecto a la experiencia (eje x). Permite observar como el rendimiento del modelo mejora a medida que se aumenta la experiencia y permite determinar si un modelo se beneficia de dicho incremento y la eficacia del algoritmo usado.

La métrica utilizada para evaluar el **aprendizaje (eje y)** puede ser la maximización o la minimización en función de la variable que se analice. Un ejemplo de maximización podría ser evaluar la precisión de la clasificación, lo que significa que mejores puntuaciones (valores más altos) indican más aprendizaje. Mientras que la minimización, podría ser la pérdida o error, lo que significa que mejores puntuaciones (valores más pequeños) indican mayor aprendizaje. Es más habitual utilizar minimización.

La variable utilizada para evaluar la **experiencia (eje x)** puede ser el tiempo o experiencia representado en número de iteraciones de un algoritmo o tamaño del conjunto de entrenamiento. A mayores iteraciones más precisión o menor error, al igual que a mayor conjunto de entrenamiento, más muestras para aprender, lo que conlleva mejores resultados.

Evaluar las curvas de aprendizaje de entrenamiento y test nos puede llevar a identificar que modelos se adaptan mejor a la problemática que queremos solucionar así como guiarnos en que paso dar para llegar a un modelo eficiente.

Se puede identificar un modelo **sobre-generalizado** cuando:

- La curva de aprendizaje de entrenamiento permanece plana independientemente del entrenamiento. Esto podría deberse a una **elección de características incorrectas** que no tengan correlación con la salida, ya que no permitiría al modelo predecir bien la salida en base a los datos de entrada dados o un **algoritmo poco eficiente**, que no tiene la capacidad suficiente para aprender, dada la complejidad del conjunto de datos.

- La curva de aprendizaje de entrenamiento continúa mejorando con la experiencia. Esto podría deberse a que el **proceso de aprendizaje se detuvo prematuramente**, indica que el modelo es capaz de aprender más.

Se puede identificar un modelo **sobre-ajustado** cuando:

- La curva de aprendizaje de entrenamiento continúa mejorando con la experiencia.

- La curva de aprendizaje de test mejora hasta un punto y comienza a empeorar nuevamente. Esto podría deberse a un **algoritmo demasiado eficiente**, que tiene más capacidad y flexibilidad de la necesaria para el problema dado o el **proceso de aprendizaje dura demasiado tiempo**.

Se puede identificar un modelo **bien ajustado** cuando las curvas de aprendizaje de entrenamiento y test mejoran hasta un punto de estabilidad, existiendo entre ellos una "pequeña brecha".

Se puede identificar una **descompensación en la elección de datos de entrenamiento y test** cuando:

- Las curvas de aprendizaje entrenamiento y test mejoran hasta un punto de estabilidad, existiendo entre ellos una gran brecha. Esto puede deberse a la existencia de **pocos datos de entrenamiento** frente a los datos de test (conjunto de entrenamiento no representativo del problema) o a una **elección de características**

**o modelo incorrecto** que no permita generalizar y representar bien las predicciones ante nuevas observaciones.

– La curva de aprendizaje de entrenamiento mejora, mientras que la de test muestra ruido alrededor de la de entrenamiento o incluso tiene mayor mejora que la de entrenamiento. Esto puede deberse a la existencia de **pocos datos de test** frente a los de entrenamiento (conjunto de test no representativo) o **datos de test demasiado fáciles de predecir** para el modelo (conjunto de test poco complejo).

# Capítulo 3

# Diseño y desarrollo

## 3.1. Planteamiento del proyecto

Para la implementación del proyecto se plantea la división en diferentes módulos:

- Módulo de aprendizaje del **contaje de dedos o análisis estático** de la mano.
- Módulo de **análisis del movimiento o análisis dinámico** realizado por la mano.
- Módulo de **integración** de los previos módulos con la problemática de la **separación del movimiento**.

Se plantean las siguientes tareas para cada uno de los módulos de entrenamiento:

- **Elección del conjunto de vídeos:** número de vídeos, calidad de los vídeos, variabilidad en la iluminación, posturas, movimiento y usuarios.
- **Pre-procesamiento y análisis de los vídeos:** eliminación de vídeos malos o poco representativos, tratamiento de los vídeos con el objetivo de obtener mejoras en el cálculo de los datos, así como la elaboración de distintas colecciones en base a este análisis.
- **Elección de características:** parámetros que se extraerán de los vídeos que resulten representativos para la clasificación así como generalizables.
- **Pre-procesamiento y análisis de los datos:** esto se basa en un análisis exhaustivo de los datos que incluye la eliminación de datos faltantes, eliminación de ruido y muestras atípicas de los datos.
- **Entrenamiento y evaluación de modelos:** planteamiento de como seleccionar el "mejor modelo" para cada módulo así como las métricas usadas para validación.

– **Selección de los modelos a entrenar:** selección de los métodos de balanceo de datos (reducción o ampliación del conjunto de muestras, transformación vertical), métodos de selección de características (reducción de la dimensionalidad o reducción horizontal), normalización de los datos y estimadores.

Para una mejor implementación del proyecto, visionado y manipulación de los vídeos, así como mantenibilidad a futuro, se plantea la implementación de una interfaz en *Tkinter* [43] que apoye y permita la evaluación de diferentes estrategias de análisis de datos y entrenamiento y mejorar de esta forma la toma de decisiones durante el proyecto. En el anexo C se describe la planificación inicial del proyecto basado en módulos.

## 3.2. Hardware, herramientas y modelos utilizados

El hardware utilizado durante todo el proyecto ha sido una Raspberry Pi 4B junto a una cámara Pi Camera v2 NoIR. Las especificaciones de la Raspberry Pi son [44]:

– Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @1.8GHz.

– 1 GB, 2 GB, 4 GB o 8 GB LPDDR4-3200 SDRAM.

– 2.4 GHz y 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE Gigabit Ethernet.

– 2 puertos USB 3.0; 2 puertos USB 2.0.

– Raspberry Pi estándar 40 pin GPIO header.

– 2 puertos micro-HDMI® (hasta 4kp60).

– 2-lane MIPI DSI display port.

– 2-lane MIPI CSI camera port.

– 4-polos estéreo audio y puerto de vídeo compuesto.

– H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode).

– OpenGL ES 3.1, Vulkan 1.0.

– Slot para tarjeta micro-SD.

– 5V DC via conector USB-C (mínimo 3A).

– 5V DC via GPIO header (mínimo 3A).

– Alimentación a través de Ethernet (PoE).

– Temperatura de operación: 0 – 50 grados C.

Las especificaciones de la Raspberry Pi Camera Module 2 NoIR son [45]:

– Sensor Sony IMX219 de 8-megapixeles.

– No lleva filtra infrarrojo (IR).

La razón principal en escoger esta conjunto de hardware es porque ofrece una excelente flexibilidad para probar diferentes configuraciones tanto a nivel hardware como a nivel software. Además tiene soporte para usar `Python` y existe un foro de usuarios muy amplio en caso de necesitar soporte. La figura 3.1 muestra fotos de los componentes por separado mientras que la figura 3.2 muestra fotos del prototipo con todos los elementos de la primera versión de este diseño juntos.

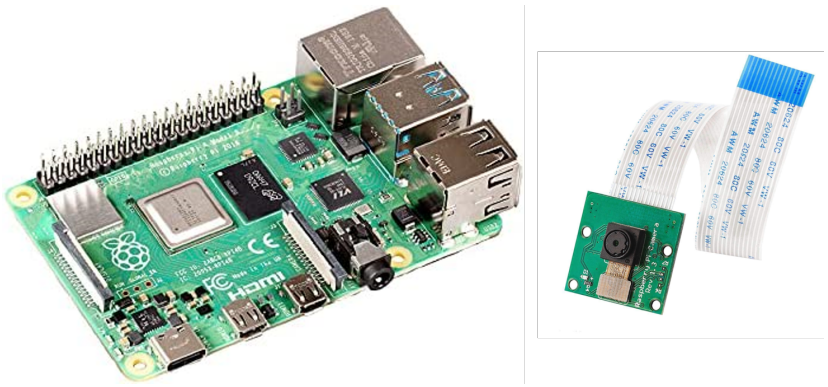


Figura 3.1: Raspberry Pi 4b (izquierda) y Pi Camera NoIR v2 (derecha). Este conjunto ofrece una excelente flexibilidad para prototipos tanto a nivel hardware como software con una amplia base de usuarios.

El software usado para la detección de los puntos de la mano es `Mediapipe`.

## 3.3. Elección del conjunto de vídeos

Todas las colecciones de vídeos creadas tienen algunas características en común:

– **Postura o número de dedos:** se han realizado vídeos únicamente de una configuración de dedos por cada posibilidad de número de dedos (de 1 a 5). Las configuraciones han sido las siguientes:

  • Un dedo: índice.

  • Dos dedos: índice y corazón.

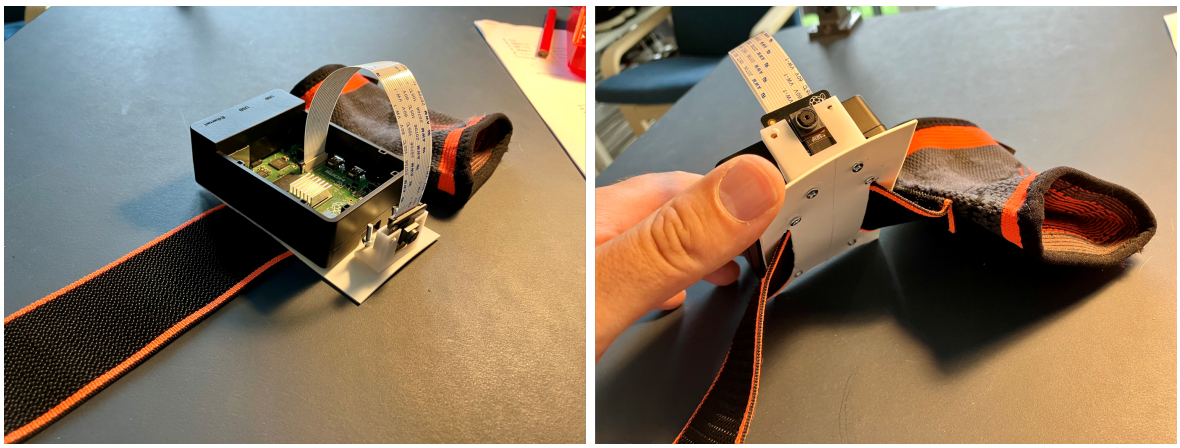


Figura 3.2: Primera versión del prototipo en la que la Raspberry Pi está montada en un soporte plástico que se adapta al brazo y la cámara montada en un soporte que apunta hacía la mano. Todo el conjunto va montado sobre una brazalete elástico que permite ajustar el conjunto alrededor del antebrazo.

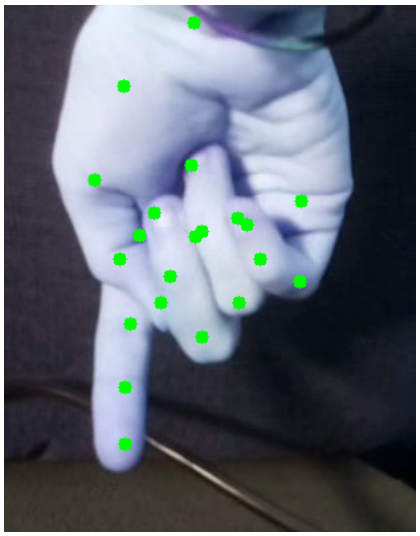


Figura 3.3: Postura: 1 dedo.

- Tres dedos: corazón, anular y meñique.
- Cuatro dedos: índice, corazón, anular y meñique.
- Cinco dedos: pulgar, índice, corazón, anular y meñique.

Se han decidido estas configuraciones y no otras por comodidad. Por otro lado, solo se ha decidido realizar una de cada tipo porque realizar vídeos por cada configuración de dedos habría sido mucho más costoso en tiempo, además de que con una resulta suficiente como para obtener ejemplos de cada uno de los dedos en sus dos formas "estirada" y "no estirada".

– **Gesto o movimiento de la mano:** se han realizado vídeos de cada movimiento posible según la lógica explicada en la sección 2.1.3. La postura inicial de la mano

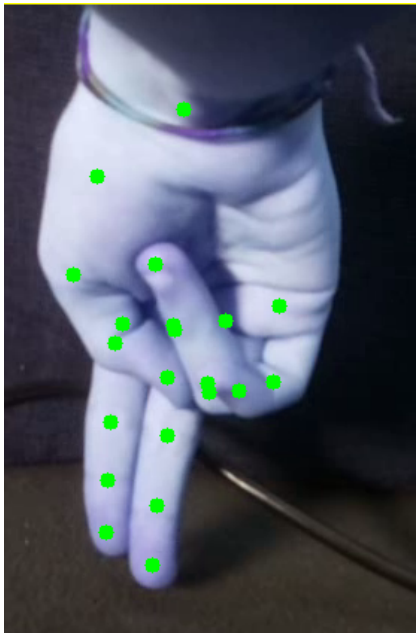


Figura 3.4: Postura: 2 dedos.

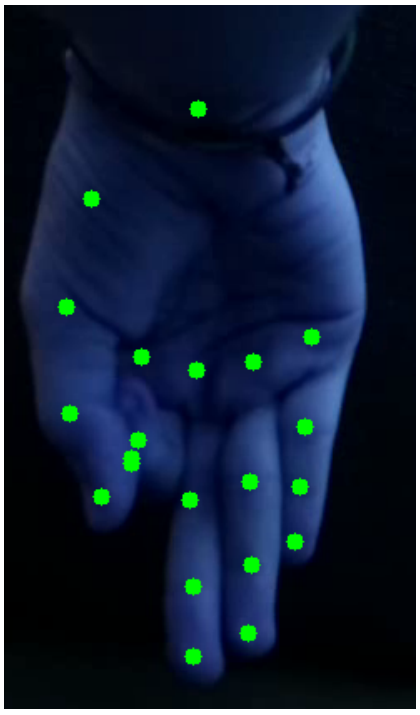


Figura 3.5: Postura: 3 dedos.

consiste en la mano estirada con la configuración de dedos elegida explicada en el punto anterior y con la palma mirando hacia la cámara. Los movimientos posibles son los siguientes:

- **Derecha:** el movimiento consiste en una rotación de la muñeca hacia la izquierda de forma que los dedos acaben apuntando hacia el lado derecho.

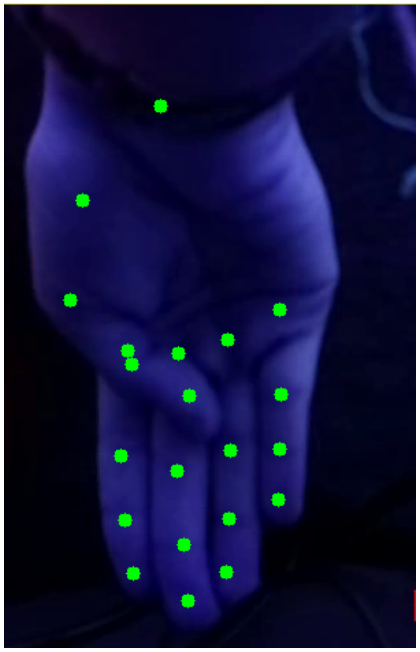


Figura 3.6: Postura: 4 dedos.

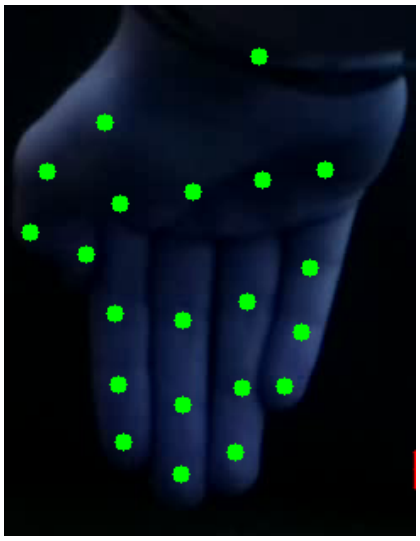


Figura 3.7: Postura: 5 dedos.

- **Izquierda:** el movimiento consiste en una rotación de la muñeca hacia la derecha de forma que los dedos acaben apuntando hacia el lado izquierdo.
- **Arriba:** el movimiento consiste en elevar la mano llevándola de la postura inicial hacia atrás de forma que los dedos acaben apuntando hacia el "fondo".
- **Abajo:** el movimiento consiste en llevar la mano hacia adelante como si se intentase tocar la muñeca con los dedos, sin llegar a cerrar la mano.
- **Centro:** el movimiento consiste en elevar la mano llevándola de la postura inicial hacia arriba de forma que los dedos sigan apuntando hacia abajo.

– **Número de vídeos:** se ha decidido grabar como mínimo 40 vídeos por cada "letra", esto es por cada par postura (número de dedos), gesto (movimiento) asociado. Es decir, cada colección se compone de como mínimo 5 * 5 * 40 = 1000 vídeos.

## 3.4. Pre-procesamiento y análisis de los vídeos

Tras obtener las distintas colecciones, se ha usado el software de `Mediapipe` para obtener las coordenadas en la imagen de los 21 puntos de la mano. Tras obtener todos estos datos y con el objetivo de asegurar que los vídeos han sido grabados correctamente y que la mano ha sido detectada por el software de `Mediapipe`, se han visualizado dos parámetros en cada vídeo:

– **Mediapipe percent:** porcentaje que representa el número de frames en los que `Mediapipe` ha detectado una mano del total de frames.

– **Mediapipe score:** media de los scores dados por el software de `Mediapipe` en cada frame que representa "como de bien" ha predicho estos 21 puntos de la mano durante el transcurso del vídeo.

En base a estos dos datos, se han ido grabando las colecciones de vídeos y verificando que las manos son detectadas y sus scores son superiores al 80 %. Esto ha permitido borrar vídeos mal grabados o poco representativos y volver a grabar otros con el objetivo de obtener un buen conjunto de datos. Además, se ha decidido grabar distintas colecciones de vídeos con diferentes iluminaciones y forma de empezar el movimiento para ver como afecta a los resultados y si estos dos parámetros se ven influenciados por ello. En las siguientes secciones se explica de forma más detallada las diferentes colecciones elaboradas durante el desarrollo del proyecto, así como los resultados.

## 3.5. Elección de características

Para entrenar los modelos se han calculado una serie de características que se han considerado representativa para implementar cada uno de los módulos de clasificación: **clasificación de la postura**, o análisis estático, y **clasificación del movimiento**, o análisis dinámico. No se han usado los datos en crudo proporcionados por `Mediapipe` sino una transformación de estos debido a que ha sido más conveniente que el modelo aprenda sobre medidas relativas a la postura de la mano en sí y no relativas a la imagen. Las características extraídas para cada módulo se explican en las siguientes secciones.

### 3.5.1. Características para la clasificación de la postura

Para la clasificación de la postura, se han usado los ángulos y distancias que permiten describir bien la postura de la mano. Como primera aproximación se podría haber calculado todos los ángulos formados por los vectores de la mano y todas las distancias entre los puntos. Este camino generaba demasiados datos y, por ende, conllevaría una dimensionalidad muy alta en el modelo y la base de datos, lo cuál requería realizar un análisis en profundidad de un número elevado de variables con el objetivo de seleccionar las características más relevantes para el modelo. Esto llevaría mucho más tiempo que analizar solo un subconjunto de estas, por ello, finalmente, se han extraído solo algunos ángulos y distancias, las cuáles han parecido más representativas para distinguir si un dedo esta estirado o no. Cabe destacar que la forma de estirar los dedos para el pulgar es diferente que para el resto de dedos, por lo que se han extraído algunas variables adicionales las cuáles a priori parecen más útiles en este caso concreto. Las características extraídas se describen a continuación, incluyendo una descripción de cada una de ellas, fórmulas y motivación para ser incluidas en el dataset.

#### Ángulo de la flexión del dedo

Este ángulo es el correspondiente a la variable *flexionDedo_False*. Este atributo tiene las siguientes características:

- **Descripción:** describe el ángulo del dedo formado por la base, el nudillo (punto de flexión del dedo) y la yema. Se mide en radianes.
- **Fórmula:** para calcularlo se ha utilizado la siguiente fórmula que calcula el ángulo formado por dos vectores:

$$angle(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{|\vec{u}| \cdot |\vec{v}|} \tag{3.1}$$

  siendo $\vec{u}, \vec{v}$ los dos vectores y $|\vec{u}|, |\vec{v}|$ sus magnitudes. El rango va de $(0, \pi)$ radianes. La fórmula es la que sigue:

$$flexionDedo_{False} = angle\left(\vec{finger}FtoP, \vec{finger}FtoB\right) \tag{3.2}$$

  siendo $\vec{finger}FtoP$ el vector de la flexión a la yema (*flexion-to-print*), y $\vec{finger}FtoB$ el vector de la flexión a la base (*flexion-to-base*).

- **Motivación:** se ha seleccionado esta variable debido a que la principal diferencia entre un dedo estirado y no estirado es este ángulo. Al estar totalmente estirado

podríamos decir que este ángulo es de $180^{o}$, mientras que al estar totalmente cerrado sería de $90^{o}$ en el caso del pulgar y de $45^{o}$ en el resto de los dedos.

– **Rango esperado:** Pulgar de $(\frac{\pi}{2}, \frac{3\pi}{4})$ no estirado, $(\frac{3\pi}{4}, \pi)$ estirado, resto de dedos de $(\frac{\pi}{4}, \frac{\pi}{2})$ no estirado, $(\frac{\pi}{2}, \pi)$ estirado.

### Ángulo de la flexión del dedo en el plano XY

Este ángulo se corresponde a las variables *flexionDedo_True_False* y *flexionDedo_True_True*, representan el mismo ángulo pero con diferente rango de representación. Estos atributos tienen las siguientes características:

– **Descripción:** describe el ángulo del dedo formado por la base, el nudillo (punto de flexión del dedo) y la yema en el plano XY, tomando como referencia el vector de la flexión a la base. Se mide en radianes.

– **Fórmula:** para calcularlo primero se ha hecho un cambio de base del vector de la flexión a la yema $\vec{fingerFtoP}$ en la referencia del vector de la flexión a la base $\vec{fingerFtoB}$ en el plano XY. Después se ha calculado la arco-tangente. Las fórmulas son las que siguen:

$$flexionDedo_{True,False} = arctan(\frac{\vec{fingerFtoP}_{fingerFtoBx}}{\vec{fingerFtoP}_{fingerFtoBy}}) \tag{3.3}$$

donde el rango va de $(\frac{-\pi}{2}, \frac{\pi}{2})$ radianes.

$$flexionDedo_{True,True} = \begin{cases} flexionDedo_{True,False}, & \text{if } \vec{fingerFtoP}_{fingerFtoBx} > 0, \\ & \vec{fingerFtoP}_{fingerFtoBy} > 0 \\ flexionDedo_{True,False} + \pi, & \text{if } \vec{fingerFtoP}_{fingerFtoBy} < 0 \\ flexionDedo_{True,False} + 2\pi, & \text{if } \vec{fingerFtoP}_{fingerFtoBx} < 0 \end{cases} \tag{3.4}$$

donde el rango va de $(0, 2\pi)$ radianes, siendo $\vec{fingerFtoP}_{fingerFtoB}$ el vector de la flexión a la base (*flexion-to-base*) en la referencia del vector de la flexión a la yema (*flexion-to-print*).

– **Motivación:** se ha seleccionado esta variable debido a que nos permite conocer el arco que forma la yema del dedo en el plano XY respecto a la flexión y base del dedo. En *flexionDedo_True_False* nos da el arco barrido respecto a este vector fingerFtoB siendo en el 1er y 3er cuadrante un ángulo comprendido en $(\frac{-\pi}{2}, 0)$

y en el 2º y 4º cuadrante un ángulo de $(0, \frac{\pi}{2})$ avanzando de izquierda a derecha del vector y en *flexionDedo_True_True* el arco de izquierda a derecha del vector de 0º a 360º. La variable *flexionDedo_True_False* por si sola probablemente no nos da una forma de distinguir si el dedo está estirado o no, pero complementada con otras variables como la explicada en el punto anterior permite saber la orientación de los dedos (cuadrante) en el plano XY. Por otro lado, la variable *flexionDedo_True_True* si nos da información por si sola de si el dedo está estirado o no, presentando un dedo no estirado un ángulo entre $(0, \frac{\pi}{2})$ o $(\frac{3\pi}{2}, 2\pi)$ y un dedo estirado un ángulo entre $(\frac{\pi}{2}, \frac{3\pi}{2})$.

- **Rango esperado variable *flexionDedo_True_False*:** $(\frac{-\pi}{2}, \frac{\pi}{2})$ estirado y no estirado.
- **Rango esperado variable *flexionDedo_True_True*:** pulgar de $(\frac{\pi}{2}, \pi)$ estirado, $(\pi, \frac{3\pi}{2})$ no estirado, resto de dedos de $(0, \frac{\pi}{2})$ o $(\frac{3\pi}{2}, 2\pi)$ no estirado, $(\frac{\pi}{2}, \frac{3\pi}{2})$ estirado.

### Ángulos entre el dedo y el dedo índice

Estos ángulos son los correspondientes a las variables:

- *dedo_dedoRef_False*
- *dedo_dedoRef_True_False*
- *dedo_dedoRef_True_True*
- *dedoFtoP_dedoRef_False*
- *dedoFtoP_dedoRef_True_False*
- *dedoFtoP_dedoRef_True_True*
- *dedoF2toP_dedoRef_False*
- *dedoF2toP_dedoRef_True_False*
- *dedoF2toP_dedoRef_True_True*

Estos atributos tienen las siguientes características:

- **Descripción:** describe el ángulo del dedo que forma respecto a un dedo de referencia, en este caso, el índice. Se mide en radianes.

- **Dedo de referencia:** se usa el índice dado por el vector que forma el dedo de la base a la yema.
- **Variables *dedo_dedoRef*:** usan el vector que forma el dedo de la base a la yema.
- **Variables *dedoFtoP_dedoRef*:** usan el vector que forma el dedo de la flexión (nudillo) a la yema.
- **Variables *dedoF2toP_dedoRef*:** usan el vector que forma el dedo de la otra flexión del dedo a la yema.
- **Fórmulas:** análogas a las fórmulas de los puntos anteriores siendo el vector $\vec{u}$ el dedo y $\vec{v}$ el dedo de referencia (índice).
- **Motivación:** se ha seleccionado esta variable por el dedo pulgar. La forma en la que se recoge o extiende este dedo es una forma diferente al resto, recogiéndose hacia la palma de la mano siguiendo el eje horizontal, mientras que el resto se recogen siguiendo el eje vertical. Por ello, estas variables nos pueden ayudar a distinguir si está estirado o no ya que proporcionan el ángulo que forma con el índice usando diferentes vectores del dedo y en distintos rangos.

### Distancias al centroide

Estas distancias son las correspondientes a las variables *centroid_d1* y *centroid_d2*. Estos atributos tienen las siguientes características:

- **Descripción:** distancia de la yema al centroide en el caso de *centroid_d1* y de la flexión (nudillo) al centroide en caso de *centroid_d2*.
- **Fórmula:** para calcular esta distancia primero se ha calculado el centroide de la mano. Se ha calculado de la siguiente forma:

$$centroid = \frac{1}{N}\sum_{i=1}^{N} palmPoints_i \tag{3.5}$$

siendo $palmPoints = [p_0, p_1, p_2, p_5, p_9, p_{13}, p_{17}]$ los puntos de la palma, y $N$ el número de puntos de la palma. Las fórmulas son las que siguen:

$$centroid_{d1} = |fingerPrint - centroid| \tag{3.6}$$

$$centroid_{d2} = |fingerFlexion - centroid| \tag{3.7}$$

- **Motivación:** estas distancias cambian al recoger y extender los dedos, especialmente la de la yema del dedo al centroide, ya que el movimiento de recoger el dedo lo lleva hacia la palma de la mano reduciendo esta distancia.

- **Rango esperado:** (0, 1) siendo los valores más cercanos a 0 los que configuran un dedo no estirado y los valores mayores los que indican que está estirado. Esta distancia puede variar de la distancia de la mano a la cámara, así como del tamaño de la mano del usuario.

#### Distancias a la base

Estas distancias son las correspondientes a las variables *fingerBase_d1* y *fingerBase_d2*. Estos atributos tienen las siguientes características:

- **Descripción:** distancia de la yema a la base del dedo en el caso de *centroid_d1* y de la flexión (nudillo) a la base del dedo en caso de *centroid_d2*.

- **Fórmula:** análoga a la del punto anterior pero usando la base del dedo en vez de el centroide.

- **Motivación:** análoga a la del apartado anterior, podría proporcionar especial mejora para el dedo pulgar. Debido a la forma en la que se recoge o expande el dedo pulgar parece más conveniente esta variable que la anterior para este caso.

#### Distancias en el eje x respecto a un dedo de referencia

Estas distancias son las correspondientes a las variables *d1_difx* y *d1_xRespectoRef*. Estos atributos tienen las siguientes características:

- **Descripción:** distancia de la yema a la base del dedo de referencia (índice) en el eje x.

- **Fórmula:** las fórmulas son las siguientes:

$$d1_{difx} = fingerPrint_x - fingerRefBase_x \tag{3.8}$$

$$d1_{xRespectoRef} = fingerPrint_{fingerR\vec{e}fBtoPx} - fingerRefBase_{fingerR\vec{e}fBtoPx} \tag{3.9}$$

siendo $fingerPrint_{fingerR\vec{e}fBtoP}$ la yema del dedo en la referencia del vector formado por la base a la yema del dedo de referencia, el índice, tomando como origen

su base, y $fingerRefBase_{finger\vec{R}efBtoP}$ la base del dedo en la referencia del vector formado por la base a la yema del dedo de referencia, el índice, tomando como origen su base.

– **Motivación:** la distancia en x de la yema al índice puede ser una variable útil para distinguir si el pulgar está estirado o no, ya que, al recogerlo, la yema va hacia la palma acercándose a la base del índice y pudiendo sobrepasar su límite formado por su vector de la base a la yema $finger\vec{R}efBtoP$.

**Distancias en el eje y respecto a su vector** $finger\vec{F}toB$

Esta distancia es la correspondiente a la variable *y_zRespectoFB_d1*. Este atributo tiene las siguientes características:

– **Descripción:** distancia en el eje y de la yema a la base del dedo en la referencia del vector formado por la flexión a la base del dedo, tomando como origen la flexión.

– **Fórmula:**

$$y_{zRespectoFB,d1} = fingerPrint_{finger\vec{F}toBy} - fingerBase_{finger\vec{F}toBy} \tag{3.10}$$

– **Motivación:** la distancia en y de la yema a la base es una variable que puede ayudar a distinguir si el dedo está estirado o no.

### 3.5.2. Características para la clasificación del movimiento

De forma análoga al apartado anterior, se han extraído algunas características que a priori parecen relevantes para distinguir el movimiento. Estas características son descritas, de la misma forma que en el apartado anterior, en los siguientes puntos de esta sección. Se han calculado dos subconjuntos de características que se diferencian en el hecho de conocer a priori o no el número de dedos involucrados en el movimiento. Se han generado estos dos subconjuntos para ver si el hecho de calcular los datos conociendo el número de dedos afecta significativamente en los resultados. En caso de que se obtenga una clara mejora conociendo el número de dedos, se planteará ejecutar el módulo de movimiento tras obtener el resultado del módulo de contaje del número de dedos, lo cuál conllevará a peores resultados en tiempo, pero mejores resultados de predicción del movimiento. Si esto no afecta significativamente, se podrían realizar las predicciones de estos dos módulos de forma paralela lo cuál proporcionará al usuario una respuesta de forma más rápida. El análisis de estos dos subconjuntos se realizará en las siguientes secciones y se tomará una decisión en base a ellos.

**Dirección del movimiento**

Estas direcciones son las correspondientes a:

- *dirGenx*
- *dirGeny*
- *dirGenz*
- *dirDedx*
- *dirDedy*
- *dirDedz*

Estos atributos tienen las siguientes características:

- **Descripción:** media de los vectores que describen la dirección de avance de los dedos de la mano en todos sus puntos. En el caso de *dirGen* se usan todos los puntos y en caso de *dirDed* se usan solo los puntos de los dedos involucrados en el movimiento (los estirados). En ambos casos son 4 puntos por cada dedo, es decir, 20 puntos totales de la mano.
- **Fórmula:**

$$dirGen = \frac{1}{20}\sum_{i=1}^{20}(landmarksActual_i - landmarksAnterior_i) \tag{3.11}$$

$$dirDed = \frac{1}{N}\sum_{i=1}^{N}(landmarksActualEstirados_i - landmarksAnteriorEstirados_i) \tag{3.12}$$

- **Motivación:** el vector que describe la dirección de avance es una variable relevante para el movimiento ya que describe hacia donde se dirige el movimiento en cada eje.

**Arco de la dirección del movimiento**

Estas direcciones son las correspondientes a:

- *orientDirGenXY*
- *orientDirGenZY*

- *orientDirXY*
- *orientDirZY*

Estos atributos tienen las siguientes características:

- **Descripción:** arco de la media de las direcciones de avance (variables del apartado anterior) en el plano XY y plano ZY. *orientDirGen* usa datos de todos los dedos, mientras que *orientDir* usa solo los de los dedos involucrados.
- **Fórmula:**

$$orientDirGenXY = \arctan\left(\frac{dirGen_x}{dirGen_y}\right) \tag{3.13}$$

$$orientDirGenZY = \arctan\left(\frac{dirGen_z}{dirGen_y}\right) \tag{3.14}$$

$$orientDirXY = \arctan\left(\frac{dirDed_x}{dirDed_y}\right) \tag{3.15}$$

$$orientDirZY = \arctan\left(\frac{dirDed_z}{dirDed_y}\right) \tag{3.16}$$

- **Motivación:** al igual que los parámetros anteriores, estos parámetros describen características sobre la dirección de movimiento. En este caso describe el arco descrito por la dirección en cada plano.

**Arco de la dirección de los dedos**

Estas direcciones son las correspondientes a:

- *orientDedGenXY*
- *orientDedGenZY*
- *orientDedXY*
- *orientDedZY*

Estos atributos tienen las siguientes características:

– **Descripción:** arco de la media de las direcciones de los dedos en el plano XY y plano ZY. *orientDedGen* usa datos de la dirección de la base a la flexión de todos los dedos, mientras que *orientDed* usa la dirección de la flexión a la yema de solo los dedos involucrados. En caso de usar los datos de todos los dedos parece más conveniente usar la dirección de la base a la flexión ya que no varía el sentido del vector estando los dedos estirados o no, mientras que si conocemos el dato de los dedos estirados podemos usar la dirección de la flexión a la yema de solo esos dedos lo cuál podría aportarnos más información.

– **Fórmula:**

$$orientDedGenXY = \arctan\left(\frac{dirBF_x}{dirBF_y}\right) \tag{3.17}$$

$$orientDedGenZY = \arctan\left(\frac{dirBF_z}{dirBF_y}\right) \tag{3.18}$$

$$orientDedXY = \arctan\left(\frac{dirFP_x}{dirFP_y}\right) \tag{3.19}$$

$$orientDedZY = \arctan\left(\frac{dirFP_z}{dirFP_y}\right) \tag{3.20}$$

– **Motivación:** al igual que los parámetros anteriores, estos parámetros describen características sobre la dirección de movimiento. En este caso describe el arco descrito por la dirección en cada plano.

Cada vídeo de las colecciones está compuesto por una serie de frames, el cuál es variable en función de la velocidad a la que se ha realizado el movimiento. Para desacoplar el tiempo del análisis del movimiento y reducir la dimensionalidad de la entrada de datos al modelo, se plantea transformar todos estos datos $x\frac{variables}{frame} * numframes$ en $xvariables$ realizando por cada intervalo de frames y característica transformaciones como suma de los datos en diferentes ejes, media de los datos en el intervalo, etc. según se requiera. Esto se explicará más adelante en posteriores secciones tras realizar un análisis de los datos de movimiento.

## 3.6. Diseño de la arquitectura

### 3.6.1. Diseño de la base de datos

La figura 3.8 muestra el diagrama entidad-relación de la base de datos mientras que la figura 3.9 muestra el diagrama relacional. En el anexo B se describen los diferentes

campos y sus atributos.

### 3.6.2. Diseño de la GUI en Tkinter

No se adjunta pero se ha realizado una interfaz para la elaboración de las pruebas.

## 3.7. Colecciones de vídeos

El primer paso tras la elaboración del prototipo hardware fue realizar las colecciones de vídeos. Debido a la cámara utilizada, al grabar las colecciones de vídeos, la pulsera tuvo que ser colocada casi en el codo para que la cámara pudiera obtener imágenes de la mano completa. Las colecciones obtenidas fueron las siguientes:

- **Colección 1 o *C-Rasp-First*:** esta colección fue grabada sin iluminación directa a la mano y los movimientos fueron grabados con la postura de la mano (número de dedos) ya fijada al inicio.

- **Colección 2 o *C-Rasp-SWIlum*:** consiste en la colección 1 a la que se le ha aplicado un filtro en la imagen para darle más iluminación.

- **Colección 3 o *C-Rasp-Ilum*:** esta colección fue grabada con iluminación directa a la mano, consistente en un flexo apuntando hacia la mano, y los movimientos fueron grabados con la postura de la mano (número de dedos) sin fijar, es decir, la mano comienza abierta con todos los dedos estirados y el cambio de configuración se realiza al inicio del vídeo.

En la sección 4 se explica que impacto tienen estos cambios sobre la forma de grabar los movimientos en los resultados y por qué se realizaron estas tres colecciones.

## 3.8. Pre-procesamiento y análisis de los datos

El análisis realizado en los datos se basa en los parámetros descriptivos básicos explicados en el anexo A así como en el análisis hecho en la elección de características de la sección anterior 3.5.

### 3.8.1. Análisis de los datos del módulo de contaje de dedos

Las siguientes gráficas corresponden a los diagramas de cajas de las características para el dedo pulgar. Para todas las características y dedos, así como movimiento se ha realizado un análisis. No se incluye aquí en la memoria pero se ponen algunas gráficas elaboradas.

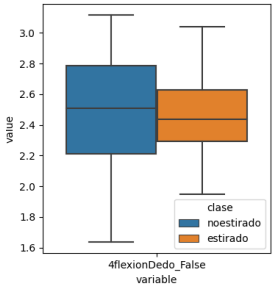



Figura 3.10: Característica 4flexionDedo_False.

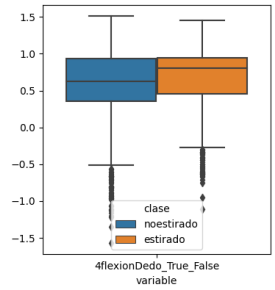



Figura 3.11: Característica 4flexionDedo_True_False.

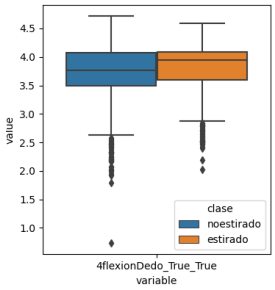



Figura 3.12: Característica 4flexionDedo_True_True.

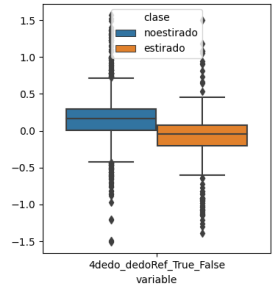



Figura 3.13: Característica 4dedo$_d$*edoRef_True_False*.

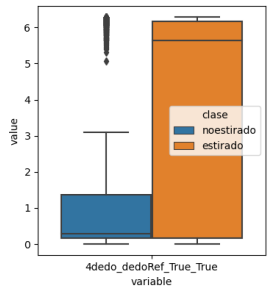



Figura 3.14: Característica 4dedo_dedoRef_True_True.

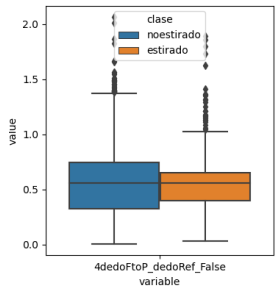



Figura 3.15: Característica 4dedoFtoP_dedoRef_False.

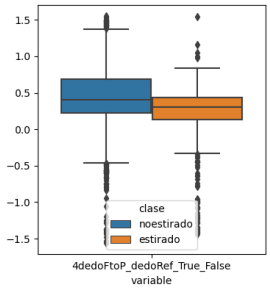



Figura 3.16: Característica 4dedoFtoP_dedoRef_True_False.

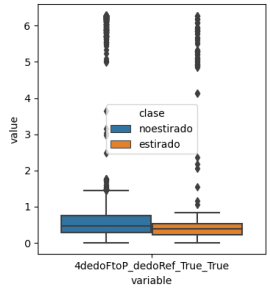



Figura 3.17: Característica 4dedoFtoP_dedoRef_True_True.

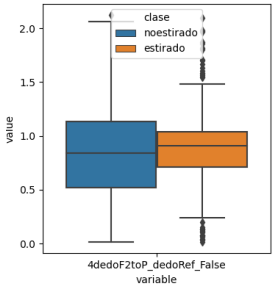



Figura 3.18: Característica 4dedoF2toP_dedoRef_False.

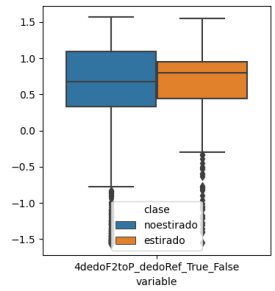



Figura 3.19: Característica 4dedoF2toP_dedoRef_True_False.

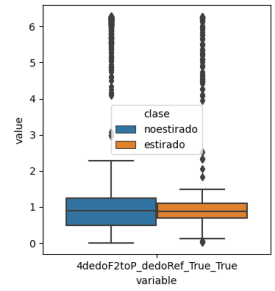



Figura 3.20: Característica 4dedoF2toP_dedoRef_True_True.

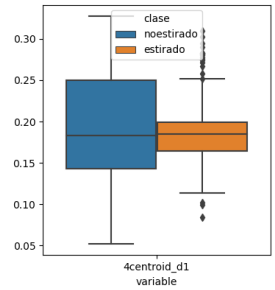



Figura 3.21: Característica 4centroid_d1.

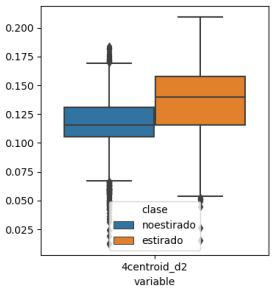



Figura 3.22: Característica 4centroid_d2.

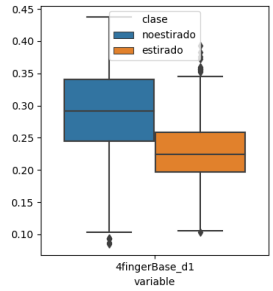



Figura 3.23: Característica 4fingerBase_d1.

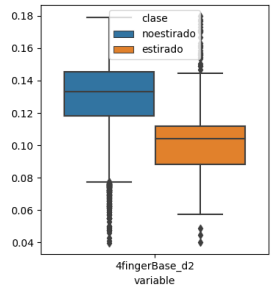



Figura 3.24: Característica 4fingerBase_d2.

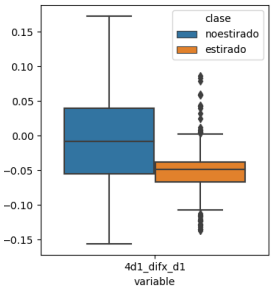



Figura 3.25: Característica 4d1_difx_d1.

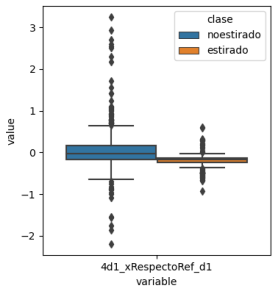



Figura 3.26: Característica 4d1_xRespectoRef_d1.

## 3.9. Entrenamiento y evaluación de modelos

La figura 3.27 muestra un diagrama con el *pipeline* para el entrenamiento y validación de los diferentes modelos.

Ambos módulos, contaje de dedos y movimiento, usan el enfoque planteado en el diagrama de la figura 3.27. Primero, se parte el conjunto de datos en dos conjuntos de proporciones 80 y 20 formando los conjuntos que se usan para entrenamiento y test respectivamente. Posteriormente, se ha decidido usar la técnica *cross-validation* con 5 pliegues sobre el conjunto de entrenamiento para la selección del "mejor modelo" en base a las métricas obtenidas sobre estos conjuntos de validación. Por último, se re-entrena este modelo seleccionado sobre el conjunto de entrenamiento y test como evaluación final. Para realizar ambas separaciones en los conjuntos de datos (entrenamiento-test y *cross-validation*) se ha usado su versión estratificada, que garantiza para todas las separaciones la misma proporción de datos de las clases, y se ha fijado el parámetro *random_state* para que las pruebas sean repetibles y los modelos sean comparados en base a las mismas condiciones.

Se ha decidido usar *cross-validation* para seleccionar el modelo, ya que evaluar diferentes modelos sobre un mismo conjunto de datos (entrenamiento-test) podría originar sobre-ajuste en el conjunto de test. Habría sobre-ajuste, ya que seleccionar el mejor modelo para dicho conjunto de test provocaría que sus datos se "filtrasen" al modelo (fuga de datos), seleccionando el modelo en base a datos no disponibles en un caso real (los datos de test), lo que sobre-estimaría las métricas de evaluación y ya no reflejarían la capacidad de generalización del modelo. Para resolver el problema del sobre-ajuste se podría dividir el conjunto de datos en tres añadiendo un "conjunto de validación", de esta forma, se realizaría la selección del modelo en base a las métricas obtenidas sobre este conjunto de validación y la evaluación final sobre el conjunto de test. Sin embargo, dividirlo en tres provoca tener menos muestras para entrenamiento y el clasificador tendría menos muestras de las que aprender, por ello se ha decidido usar *cross-validation*, donde la separación en entrenamiento y validación se hace dentro del conjunto de entrenamiento original y desperdicia menos datos.

En cuanto al entrenamiento, primero se realizan las particiones y luego se aplican los métodos de balanceo, selección de características, normalización, entrenamiento y, por último, evaluación del modelo. Las particiones se hacen al principio, ya que si se realizase al revés podría haber "fuga de datos". Por ejemplo, si balanceamos con el conjunto entero de datos, al realizar después la partición entrenamiento-test podrían aparecer muestras o información relativa al conjunto de test en el conjunto de entrenamiento, debido a haber balanceado sobre todo el conjunto, lo que llevaría a que el modelo aprenda de información que no debería tener disponible. Del mismo modo para la selección de características, seleccionaría aquellas características que funcionan mejor en relación a todo el conjunto de datos (entrenamiento y test), por lo que usa información del conjunto de test, lo que sobre-estimaría la calidad del modelo. Por otro

lado, el balanceo se aplica a los datos de entrenamiento con el objetivo de representar mejor la clase minoritaria, de modo que el clasificador tenga las mismas muestras de las que aprender de ambas clases y no aprenda una mejor que otra, pero no se aplica a los datos de test. De forma análoga a lo previamente explicado, en un escenario real no dispondríamos de estas etiquetas, necesarias para balancear las clases, ya que es la variable objetivo, la que se desea predecir. Por lo que si balanceásemos los datos de test, estaríamos usando información de la que no dispondríamos en un caso real.

Por último, se ha decidido realizar el balanceo de datos antes de aplicar la selección de características, ya que se ha pensado que realizar la selección de características en base a un conjunto de datos no balanceado seleccionaría las características más relevantes para la clase mayoritaria y tendría menos en cuenta a la clase minoritaria, sobre todo teniendo en cuenta que algunas técnicas de selección de características utilizan como criterio de selección las métricas obtenidas del entrenamiento de un clasificador, los cuáles a veces asumen como entrada la existencia de un conjunto de datos balanceado. Por otro lado, también podría haberse tomado la decisión contraria en base a otros razonamientos. En la literatura, existen algunas publicaciones que hablan sobre en que orden aplicar estas selecciones. En algunos de ellos, denominan la aproximación elegida como IFS (Instance Selection, Features Selection) y la contraria como FIS (Features Selection, Instance Selection).

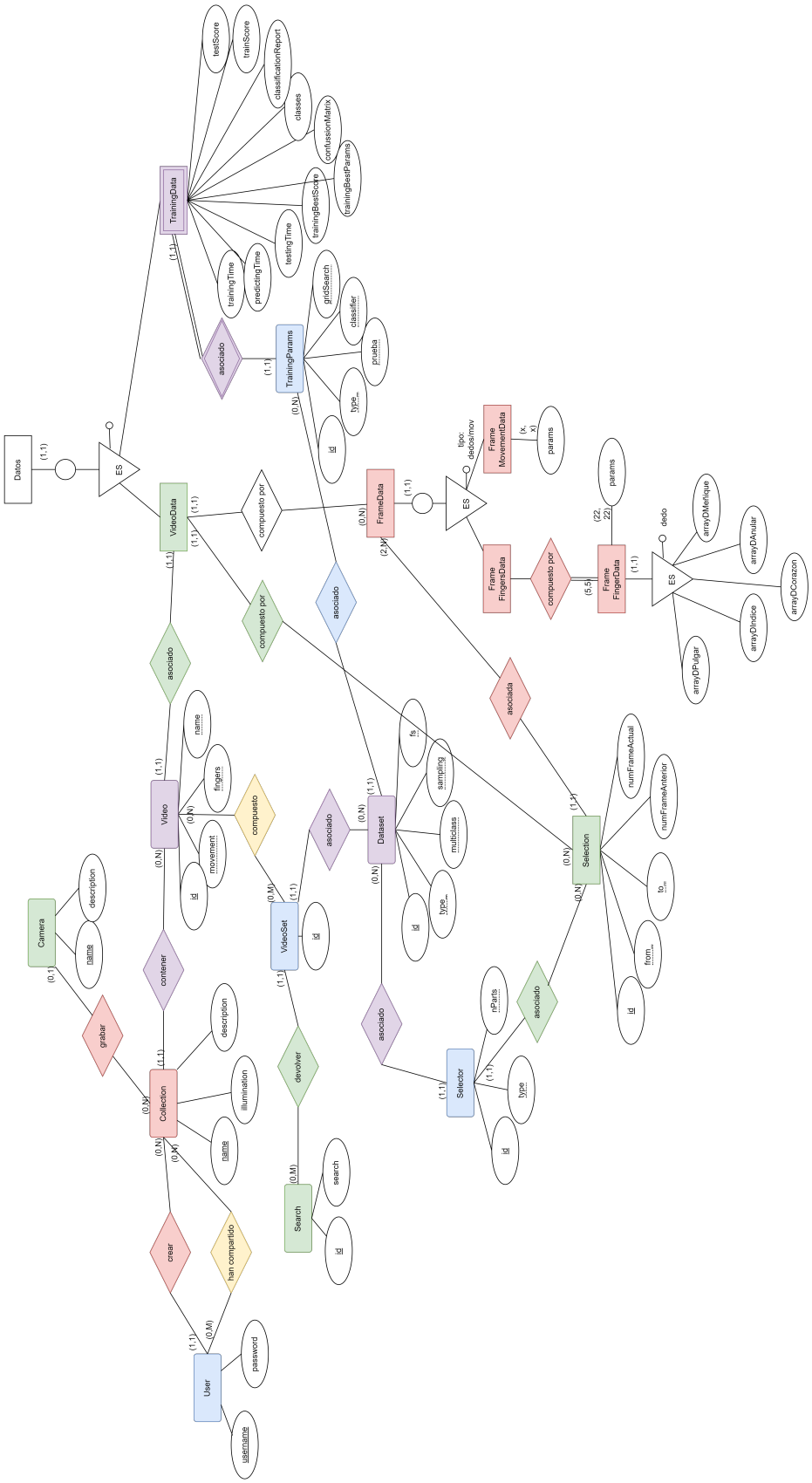



Figura 3.8: Diagrama entidad-relación.

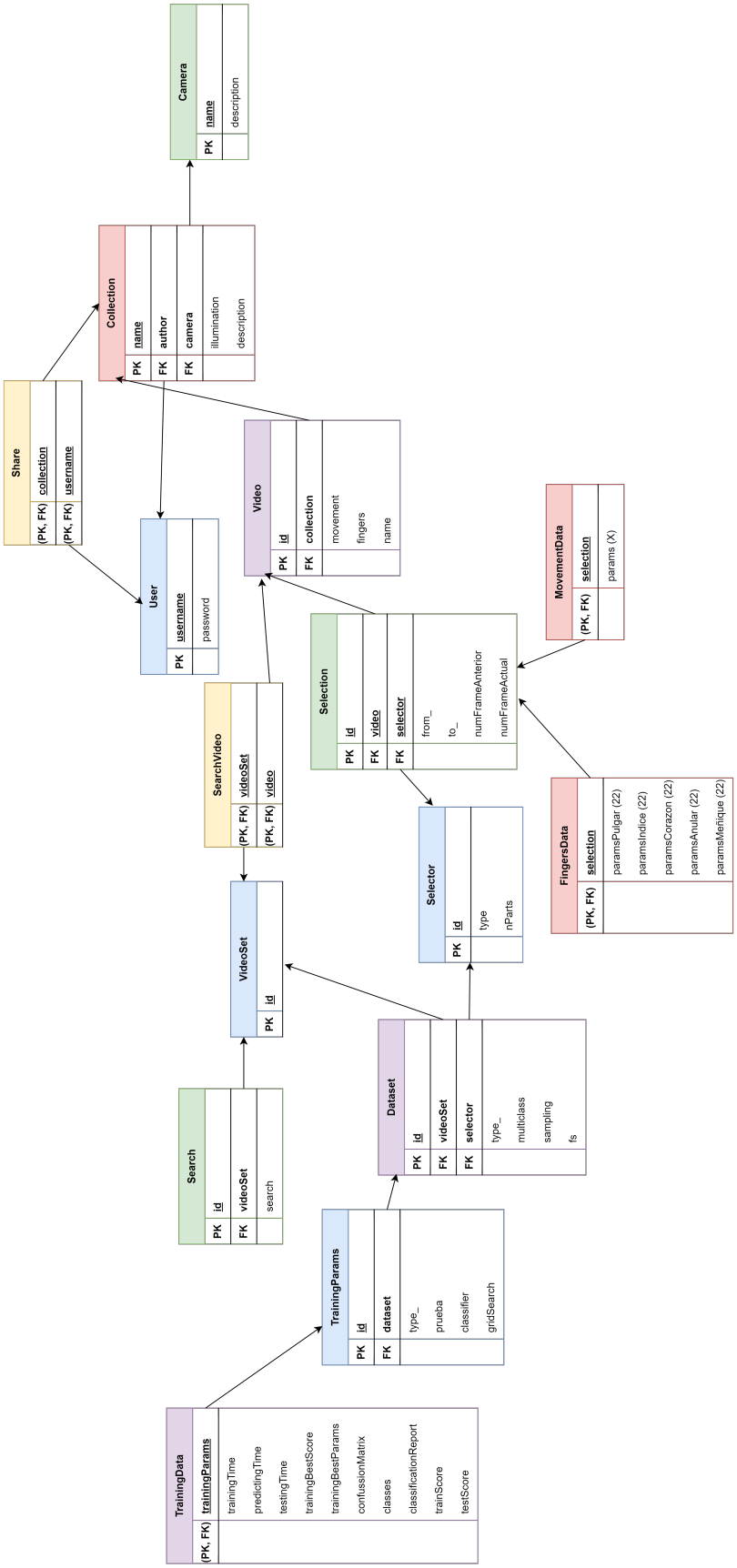



Figura 3.9: Diagrama relacional.

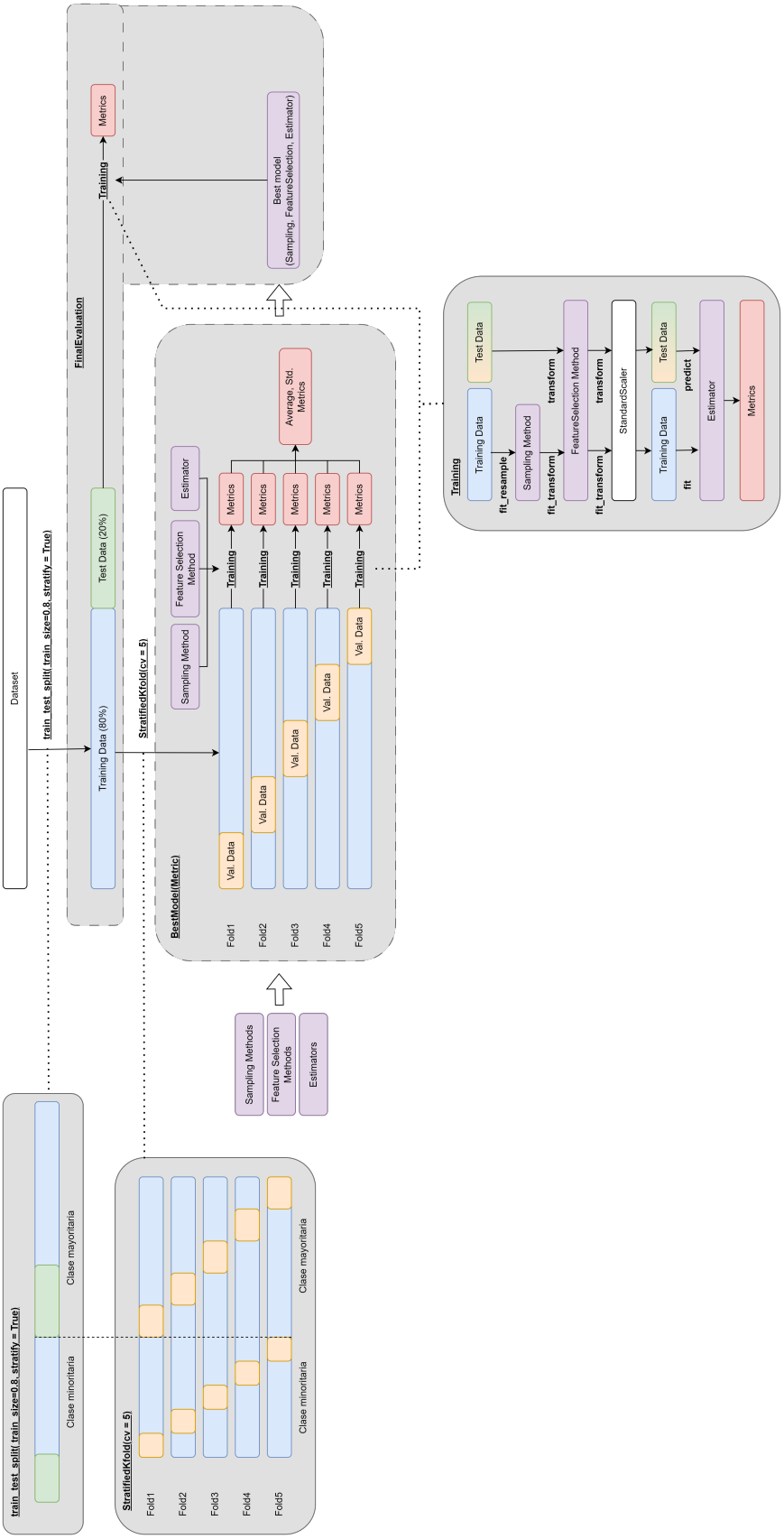



Figura 3.27: Entrenamiento y validación de modelos.

# Capítulo 4

# Pruebas y resultados

## 4.1. Resultados de los modelos de contaje de dedos

Para que el modelo aprenda el número de dedos estirados, independientemente de la configuración de dedos que se realice, se ha decidido realizar un entrenamiento binario (dedo estirado o no estirado) por cada dedo, ya que realizar un entrenamiento multiclase habría provocado la necesidad de realizar vídeos por cada combinación de dedos diferentes ($2^5$ configuraciones posibles). Esto provocaría en los datos la existencia de muchas características en el entrenamiento (5x, siendo x el número de características por cada dedo), demasiadas etiquetas en el entrenamiento (una por configuración de dedos posible) y la necesidad de elaborar muchos vídeos. Debido a la alta dimensionalidad en características y muestras, así como el gran número de etiquetas, llevaría a la creación de un modelo muy complejo y costoso de elaborar en términos de tiempo.

En los siguientes apartados se presentan las pruebas de los entrenamientos binarios realizados a cada dedo. Para todos los dedos, se han realizado 3 pruebas y el enfoque de éstas ha sido el mismo:

- **Prueba 1**: para esta prueba se han utilizado los algoritmos de balanceo básicos, los métodos de selección de características de filtro y los modelos lineales, basados en Naive Bayes y los basados en vecinos más cercanos.

  - Métodos de balanceo: Ninguno, RUS, ROS.
  - Métodos de selección de características: FValue, MutualInfo (MI).
  - Algoritmos: RidgeClassifier (RC), SGDClassifier (SGDC), BernoulliNB (BNB), KNeighborsClassifier (KNC) y NearestCentroids (NC).

- **Prueba 2**: para esta prueba se han utilizado los algoritmos de balanceo básicos, lo métodos de selección de características de filtro y los modelos basados en máquinas de vectores de soporte, árboles, métodos de conjunto y redes neuronales.

- Métodos de balanceo: Ninguno, RUS, ROS.

- Métodos de selección de características: FValue, MutualInfo (MI).

- Algoritmos: SVC, DecisionTreeClassifier, ExtraTreeClassifier (ETC), BaggingClassifier, RandomForestClassifier (RFC), AdaBoostClassifier (ABC), GradientBoostingClassifier (GBC) y MLPClassifier (MLPC).

– **Prueba 3**: para esta prueba se ha utilizado como algoritmo de balanceo el que mejores resultados ha obtenido en las pruebas anteriores, los métodos de selección de características de wrapper y los mejores clasificadores obtenidos en las pruebas 1 y 2.

  - Métodos de balanceo: ROS.

  - Métodos de selección de características: SequentialForwardSelection y SequentialBackwardElimination.

  - Algoritmos: KNeighborsClassifier (KNC), RandomForestClassifier (RFC), GradientBoostingClassifier (GBC) y MLPClassifier (MLPC).

Los resultados de cada entrenamiento binario por dedo se muestran a continuación. Los resultados de todas estas pruebas se han basado en la métrica de "accuracy". Además, aunque se han ejecutado las pruebas sin balanceo de datos, no se han tenido en cuenta entre las tablas de los mejores resultados. Esto ha sido así ya que al no realizar balanceo, los modelos pueden ajustarse más a la clase mayoritaria, por lo que predecirían bien la mayoría de datos dando unos valores de accuracy altos. Sin embargo, estos modelos no serían válidos ya que no generalizan bien el problema, no distingue entre las clases.

### 4.1.1. Resultados pulgar

#### Prueba 1

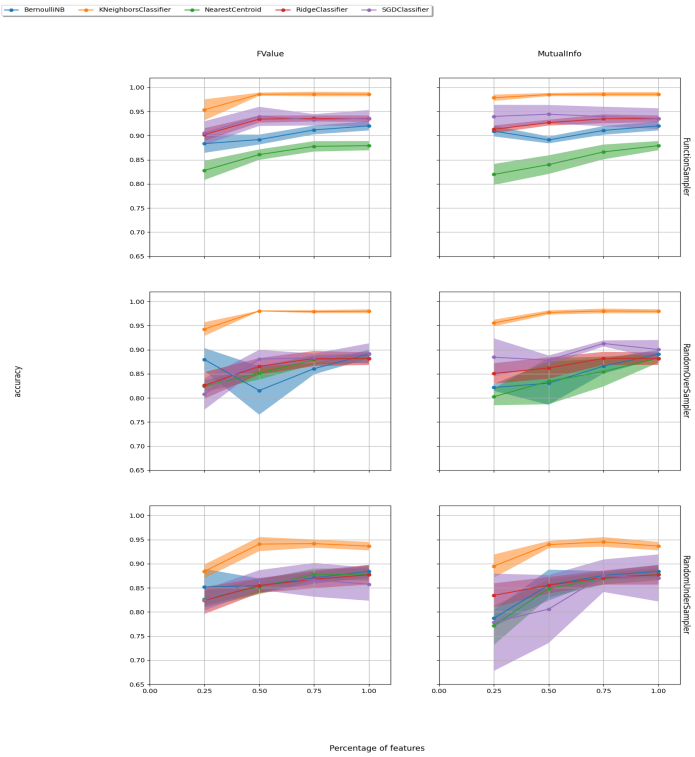



Figura 4.1: Resultados de accuracy para la prueba 1

En la figura 4.1 vemos que los mejores son aquellos que utilizan el clasificador KNeighborsClassifier (KNC) llegando a alcanzar valores de 0.98 en accuracy. Por otro lado, se observa que es mejor utilizar RandomOverSampler (ROS) que balancea el conjunto utilizando sobre-muestreo que RUS que utiliza sub-muestreo.

En cuanto al método de selección de características, no se observa claramente que uno sea mejor que otro, ambos métodos de filtro obtienen resultados parecidos. Por otro lado, como número de características parece conveniente descartar usar tan solo el 25 % ya que da peores resultados y el 100 % de características ya que no ofrece mejoras significativas.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 folds).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.5-KNC | 0.98034 | 0.00071 |
| ROS-MI0.75-KNC | 0.98033 | 0.00491 |
| ROS-FValue1-KNC | 0.97963 | 0.00410 |
| ROS-MI1-KNC | 0.97963 | 0.00410 |
| ROS-FValue0.75-KNC | 0.97893 | 0.00315 |

Tabla 4.1: Mejores modelos prueba 1.

Las 5 peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.5-SGDC | 0.80587 | 0.07047 |
| ROS-MI0.25-NC | 0.80233 | 0.01756 |
| RUS-MI0.25-BNB | 0.78687 | 0.01712 |
| RUS-MI0.25-SGDC | 0.77846 | 0.10095 |
| RUS-MI0.25-NC | 0.77178 | 0.04050 |

Tabla 4.2: Peores modelos prueba 1.

**Prueba 2**

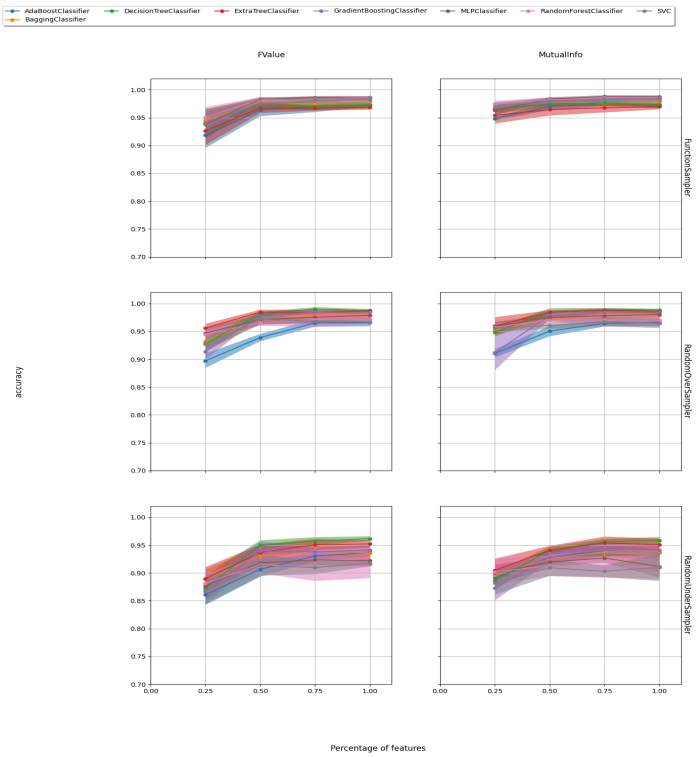



Figura 4.2: Resultados de accuracy para la prueba 2.

En la figura 4.2 se observa que los mejores resultados se dan para los modelos que utilizan como balanceo el algoritmo de RandomOverSampler (ROS). En cuanto

a los clasificadores, todos obtienen valores altos, destacando entre ellos MLPClassifier (MLPC) con un 0.99 de accuracy. Después de este, en segundo lugar destaca RandomForestClassifier (RFC) y en el tercero, SVC.

Por otro lado, el algoritmo de filtro para selección de características no afecta significativamente en los resultados. Lo que si parece claro, es que los mejores modelos usan un porcentaje más alto de características (75 o 100 %).

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 folds).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.75-MLPC | 0.99017 | 0.00345 |
| ROS-MI1-MLPC | 0.98876 | 0.00179 |
| ROS-MI0.75-MLPC | 0.98876 | 0.00345 |
| ROS-MI0.75-RFC | 0.98806 | 0.00303 |
| ROS-FValue1-MLPC | 0.98736 | 0.00301 |

Tabla 4.3: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.25-SVC | 0.87289 | 0.02213 |
| RUS-FValue0.25-MLPC | 0.87289 | 0.02317 |
| RUS-FValue0.25-ETC | 0.86868 | 0.02446 |
| RUS-FValue0.25-SVC | 0.86587 | 0.02080 |
| RUS-FValue0.25-ABC | 0.86131 | 0.01894 |

Tabla 4.4: Peores modelos prueba 2.

**Prueba 3**

En la figura 4.3 se observa que el resultado de estas pruebas es similar al del apartado anterior obteniendo valores de 0.98 en accuracy. Mientras que en las pruebas anteriores con métodos de filtro se ve que los mejores modelos se dan para un porcentaje de características grande (75 % o 100 %), para esta prueba se ve que el porcentaje de características puede ser reducido a (50 o 75 %) lo cuál podría ser una ventaja para el modelo final ya que implementaría un modelo de menor dimensionalidad. De esta manera, ahorraríamos en tiempo y memoria debido a que hay un menor número de características.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

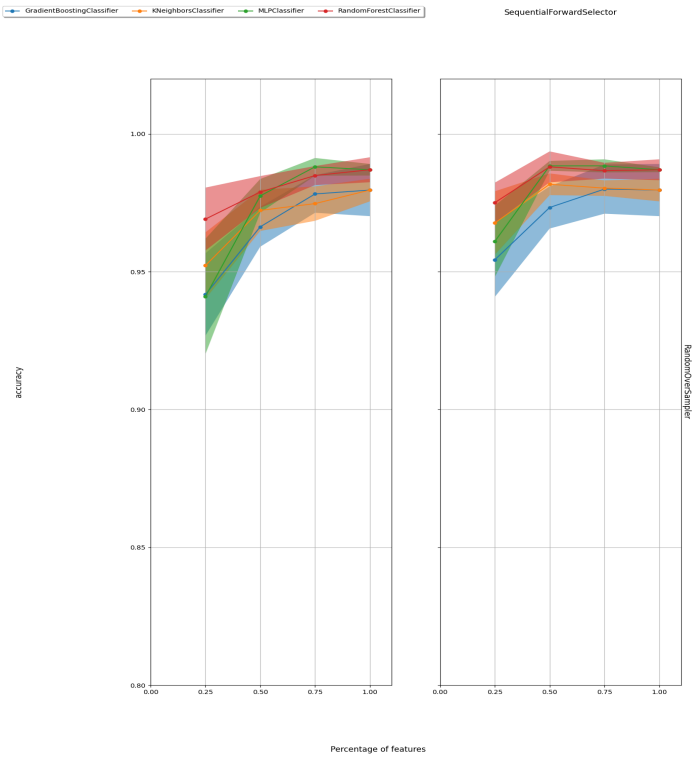



Figura 4.3: Resultados de accuracy para la prueba 3

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.75-MLPC | 0.98841 | 0.00237 |
| ROS-SFS0.5-MLPC | 0.98841 | 0.00179 |
| ROS-SBS0.75-MLPC | 0.98806 | 0.00321 |
| ROS-SFS0.5-RFC | 0.98806 | 0.00561 |
| ROS-SBS1-MLPC | 0.98701 | 0.00210 |

Tabla 4.5: Mejores modelos prueba 3.

Las cinco mejores pruebas observadas son las siguientes:

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.25-MLPC | 0.96102 | 0.01294 |
| ROS-SFS0.25-GBC | 0.95435 | 0.01348 |
| ROS-SBS0.25-KNC | 0.95225 | 0.01204 |
| ROS-SBS0.25-GBC | 0.94172 | 0.01494 |
| ROS-SBS0.25-MLPC | 0.94101 | 0.02094 |

Tabla 4.6: Mejores modelos prueba 3.

### 4.1.2. Resultados índice

#### Prueba 1

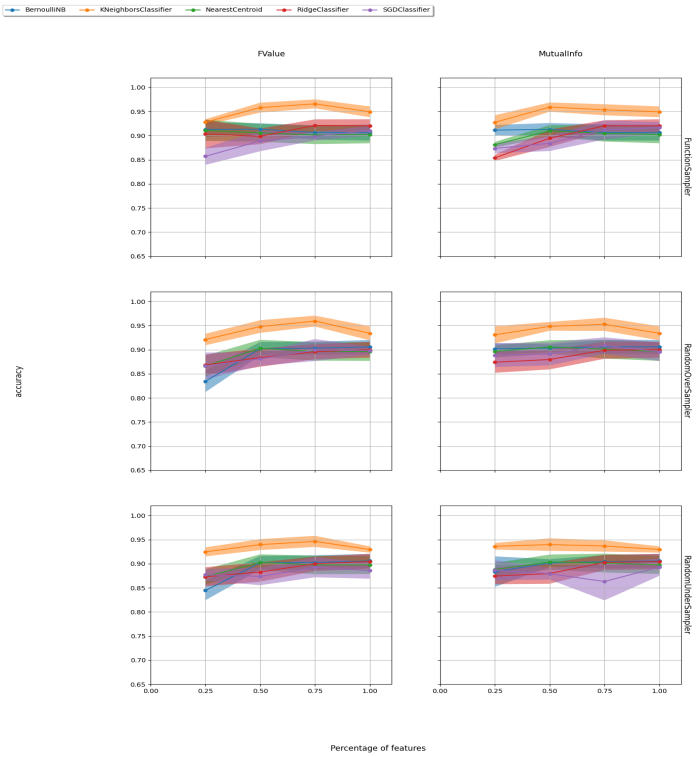



Figura 4.4: Resultados de accuracy para la prueba 1

En la imagen 4.4 se observa que para el dedo índice pasa algo similar que en el pulgar, los mejores resultados se dan con RandomOverSampler (ROS) y KNeighborsClassifier (KNC). En este caso, el accuracy llega a alcanzar el valor de 0.95.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.75-KNC | 0.95927 | 0.01130 |
| ROS-MI0.75-KNC | 0.95259 | 0.01367 |
| ROS-MI0.5-KNC | 0.94839 | 0.00906 |
| ROS-FValue0.5-KNC | 0.94803 | 0.01318 |
| RUS-FValue0.75-KNC | 0.94628 | 0.01154 |

Tabla 4.7: Mejores modelos prueba 1.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.75-SGDC | 0.86305 | 0.03903 |
| FunctionSampler-FValue0.25-SGDC | 0.85708 | 0.01776 |
| FunctionSampler-MI0.25-RC | 0.85393 | 0.00653 |
| RUS-FValue0.25-BNB | 0.84446 | 0.02025 |
| ROS-FValue0.25-BNB | 0.83427 | 0.02224 |

Tabla 4.8: Peores modelos prueba 1.

**Prueba 2**

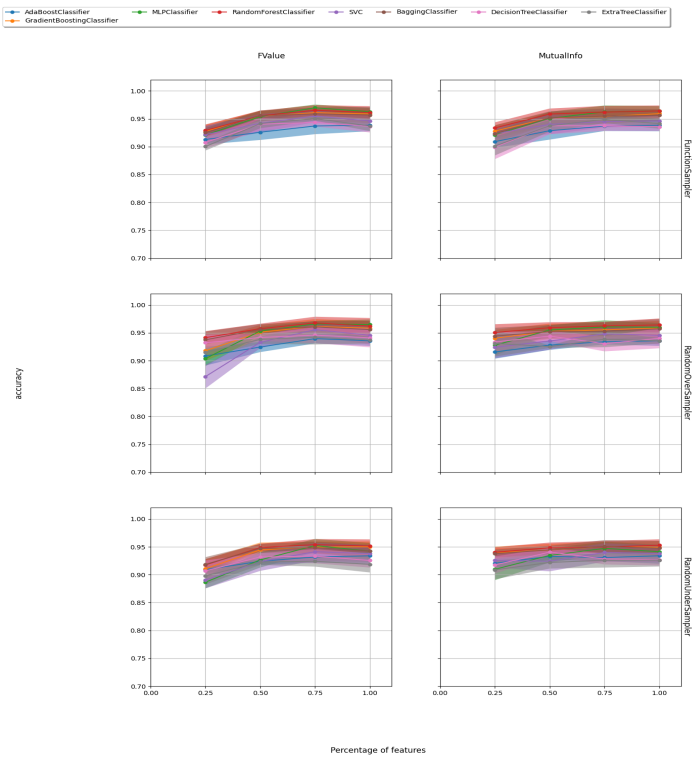



Figura 4.5: Resultados de accuracy para la prueba 2

En la figura 4.5 se puede observar que para muestreo funciona mejor el algoritmo RandomSampler y como clasificadores RandomForestClassifier (RFC) y MLPClassifier (MLPC) llegando a obtener valores de accuracy de 0.96 El número de características usadas es alto (75 o 100 % de las características).

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0,75-RFC | 0.96769 | 0.01121 |
| ROS-FValue0,75-MLPC | 0.96594 | 0.00552 |
| ROS-FValue1-MLPC | 0.96488 | 0.00911 |
| ROS-MI1-RFC | 0.96418 | 0.01159 |
| ROS-MI0,75-RFC | 0.96243 | 0.00743 |

Tabla 4.9: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| FunctionSampler-MI0,25-DecisionTreeClassifier | 0.89923 | 0.02164 |
| RUS-FValue0,25-ETC | 0.89783 | 0.01520 |
| RUS-FValue0,25-SVC | 0.89046 | 0.01493 |
| RUS-FValue0,25-MLPC | 0.88660 | 0.01111 |
| ROS-FValue0,25-SVC | 0.87149 | 0.02133 |

Tabla 4.10: Peores modelos prueba 2.

### Prueba 3

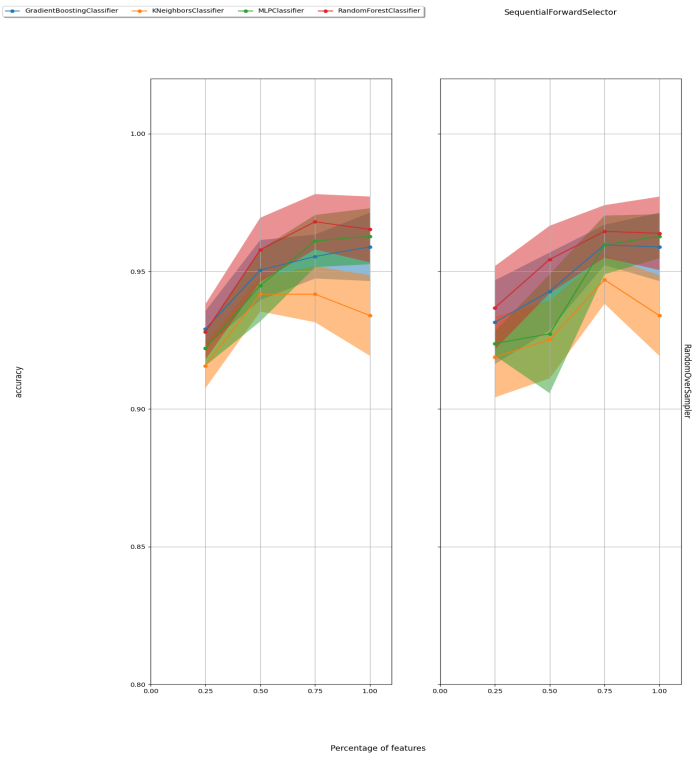



Figura 4.6: Resultados de accuracy para la prueba 3

En la figura 4.6 se puede observar que los resultados obtenidos son similares a los de la prueba anterior con los mismos porcentajes de características, por lo que usar los

métodos de wrapper en este caso no implica una mejora significativa en los resultados. Los mejores resultados de accuracy obtenidos llegan al 0.96.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS0.75-RFC | 0.96804 | 0.01004 |
| ROS-SBS1-RFC | 0.96523 | 0.01195 |
| ROS-SFS0.75-RFC | 0.96453 | 0.00953 |
| ROS-SFS1-RFC | 0.96383 | 0.01333 |
| ROS-SFS1-MLPC | 0.96278 | 0.00798 |

Tabla 4.11: Mejores modelos prueba 3.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.5-KNC | 0.92521 | 0.01407 |
| ROS-SFS0.25-MLPC | 0.92381 | 0.00438 |
| ROS-SBS0.25-MLPC | 0.92205 | 0.00631 |
| ROS-SFS0.25-KNC | 0.91889 | 0.01467 |
| ROS-SBS0.25-KNC | 0.91573 | 0.00818 |

Tabla 4.12: Peores modelos prueba 3.

### 4.1.3. Resultados corazón

#### Prueba 1

En la imagen 4.7 se observa que los mejores resultados se dan con RandomOverSampler (ROS) y KNeighborsClassifier (KNC). En este caso, el accuracy llega a alcanzar el valor de 0.97.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.5-KNC | 0.97261 | 0.00344 |
| ROS-MI0.75-KNC | 0.97261 | 0.00879 |
| ROS-FValue0.75-KNC | 0.96980 | 0.00825 |
| ROS-MI0.5-KNC | 0.96770 | 0.01020 |
| ROS-FValue1-KNC | 0.96629 | 0.00408 |

Tabla 4.13: Mejores modelos prueba 1.

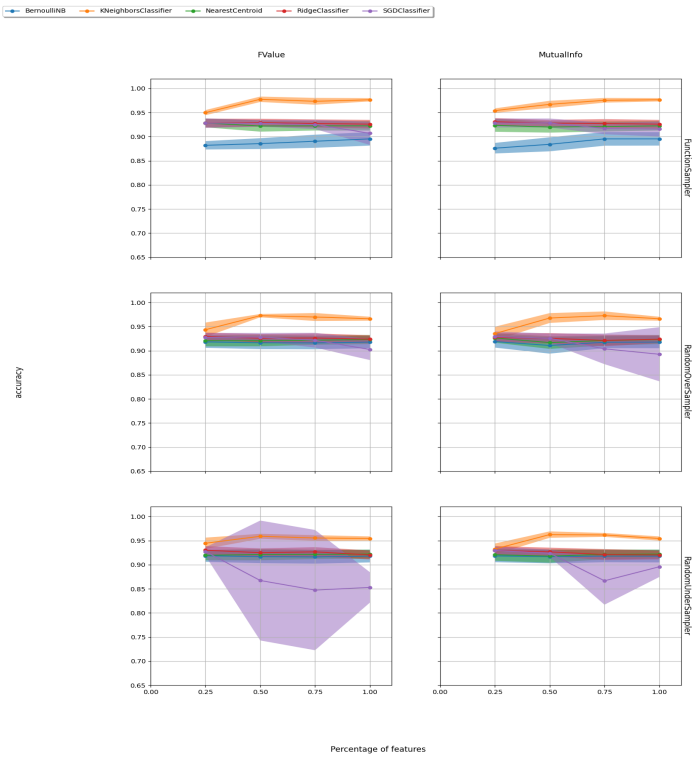



Figura 4.7: Resultados de accuracy para la prueba 1

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| FunctionSampler-MI0.25-BNB | 0.87605 | 0.01093 |
| RUS-FValue0.5-SGDC | 0.86721 | 0.12435 |
| RUS-MI0.75-SGDC | 0.86659 | 0.04957 |
| RUS-FValue1-SGDC | 0.85288 | 0.03157 |
| RUS-FValue0.75-SGDC | 0.84729 | 0.12475 |

Tabla 4.14: Peores modelos prueba 1.

### Prueba 2

En la figura 4.8 se observa que los mejores resultados se dan para RandomOverSampler (ROS) y RandomForestClassifier (RFC) llegando a obtener valores de accuracy de 0.98.

Ambos algoritmos de selección de características funcionan bien con un porcentaje de características alto (75 %, 100 %).

Las cinco mejores pruebas observadas son las siguientes:

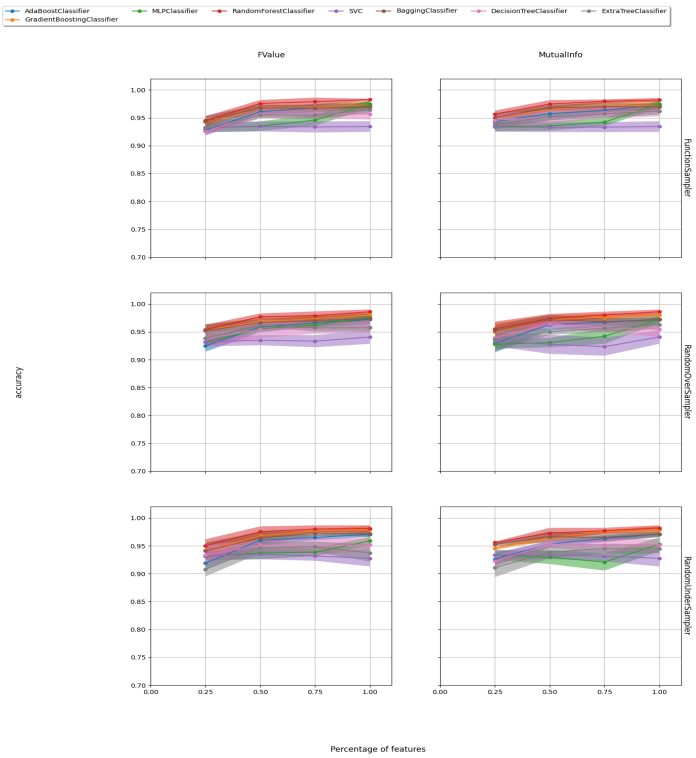



Figura 4.8: Resultados de accuracy para la prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98596 | 0.00430 |
| ROS-MI1-RFC | 0.98595 | 0.00401 |
| RUS-MI1-RFC | 0.98209 | 0.00489 |
| RUS-FValue1-RFC | 0.98104 | 0.00581 |
| RUS-FValue0.75-RFC | 0.98034 | 0.00640 |

Tabla 4.15: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.25-DecisionTreeClassifier | 0.92275 | 0.00993 |
| RUS-MI0.75-MLPC | 0.92099 | 0.01542 |
| RUS-FValue0.25-ABC | 0.91924 | 0.01053 |
| RUS-MI0.25-ETC | 0.91081 | 0.01685 |
| RUS-FValue0.25-ETC | 0.90800 | 0.01294 |

Tabla 4.16: Peores modelos prueba 2.

### Prueba 3

En la figura 4.9 se observa que los mejores resultados son parecidos a los del apartado anterior, no obteniendo mejoras significativas aplicando los métodos de wrapper para

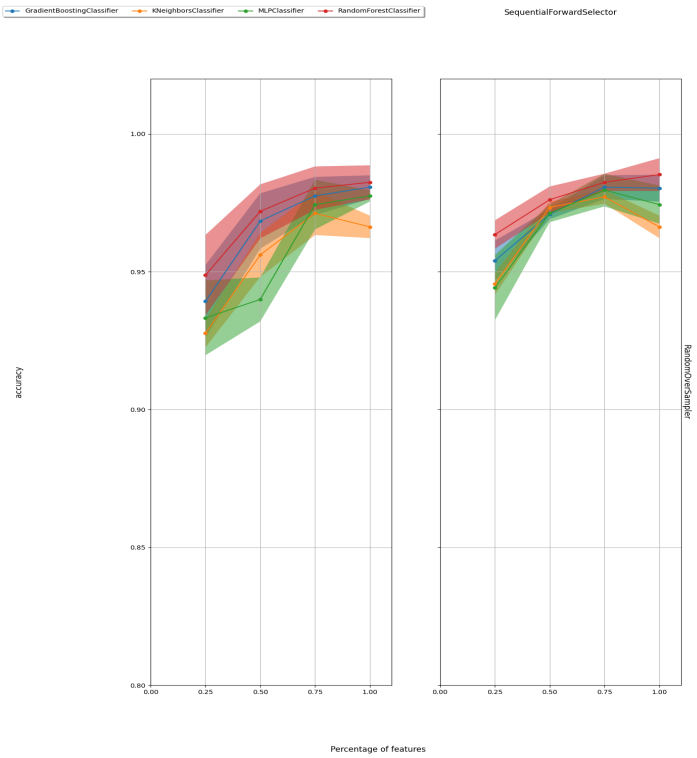



Figura 4.9: Resultados de accuracy para la prueba 3

selección de características.

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS1-RFC | 0.98525 | 0.00593 |
| ROS-SBS1-RFC | 0.98245 | 0.00618 |
| ROS-SFS0.75-RFC | 0.98245 | 0.00313 |
| ROS-SFS0.75-GBC | 0.98069 | 0.00429 |
| ROS-SBS1-GBC | 0.98069 | 0.00430 |

Tabla 4.17: Mejores modelos prueba 3.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.25-MLPC | 0.94417 | 0.01182 |
| ROS-SBS0.5-MLPC | 0.93995 | 0.00799 |
| ROS-SBS0.25-GBC | 0.93926 | 0.01313 |
| ROS-SBS0.25-MLPC | 0.93328 | 0.01357 |
| ROS-SBS0.25-KNC | 0.92767 | 0.00513 |

Tabla 4.18: Peores modelos prueba 3.

### 4.1.4. Resultados anular

#### Prueba 1

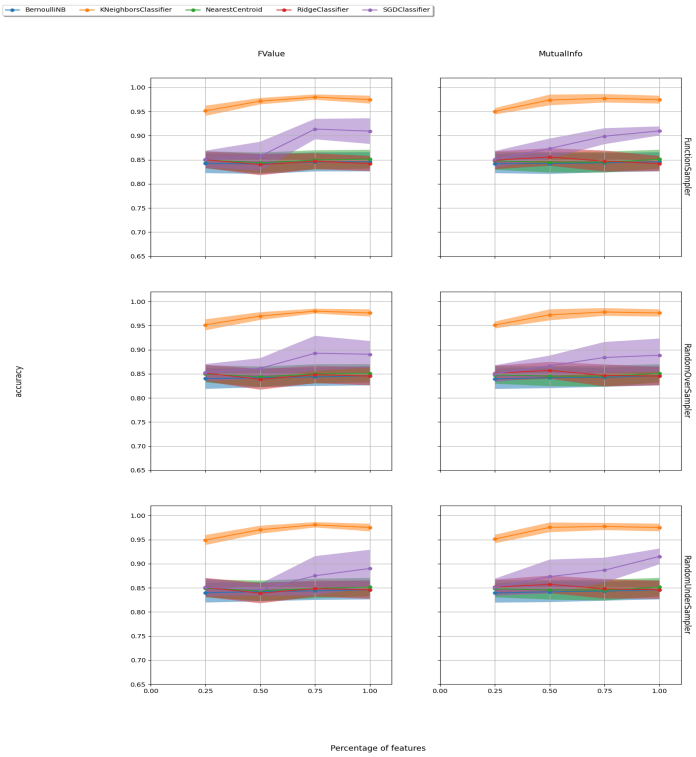



Figura 4.10: Resultados de accuracy para la prueba 1

En la imagen 4.10 se observa que los mejores resultados se dan para ambos RandomOverSampler (ROS) y RandomUnderSampler (RUS) con el clasificador KNeighborsClassifier (KNC). En este caso, el accuracy llega a alcanzar el valor de 0.98.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.75-KNC | 0.98069 | 0.00556 |
| ROS-FValue0.75-KNC | 0.97999 | 0.00505 |
| ROS-MI0.75-KNC | 0.97823 | 0.00805 |
| RUS-MI0.75-KNC | 0.97718 | 0.00745 |
| ROS-FValue1-KNC | 0.97613 | 0.00750 |

Tabla 4.19: Mejores modelos prueba 1.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.25-BNB | 0.83990 | 0.02052 |
| ROS-MI0.25-BNB | 0.83955 | 0.02103 |
| RUS-MI0.25-BNB | 0.83955 | 0.02029 |
| RUS-FValue0.5-RC | 0.83885 | 0.02108 |
| ROS-FValue0.5-RC | 0.83850 | 0.02162 |

Tabla 4.20: Peores modelos prueba 1.

### Prueba 2

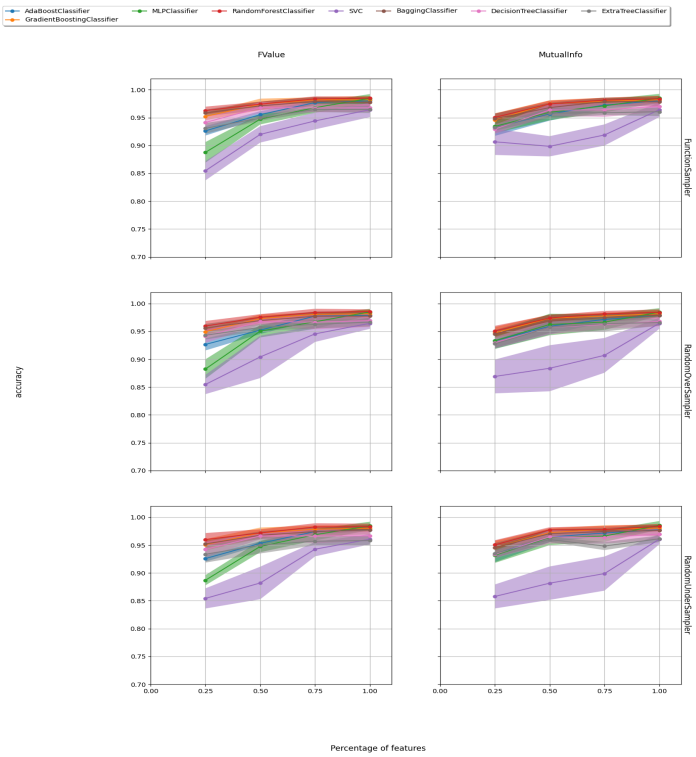



Figura 4.11: Resultados de accuracy para la prueba 2

En la figura 4.11 se observa que los mejores resultados se dan para RandomOverSampler (ROS) usado con los clasificadores RandomForestClassifier (RFC) y MLPClassifier (MLPC) llegando a valores de accuracy de 0.98. Los métodos de selección de características de filtro FValue y MutualInfo (MI) se comportan de forma similar obteniendo buenos resultados con un porcentaje alto de características (100 %).

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98560 | 0.00406 |
| RUS-MI1-MLPC | 0.98560 | 0.00773 |
| ROS-FValue1-MLPC | 0.98490 | 0.00562 |
| ROS-MI1-MLPC | 0.98455 | 0.00715 |
| RUS-FValue1-MLPC | 0.98455 | 0.00757 |

Tabla 4.21: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.25-SVC | 0.86905 | 0.03035 |
| RUS-MI0.25-SVC | 0.85781 | 0.02149 |
| FunctionSampler-FValue0.25-SVC | 0.85465 | 0.01734 |
| ROS-FValue0.25-SVC | 0.85465 | 0.01734 |
| RUS-FValue0.25-SVC | 0.85429 | 0.01809 |

Tabla 4.22: Peores modelos prueba 2.

### Prueba 3

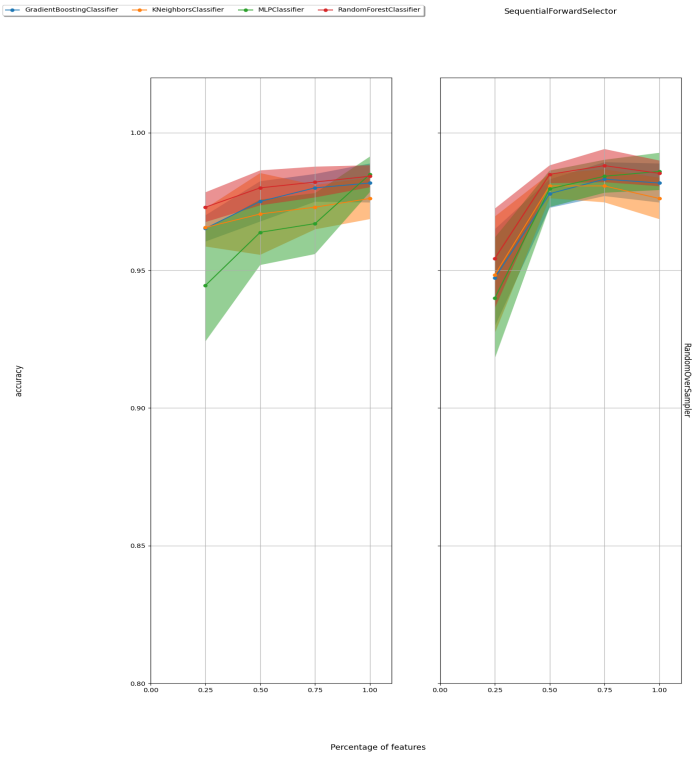



Figura 4.12: Resultados de accuracy para la prueba 3

En la figura 4.12 se observa que los resultados son parecidos a los del apartado anterior, aunque en este caso se observa que en algunos casos ha bajado el porcentaje de características usadas (75 o 100 %) en un caso concreto usa solo el 50 %.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.75-RFC | 0.98806 | 0.00602 |
| ROS-SFS1-MLPC | 0.98595 | 0.00676 |
| ROS-SFS1-RFC | 0.98525 | 0.00466 |
| ROS-SFS0.5-RFC | 0.98490 | 0.00325 |
| ROS-SBS1-MLPC | 0.98490 | 0.00644 |

Tabla 4.23: Mejores modelos prueba 3.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.25-RFC | 0.95436 | 0.01809 |
| ROS-SFS0.25-KNC | 0.94839 | 0.02117 |
| ROS-SFS0.25-GBC | 0.94734 | 0.01761 |
| ROS-SBS0.25-MLPC | 0.94453 | 0.02034 |
| ROS-SFS0.25-MLPC | 0.93997 | 0.02194 |

Tabla 4.24: Peores modelos prueba 3.

### 4.1.5. Resultados meñique

#### Prueba 1

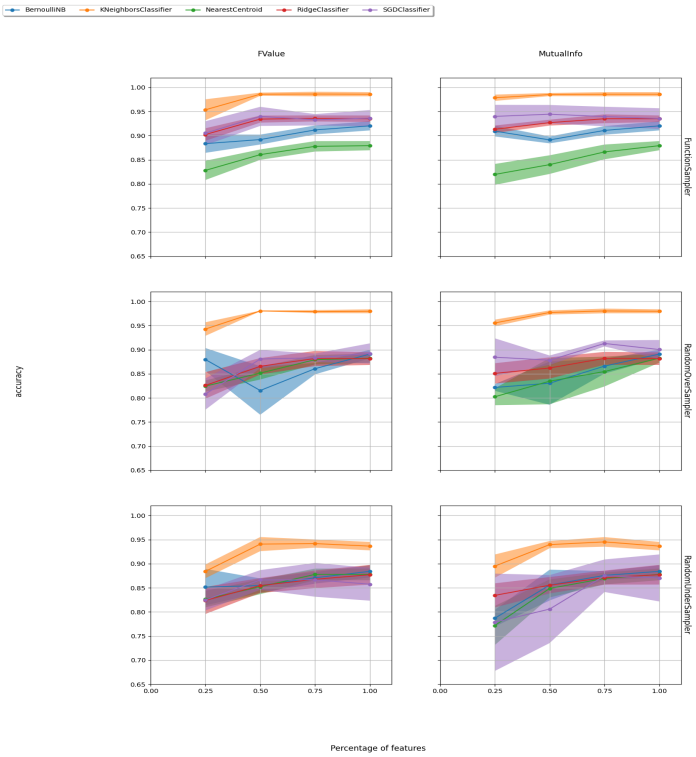



Figura 4.13: Resultados de accuracy para la prueba 1

En la imagen 4.13 se observa que los mejores resultados se dan para ambos RandomOverSampler (ROS) y RandomUnderSampler (RUS) con el clasificador KNeighborsClassifier (KNC). En este caso, el accuracy llega a alcanzar el valor de 0.98.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.75-KNC | 0.98420 | 0.00638 |
| RUS-FValue0.75-KNC | 0.98420 | 0.00793 |
| ROS-FValue0.75-KNC | 0.98385 | 0.00697 |
| RUS-MI0.5-KNC | 0.98279 | 0.00642 |
| RUS-MI0.75-KNC | 0.98244 | 0.00618 |

Tabla 4.25: Mejores modelos prueba 1.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| FunctionSampler-FValue0.5-SGDC | 0.83743 | 0.01597 |
| FunctionSampler-FValue0.75-NC | 0.83709 | 0.00990 |
| ROS-FValue0.75-NC | 0.83673 | 0.01048 |
| RUS-FValue0.75-NC | 0.83533 | 0.01150 |
| RUS-FValue0.5-SGDC | 0.83428 | 0.02403 |

Tabla 4.26: Peores modelos prueba 1.

### Prueba 2

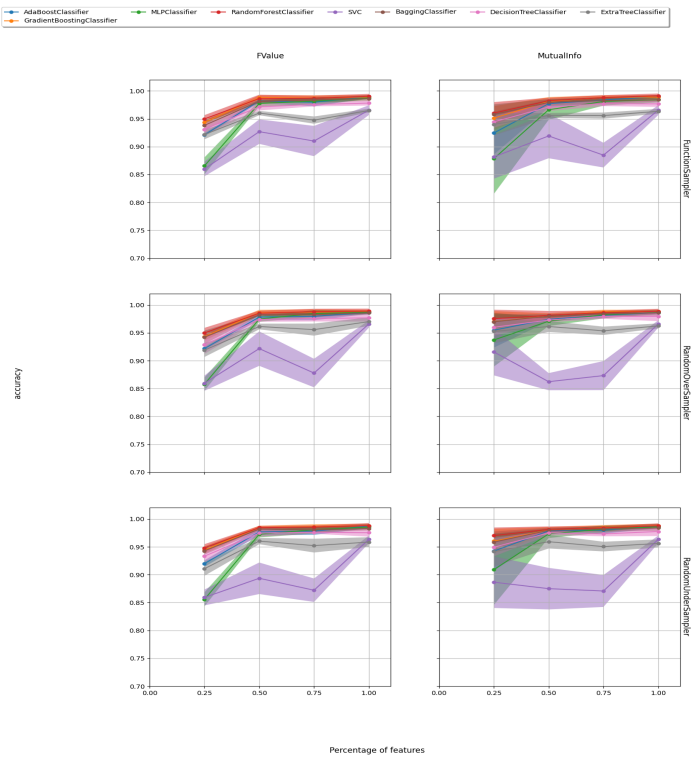



Figura 4.14: Resultados de accuracy para la prueba 2

En 4.14 se observa que los mejores resultados se dan para ROS usando los clasificadores RandomForestClassifier (RFC) y GradientBoostingClassife con un porcentaje alto de características (100 %). El mejor valor de accuracy obtenido en este caso es de 0.98.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98911 | 0.00436 |
| ROS-FValue1-GBC | 0.98876 | 0.00263 |
| ROS-MI1-GBC | 0.98876 | 0.00263 |
| ROS-MI1-RFC | 0.98876 | 0.00439 |
| RUS-FValue1-RFC | 0.98806 | 0.00450 |

Tabla 4.27: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| FunctionSampler-FValue0.25-SVC | 0.85991 | 0.01300 |
| ROS-FValue0.25-SVC | 0.85956 | 0.01354 |
| RUS-FValue0.25-SVC | 0.85921 | 0.01421 |
| ROS-FValue0.25-MLPC | 0.85780 | 0.01198 |
| RUS-FValue0.25-MLPC | 0.85570 | 0.01254 |

Tabla 4.28: Peores modelos prueba 2.

### Prueba 3

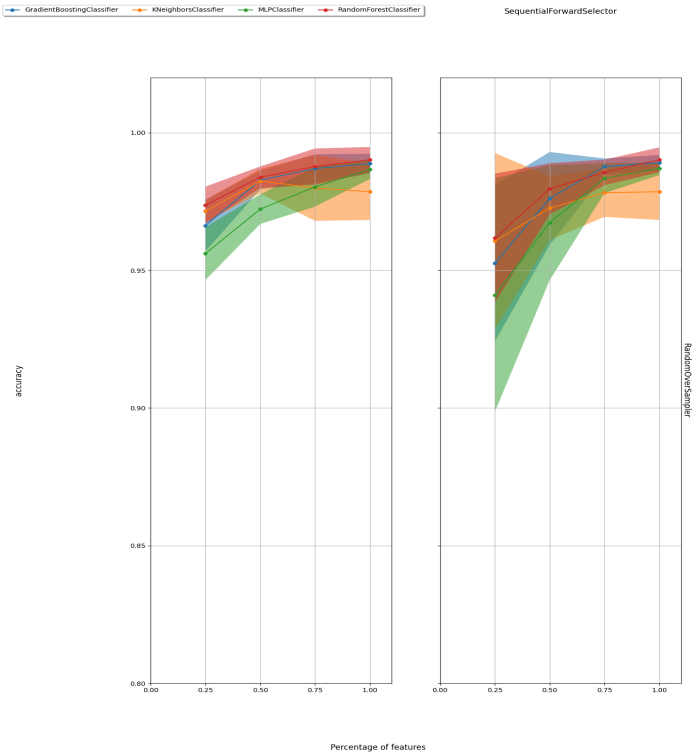



Figura 4.15: Resultados de accuracy para la prueba 3

En la figura 4.15 se observa que los resultados son parecidos a los del apartado anterior llegando a obtener valores de accuracy de 0.99. Los porcentajes de características usados en estas mejores pruebas tampoco varían respecto al apartado anterior.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS1-RFC | 0.99017 | 0.00465 |
| ROS-SFS1-RFC | 0.99017 | 0.00452 |
| ROS-SFS1-GBC | 0.98912 | 0.00281 |
| ROS-SBS1-GBC | 0.98876 | 0.00361 |
| ROS-SBS0.75-RFC | 0.98771 | 0.00657 |

Tabla 4.29: Mejores modelos prueba 3.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.25-RFC | 0.96172 | 0.02341 |
| ROS-SFS0.25-KNC | 0.96065 | 0.03199 |
| ROS-SBS0.25-MLPC | 0.95611 | 0.00961 |
| ROS-SFS0.25-GBC | 0.95259 | 0.02842 |
| ROS-SFS0.25-MLPC | 0.94099 | 0.04242 |

Tabla 4.30: Peores modelos prueba 3.

### 4.1.6. Resumen de las pruebas del contaje de dedos y modelos elegidos

En las siguientes tablas se muestra de forma resumida los mejores modelos observados en los apartados anteriores.

#### Pulgar

Los mejores modelos vistos son los siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.75-MLPC | 0.99017 | 0.00345 |
| ROS-MI1-MLPC | 0.98876 | 0.00179 |
| ROS-MI0.75-MLPC | 0.98876 | 0.00345 |
| ROS-SFS0.75-MLPC | 0.98841 | 0.00237 |
| ROS-SFS0.5-MLPC | 0.98841 | 0.00179 |

Tabla 4.31: Resumen mejores modelos.

Se observa que los modelos han tenido resultados muy altos de accuracy llegando a 0.99. Este caso se da combinando el balanceo con RandomOverSampler (ROS),

selección de características FValue con el 75 % de ellas y clasificador MLPClassifier (MLPC).

### Índice

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS0.75-RFC | 0.96804 | 0.01004 |
| ROS-FValue0,75-RFC | 0.96769 | 0.01121 |
| ROS-FValue0,75-MLPC | 0.96594 | 0.00552 |
| ROS-SBS1-RFC | 0.96523 | 0.01195 |
| ROS-FValue1-MLPC | 0.96488 | 0.00911 |

Tabla 4.32: Resumen mejores modelos.

Se observa que los modelos han tenido resultados altos de accuracy llegando a 0.96. Este caso se da combinando el balanceo con RandomOverSampler (ROS), selección de características SequentialBackwardSelector (SBS) con el 75 % de ellas y clasificador RandomForestClassifier (RFC). También, se observa que el segundo con mejores resultados usa el mismo algoritmo de balanceo, clasificador y porcentaje de características distinguiéndose únicamente por el método de selección de características que ha sido FValue. Por ello, elegiríamos este ya que es menos costoso en tiempo ejecutar un método de filtro, en comparación con los de wrapper.

### Corazón

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98596 | 0.00430 |
| ROS-MI1-RFC | 0.98595 | 0.00401 |
| ROS-SFS1-RFC | 0.98525 | 0.00593 |
| ROS-SBS1-RFC | 0.98245 | 0.00618 |
| ROS-SFS0.75-RFC | 0.98245 | 0.00313 |
| RUS-MI1-RFC | 0.98209 | 0.00489 |

Tabla 4.33: Resumen mejores modelos.

Se observa que los modelos han tenido resultados muy altos de accuracy llegando a 0.98. Este caso se da combinando el balanceo con RandomOverSampler (ROS), selección de características FValue con el 100 % de ellas y clasificador RandomForestClassifier (RFC).

**Anular**

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.75-RFC | 0.98806 | 0.00602 |
| ROS-SFS1-MLPC | 0.98595 | 0.00676 |
| ROS-FValue1-RFC | 0.98560 | 0.00406 |
| RUS-MI1-MLPC | 0.98560 | 0.00773 |
| ROS-SFS1-RFC | 0.98525 | 0.00466 |

Tabla 4.34: Resumen mejores modelos.

Se observa que los modelos han tenido resultados muy altos de accuracy llegando a 0.98. Este caso se da combinando el balanceo con RandomOverSampler (ROS), selección de características SequentialForwardSelector (SFS) con el 75 % de ellas y clasificador RandomForestClassifier (RFC).

**Meñique**

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS1-RFC | 0.99017 | 0.00465 |
| ROS-SFS1-RFC | 0.99017 | 0.00452 |
| ROS-SFS1-GBC | 0.98912 | 0.00281 |
| ROS-FValue1-RFC | 0.98911 | 0.00436 |
| ROS-FValue1-GBC | 0.98876 | 0.00263 |

Tabla 4.35: Resumen mejores modelos.

Se observa que los modelos han tenido resultados muy altos de accuracy llegando a 0.99. Este caso se da combinando el balanceo con RandomOverSampler (ROS), selección de características SequentialForwardSelector (SFS) con el 100 % de ellas y clasificador RandomForestClassifier (RFC).

## 4.2. Resultados de los modelos de movimiento

### 4.2.1. Movimiento versión 1

La versión 1 de entrenamiento del módulo de análisis de movimiento se refiere a las características explicadas en la sección 3.5. En esta versión, las características de cada una de las muestras presentan información del avance de movimiento de un frame a otro. Se ha utilizado esta primera aproximación para ver los resultados que se podrían obtener usando tan solo un único avance entre dos frames. En esta versión no se espera obtener buenos resultados, ya que el uso de solo la información de un avance

de movimiento para obtener la clasificación de movimiento de un vídeo completo parece insuficiente. Además, en el contexto del prototipo final, esta aproximación requeriría realizar muchas predicciones, una por cada frame de vídeo que se procese lo cuál es costoso en tiempo y el prototipo no sería capaz de analizar el gesto en tiempo real.

### Prueba 1

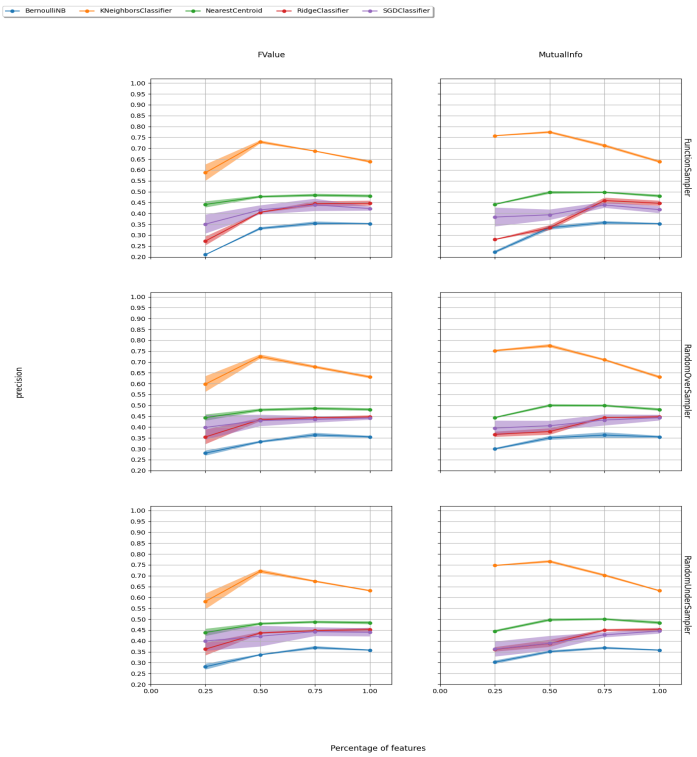



Figura 4.16: Resultados de accuracy para la prueba 1

En la figura 4.16 se observa que los mejores resultados se dan para los modelos que usan técnicas de sobre-muestreo ROS y el mejor clasificador en estos modelos es KNeighborsClassifier (KNC). En cuanto a los métodos de selección de características, destaca MutualInfo (MI)usando el 50 % de las características. En estas pruebas los mejores valores obtenidos de accuracy han sido de 0.77.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean precision | Std precision |
|---|---|---|
| ROS-MI0.5-KNC | 0.77467 | 0.00738 |
| RUS-MI0.5-KNC | 0.76583 | 0.00589 |
| ROS-MI0.25-KNC | 0.75219 | 0.00420 |
| RUS-MI0.25-KNC | 0.74708 | 0.00285 |
| ROS-FValue0.5-KNC | 0.72495 | 0.00964 |

Tabla 4.36: Mejores modelos prueba 1.

Las cinco peores pruebas observadas son las siguientes:

| | Mean precision | Std precision |
|---|---|---|
| ROS-FValue0.25-BNB | 0.28042 | 0.01182 |
| FunctionSampler-MI0.25-RC | 0.27981 | 0.00199 |
| FunctionSampler-FValue0.25-RC | 0.27280 | 0.02067 |
| FunctionSampler-MI0.25-BNB | 0.22204 | 0.00564 |
| FunctionSampler-FValue0.25-BNB | 0.20986 | 0.00134 |

Tabla 4.37: Peores modelos prueba 1.

**Prueba 2**

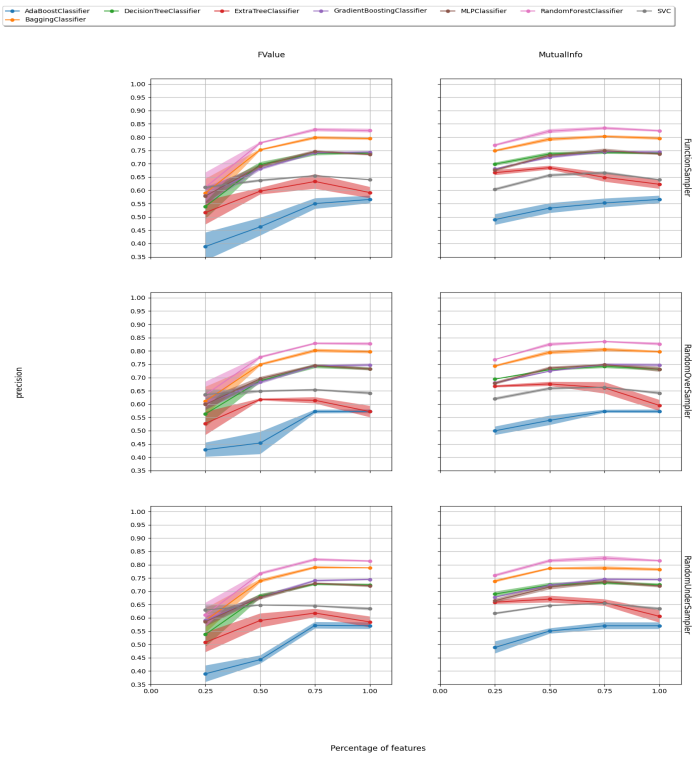



Figura 4.17: Resultados de accuracy para la prueba 2

En la figura 4.17 se observa que los mejores resultados se dan para los modelos que usan técnicas de sobre-muestreo ROS y el mejor clasificador en estos modelos es

RandomForestClassifier (RFC). En cuanto a los métodos de selección de características, no destaca ninguno en concreto, obteniendo los dos buenos resultados. Cabe destacar, que los mejores modelos han usado el 75 % de las características. En estas pruebas los mejores valores obtenidos de accuracy han sido de 0.83.

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean precision | Std precision |
|---|---|---|
| ROS-MI0.75-RFC | 0.83532 | 0.00280 |
| ROS-FValue0.75-RFC | 0.82852 | 0.00383 |
| ROS-FValue1-RFC | 0.82687 | 0.00572 |
| ROS-MI1-RFC | 0.82639 | 0.00558 |
| ROS-MI0.5-RFC | 0.82490 | 0.00616 |

Tabla 4.38: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean precision | Std precision |
|---|---|---|
| ROS-FValue0.5-ABC | 0.45362 | 0.04192 |
| RUS-FValue0.5-ABC | 0.44290 | 0.01570 |
| ROS-FValue0.25-ABC | 0.42839 | 0.02666 |
| RUS-FValue0.25-ABC | 0.38964 | 0.03102 |
| FunctionSampler-FValue0.25-ABC | 0.38877 | 0.05254 |

Tabla 4.39: Peores modelos prueba 2.

### 4.2.2. Movimiento versión 2

La versión 2 de entrenamiento del módulo de análisis de movimiento se refiere a las características explicadas en la sección 3.5. En esta versión, las características de cada una de las muestras presentan información del avance de movimiento de un movimiento completo, es decir, agrupan la información de los avances frame a frame de un vídeo completo y lo usan como muestra para el entrenamiento. En esta versión se esperan mejores resultados ya que utilizan toda la información del movimiento. Sin embargo, requiere la realización de un módulo adicional para detectar el principio y fin del movimiento. Este módulo podría estar basado en el uso de sensores. Otro enfoque que podría darse es realizar el análisis de movimiento usando información de un rango de frames (por ejemplo, 4 avances de movimiento) y añadir una nueva etiqueta al entrenamiento que sería la "ausencia de movimiento". De esto se habla en la sección de conclusiones y trabajos futuros, pero no se lleva a cabo en este proyecto.

### Prueba 1

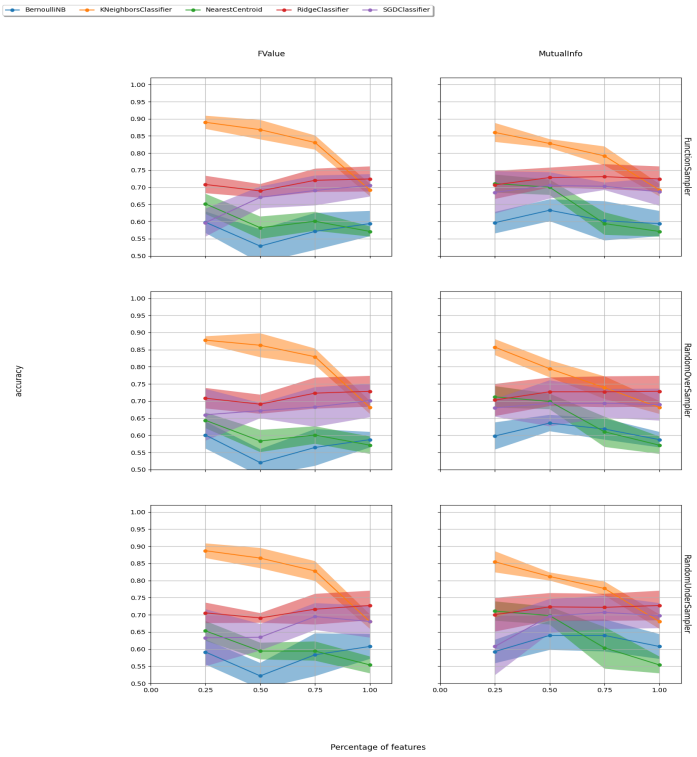



Figura 4.18: Resultados de accuracy para la prueba 1

En la figura 4.18 se observa que los mejores modelos usan ambas técnicas básicas de muestreo ROS y RUS obteniendo buenos resultados (0.88 accuracy). El clasificador que emplean los mejores modelos es KNeighborsClassifier (KNC) y la mayoría de ellos usan el método de selección de características FValue con un porcentaje bastante bajo de características (25 o 50 %).

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.25-KNC | 0.88710 | 0.02136 |
| ROS-FValue0.25-KNC | 0.87770 | 0.01144 |
| RUS-FValue0.5-KNC | 0.86555 | 0.02961 |
| ROS-FValue0.5-KNC | 0.86289 | 0.03496 |
| ROS-MI0.25-KNC | 0.85754 | 0.02334 |

Tabla 4.40: Mejores modelos prueba 1.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue1-NC | 0.55378 | 0.02462 |
| RUS-MI1-NC | 0.55378 | 0.02462 |
| FunctionSampler-FValue0.5-BNB | 0.52834 | 0.04902 |
| RUS-FValue0.5-BNB | 0.52159 | 0.03779 |
| ROS-FValue0.5-BNB | 0.52024 | 0.03953 |

Tabla 4.41: Peores modelos prueba 1.

### Prueba 2

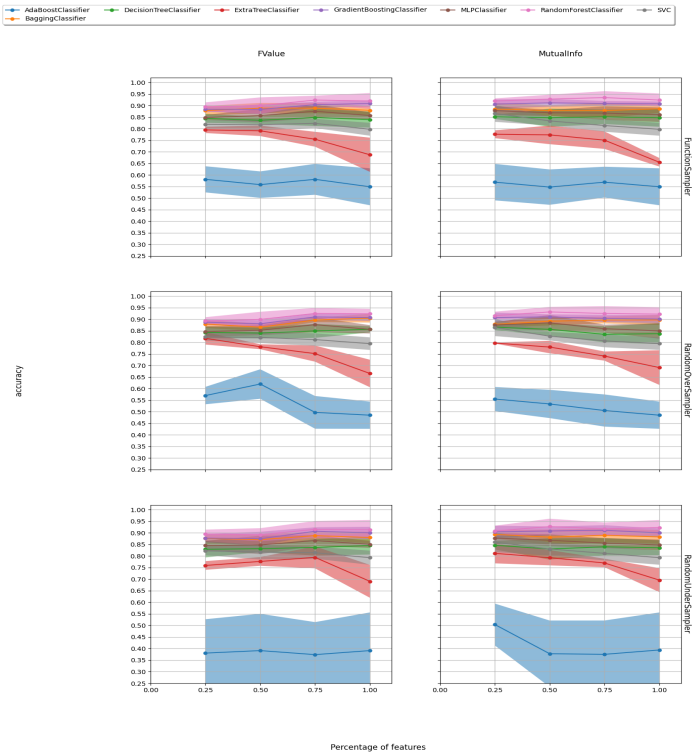



Figura 4.19: Resultados de accuracy para la prueba 2

En la figura 4.19 se observa que los mejores modelos usan ambas técnicas básicas de muestreo ROS y RUS obteniendo buenos resultados (0.93 accuracy). El clasificador que emplean los mejores modelos es RandomForestClassifier (RFC) y la mayoría de ellos usan el método de selección de características MutualInfo0 con un porcentaje bastante intermedio de características (50 o 75 %).

Como resumen se dejan los valores concretos de accuracy así como la desviación estándar en las pruebas realizadas (cross-validation con 5 fols).

Las cinco mejores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.5-RFC | 0.93146 | 0.02178 |
| RUS-MI0.5-RFC | 0.92744 | 0.03403 |
| ROS-MI0.75-RFC | 0.92478 | 0.03153 |
| ROS-FValue0.75-RFC | 0.92475 | 0.02411 |
| RUS-MI1-RFC | 0.92340 | 0.03247 |

Tabla 4.42: Mejores modelos prueba 2.

Las cinco peores pruebas observadas son las siguientes:

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.5-ABC | 0.39137 | 0.15906 |
| RUS-FValue0.25-ABC | 0.38059 | 0.14639 |
| RUS-MI0.5-ABC | 0.37789 | 0.14350 |
| RUS-MI0.75-ABC | 0.37520 | 0.14625 |
| RUS-FValue0.75-ABC | 0.37393 | 0.14056 |

Tabla 4.43: Peores modelos prueba 2.

### 4.2.3. Resumen de las pruebas del módulo de movimiento y modelos elegidos

En las siguientes tablas se muestra de forma resumida los mejores modelos observados en los apartados anteriores.

### 4.2.4. Movimiento versión 1

| | Mean precision | Std precision |
|---|---|---|
| ROS-MI0.75-RFC | 0.83532 | 0.00280 |
| ROS-FValue0.75-RFC | 0.82852 | 0.00383 |
| ROS-FValue1-RFC | 0.82687 | 0.00572 |
| ROS-MI1-RFC | 0.82639 | 0.00558 |
| ROS-MI0.5-RFC | 0.82490 | 0.00616 |

Tabla 4.44: Resumen mejores modelos.

Se observa que los modelos han tenido resultados buenos de accuracy llegando a 0.83. Este caso se da combinando el balanceo RandomOverSampler (ROS), selección de características MutualInfo (MI) con el 75 % de ellas y clasificador RandomForestClassifier (RFC).

**Movimiento versión 2**

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.5-RFC | 0.93146 | 0.02178 |
| RUS-MI0.5-RFC | 0.92744 | 0.03403 |
| ROS-MI0.75-RFC | 0.92478 | 0.03153 |
| ROS-FValue0.75-RFC | 0.92475 | 0.02411 |
| RUS-MI1-RFC | 0.92340 | 0.03247 |

Tabla 4.45: Resumen mejores modelos.

Se observa que los modelos han tenido resultados buenos de accuracy llegando a 0.93. Este caso se da combinando el balanceo RandomOverSample (ROS), selección de características MutualInfo (MI) con el 50 % de ellas y clasificador RandomForestClassifier (RFC).

## 4.3. Modelos elegidos y test

En esta sección se muestra como resumen los modelos elegidos para cada dedo y movimiento, así como los resultados de la evaluación final de accuracy sobre el conjunto de test.

| | | | | |
|---|---|---|---|---|
| Pulgar | ROS-FValue0.75-MLPC | 0.99017 | 0.00345 | 0.98878 |
| Indice | ROS-SBS0.75-RFC | 0.96804 | 0.01004 | 0.96633 |
| Indice | ROS-FValue0,75-RFC | 0.96769 | 0.01121 | 0.97194 |
| Corazon | ROS-FValue1-RFC | 0.98596 | 0.00430 | 0.97335 |
| Anular | ROS-SFS0.75-RFC | 0.98806 | 0.00602 | 0.98177 |
| Meñique | ROS-SBS1-RFC | 0.99017 | 0.00465 | 0.98878 |
| Movimiento | ROS-MI0.5-RFC | 0.93146 | 0.02178 | 0.8877 |

Tabla 4.46: Resumen mejores modelos.

En el caso del índice que elegimos dos modelos, nos quedamos con el que usa como selección de características FValue en base a los resultados y que es mucho más rápido de ejecutar. En el caso del movimiento, finalmente se ha elegido el modelo de datos de la versión 2 en base a sus resultados significativamente mejores.

Se observa que para los modelos elegidos los resultados volviendo a re-entrenar y realizando la evaluación sobre los datos de test es similar que para los de cross-validation. Por lo que, finalmente estos son los modelos que hemos seleccionado para resolver nuestro problema. Cabe destacar que para el módulo del movimiento el resultado es un poco peor que en test, pero sigue siendo un alto valor de accuracy.

# Capítulo 5

# Conclusiones y trabajos futuros

Las principales conclusiones derivadas de este proyecto pueden resumirse en:

- Es posible el uso de visión por computador juntamente con métodos de *Machine-Learning* para reconocer la posición de la palma de la mano, el número de dedos extendidos, y los movimientos capturados por una cámara situada alrededor de la muñeca.
- Es posible la implementación de la lógica de teclado TYPYN propuesta en la patente [19].
- La detección de movimiento requiere del uso de sensores externos tipo *tap* o acelerómetros para aumentar la fiabilidad de la detección del inicio y final del movimiento.

Por otro lado, las tareas que se propondrían para una segunda iteración de este proyecto son:

- Implementar sensores para aumentar la fiabilidad de detección de inicio y fin de movimiento.
- Testear y validar algoritmos de reconocimiento de dedos y movimientos con un tamaño de muestra de usuarios de unos 100.
- Utilizar un módulo de cámara que se comunique con un dispositivo móvil, que se encargará de la computación.

# Capítulo 6

# Bibliografía


[1] Wikipedia contributors. Computadora, 2023.

[2] Javier Bermúdez. Tipos de teclados y mejores modelos. *Blog PcComponentes*, 2023.

[3] Teclados virtuales - la realidad superó la fantasía, 2018.

[4] Wikipedia contributors. Teclado qwerty, 2023.

[5] Tim Harford. Cómo el "teclado qwerty"llegó a convertirse en el más popular de todos a pesar de no ser el más eficiente. *BBC News Mundo*, 2019.

[6] Bengar. La máquina de escribir, 2015.

[7] Ivan. Evolución máquina de escribir-ordenador, 2021.

[8] David David García. El primer ordenador — primer odenador apple — el primer portátil. *Blog de Info-Computer*, 2023.

[9] La historia del teclado del ordenador, 2023.

[10] Javier Pastor. Qwerty vs dvorak: así nacieron los dos grandes teclados de nuestro tiempo. *www.xataka.com*, 2020.

[11] Carolina Gonzalez Valenzuela. Historia de los teléfonos móviles: desde su origen hasta la actualidad. *Computer Hoy*, 2023.

[12] Wikipedia contributors. Teléfono móvil, 2023.

[13] Enrique Perez. La historia de las primeras pantallas táctiles y su origen en el tráfico aéreo. *xataka.com*, 2020.

[14] Este fue el primer teléfono con pantalla táctil y no se trata del iphone, 2021.

[15] Cómo han evolucionado las pantallas de los teléfonos inteligentes desde 1994 hasta hoy, 2018.

[16] Las desventajas de teclados virtuales, 2019.

[17] Yilda Morillo. Realidad virtual. qué es, tipos, ventajas, desventajas, aplicaciones. *futuroelectrico.com*, 2022.

[18] Ginebra: Organización Mundial de la Salud. Directrices de la oms sobre actividad física y hábitos sedentarios: de un vistazo. *WHO guidelines on physical activity and sedentary behaviour: at a glance*, 2020.

[19] Marcos Lara Gonzalez. Software for keyboard-less typing based upon gestures, 2023.

[20] Samantha. Gestos con las manos: su importancia para el lenguaje. *blog-es.kinedu.com*, 2015.

[21] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. TaplD: Rapid Touch Interaction in Virtual Reality using Wearable Sensing. *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, 00:519–528, 2021.

[22] Munir Oudah, Ali Al-Naji, and Javaan Chahl. Hand Gesture Recognition Based on Computer Vision: A Review of Techniques. *Journal of Imaging*, 6(8):73, 2020.

[23] Regan Mandryk, Mark Hancock, Mark Perry, Anna Cox, Marco Speicher, Anna Maria Feit, Pascal Ziegler, and Antonio Krüger. Selection-based Text Entry in Virtual Reality. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[24] Fang Hu, Peng He, Songlin Xu, Yin Li, and Cheng Zhang. FingerTrak: Continuous 3D Hand Pose Tracking by Deep Learning Hand Silhouettes Captured by Miniature Thermal Cameras on Wrist. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(2):1–24, 2020.

[25] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. MediaPipe: A Framework for Building Perception Pipelines. *arXiv*, 2019.

[26] Desarrolladores de google. Mediapipe hands, 2023.

[27] aprendeia.com. Conjunto de datos desbalanceados, 2023.

[28] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113–141, 2013.

[29] Ismael Ramos-Pérez, Álvar Arnaiz-González, Juan J. Rodríguez, and César García-Osorio. When is resampling beneficial for feature selection with imbalanced wide data? *Expert Systems with Applications*, 188:116015, 2022.

[30] Haibo He and E A Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[31] Carles Morales Boada. Introducción al feature selection (o cómo escoger las variables adecuadas). *linkedin.com*, 2018.

[32] scikit learn.org. Selección de características - sklearn, 2023.

[33] scikit learn.org. Aprendizaje supervisado - sklearn, 2023.

[34] Aman Gupta. Técnicas de selección de características en aprendizaje máquina. *analyticsvidhya.com*, 2023.

[35] minitab. Comprensión del análisis de varianza (anova) y la prueba f. *blog.minitab.com*, 2019.

[36] Glosario machine learning, 2023.

[37] Paloma Recuero. Tipos de aprendizaje en machine learning: supervisado y no supervisado. *telefonicatech.com*, 2021.

[38] Vlad Miller. Explorando algoritmos de aprendizaje automático supervisado. *toptal.com*, 2023.

[39] Pablo Huet. Cómo entrenar un modelo de machine learning con scikit-learn. *OpenWebinars.net*, 2023.

[40] aprendemachinelearning.com. Qué es overfitting y underfitting y cómo solucionarlo, 2017.

[41] aprendemachinelearning.com. Sets de entrenamiento, test y validación, 2020.

[42] sitiobigdata.com. Diagnosticar rendimiento del modelo de aprendizaje automático, 2019.

[43] Alejandro Alvarez. Guía tkinter, 2015.

[44] Raspberry. Cámara raspberry, 2023.

[45] Raspberry. Documentación cámara raspberry, 2023.

[46] Francisco Jesús Mateos Díaz. Introducción al análisis estadístico de datos, 2023.

[47] Almudena Bonaplata. Análisis exploratorio de datos, 2019.

[48] probabilidadyestadistica.net. Asimetría y curtosis, 2023.

[49] maximaformacion.es. Errores comunes que puedes evitar con un simple gráfico, 2023.

# Lista de Figuras

# Lista de Tablas

# Anexos

# Anexos A

# Análisis estadístico de datos

Para cada característica de las muestras, podemos realizar un análisis, para ver como se comporta dicha variable para las diferentes clases y ver cuáles podrían ser las más útiles para el problema a resolver. En esta sección se explican conceptos de carácter general que ayudan a realizar dicho análisis y se basa en la referencias [46, 47, 48, 49].

## A.1. Parámetros estadísticos generales de una distribución

Los parámetros estadísticos se pueden clasificar en las principales categorías:

- Medidas de posición: valores de la variable estadística que se caracterizan por la posición que ocupan dentro del rango de valores posibles de esta. Entre ellos se distinguen:
  - Medidas de tendencia central:
    - medias:
      - ◇ media aritmética: suma de un conjunto de valores dividida entre el número total de sumandos
      - ◇ media geométrica de una cantidad arbitraria de números (por ejemplo, n números): raíz enésima del producto de todos los números.
      - ◇ media armónica de una cantidad finita de números: recíproco, o inverso, de la media aritmética de los recíprocos de dichos valores.
    - moda: valor de la variable con mayor frecuencia absoluta.
    - mediana: valor de la variable que deja por debajo de sí a la mitad de los datos, una vez que estos están ordenados de menor a mayor.
  - Medidas de posición no central: cuantiles (valores de la variable estadística que dejan por debajo de sí determinada cantidad de los datos)

    - cuartiles: dividen la cantidad de datos en cuatro partes
    - deciles: dividen la cantidad de datos en diez partes
    - percentiles: divide la cantidad de datos en cien partes.

- Medidas de dispersión: resumen la heterogeneidad de los datos, lo separados que estos están entre sí. Hay dos tipos, básicamente:

    - Medidas de dispersión absolutas: vienen dadas en las mismas unidades en las que se mide la variable. Algunas son:
        - recorrido o rango: diferencia entre el mayor y el menor valor que toma la misma.
        - desviaciones medias: media aritmética de las desviaciones absolutas (desviación de los valores de la variable respecto de la tendencia central c, por ej. c puede ser la media o la mediana).
        - varianza: media aritmética de los cuadrados de las desviaciones respecto de la media.
        - desviación típica/estándar (sigma, s o sd): raíz cuadrada positiva de la varianza.
    - Medidas de dispersión relativa: informan de la dispersión en términos relativos, como un porcentaje o una proporción. Se incluyen entre estas:
        - coeficiente de variación de Pearson: desviación estándar como porcentaje de la media aritmética. Dicho de otra forma, refleja el número de veces que la media está contenida en la desviación típica en porcentaje.
        - coeficiente de apertura: cociente entre los valores extremos de la distribución de datos.
        - recorridos relativos: número de veces que la media está contenida en el recorrido.
        - índice de desviación respecto de la mediana: número de veces que la mediana está contenida en la desviación media respecto de la mediana.

- Medidas de forma: informa sobre el aspecto que tiene la gráfica de la distribución. Entre ellas están:

    - coeficientes de asimetría: de Fisher, Bowley o Pearson.
    - coeficientes de curtosis: de Fisher, Kelley o percentílico.

– Otros parámetros: existen otros parámetros con propósito más específico. Algunos ejemplos son:

  - Proporción: número de veces que se presenta ese dato respecto al total de datos.
  - Números índice.
  - Tasas.
  - Coeficiente de Gini.

Existen otros parámetros como los parámetros bidimensionales:

– Covarianza: se define como el valor esperado (o la media) del producto de sus desviaciones de sus valores esperados individuales

– Coeficiente de correlación lineal, Pearson r, o coeficiente de correlación de Pearson: es el cociente entre la covarianza de dos variables y el producto de sus desviaciones estándar. El valor obtenido es un valor entre [-1, 1] donde -1 indica una relación lineal negativa perfecta, 1 indica una relación lineal positiva perfecta y 0 indica que no hay relación lineal entre dos variables.

## A.2. Histograma

Un **histograma** es una representación gráfica de una variable en forma de barras. El punto en x donde se ubica la barra indica el valor de una característica, cuantitativa y continua; la superficie de la barra que ocupa en el eje y indica la frecuencia de dicho valor que representa.

Sirve para obtener una vista general de la distribución de la población, o de la muestra ofreciendo una visión de grupo que puede permitir observar una preferencia, o tendencia, por parte de la muestra o población por ubicarse hacia una determinada región de valores dentro del espectro de valores posibles que pueda adquirir la característica.

## A.3. Diagrama de caja

Un **diagrama de caja** (también, diagrama de caja y bigotes o *boxplot*) es un método estandarizado para representar gráficamente una serie de datos numéricos a través de sus cuartiles. Este diagrama muestra a simple vista los *cuartiles de los datos* $Q_1$ (25 %), $Q_2$ o *mediana* (50 %) y $Q_3$ (75 %), el *rango intercuartílico IQR*, los "bigotes"

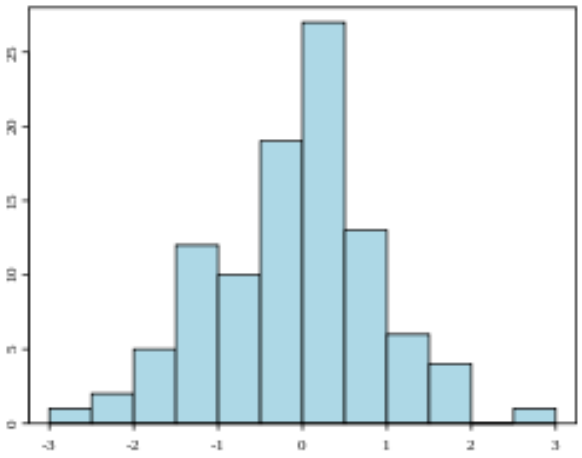


Figura A.1: Histograma.

correspondientes a los *valores mínimos y máximos* de la serie o al límite que determina el atípico leve, y los *valores atípicos leves y extremos*. Conviene recordar que se utilizan las bisagras de Tukey, y no los cuartiles, a la hora de dibujar la caja del gráfico, aunque los resultados son semejantes en muestras grandes.

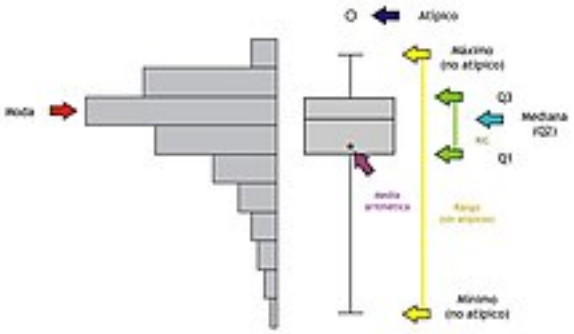



Figura A.2: Diagrama de caja.

Este tipo de diagramas pone en una sola dimensión los datos de un histograma, proporcionando una visión general de la simetría de la distribución de los datos, dada por la posición de la mediana en el rectángulo formado por los cuartiles $Q_1$ y $Q_3$; permite ver la presencia de valores atípicos; como es la dispersión de los puntos con la mediana; los percentiles 25 y 75 y los valores máximos y mínimos.

## A.4. Valores atípicos o outliers

Un **valor atípico** es una observación que es numéricamente distante del resto de los datos. Es importante tener en cuenta si en un conjunto de datos existen o no valores atípicos, ya que las estadísticas derivadas de este tipo de conjuntos de datos serán frecuentemente engañosas. Por ejemplo, la media de una variable puede ser muy

influenciada por estos valores atípicos, en este caso es mejor considerar la mediana como estadístico.

Los valores atípicos pueden ser indicativos de datos que pertenecen a una población diferente del resto de las muestras o de muestras erróneamente calculadas de la población.

Uno de los métodos más usados para clasificarlos por su sencillez y resultados es el **test de Tukey**. Este método toma como referencia la diferencia entre el primer cuartil $Q_1$ y el tercer cuartil $Q_3$ o rango intercuartílico IQR ($IQR = Q_3 - Q_1$) y evalúa si las observaciones van más allá de los llamados límites interiores $Q_1$ y $Q_3$ para determinar si es o no un valor atípico. Se considera un valor atípico leve aquel que cumple: $q < Q_1 - 1{,}5 \cdot \text{IQR}$ o $q > Q_3 + 1{,}5 \cdot \text{IQR}$ y un valor atípico extremo aquel que cumple: $q < Q_1 - 3 \cdot \text{IQR}$ o $q > Q_3 + 3 \cdot \text{IQR}$

En un diagrama de caja se puede observar un valor atípico como el que se encuentra 1,5 veces esa distancia IQR de uno de esos cuartiles (atípico leve) o a 3 veces esa distancia (atípico extremo).

## A.5. Frecuencia acumulada

La **frecuencia acumulada o frecuencia acumulativa** es la frecuencia de ocurrencia de valores de un fenómeno menor que un valor de referencia. El fenómeno puede ser una variable aleatoria que varia en el tiempo o en el espacio.

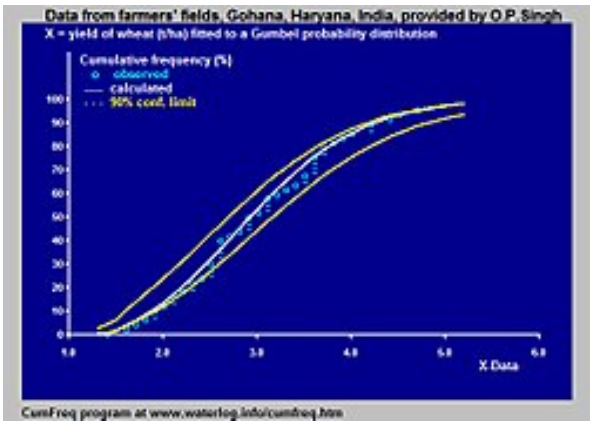



Figura A.3: Frecuencia acumulada.

## A.6. Tamaño de efecto

El **tamaño del efecto** es un valor que mide la fuerza de la relación entre dos variables en una población, o una estimación de esa cantidad basada en una muestra. Puede referirse al valor de una estadística calculada a partir de una muestra de datos , el

valor de un parámetro para una población hipotética o a la ecuación que operacionaliza cómo las estadísticas o los parámetros conducen al valor del tamaño del efecto.

Existen muchas medidas diferentes del tamaño de efecto y distintos tipos. Aquí se explican algunas tipos:

### A.6.1. Medidas basadas en la varianza explicada

Se pueden usar algunas medidas como:

- Pearson r o coeficiente de correlación: se usa ampliamente como tamaño del efecto cuando se dispone de datos cuantitativos pareados. Según Cohen: r es pequeño cuando es 0.1, medio cuando es 0.3 y grande cuando es 0.5.

- Coeficiente de determinación o $r^2$: es el cuadrado del coeficiente de correlación de Pearson. En el caso de datos apareados, esta es una medida de la proporción de varianza que comparten las dos variables y varía de 0 a 1.

- $F^2$ de Cohen: se usa en el contexto de una prueba F para ANOVA o regresión múltiple . Su cantidad de sesgo (sobre-estimación del tamaño del efecto para el ANOVA) depende del sesgo de su medida subyacente de varianza explicada (p. ej., $r^2$, $eta^2$, $omega^2$). Para el coeficiente de determinación se calcula como $f^2 = r^2/(1 - r^2)$

### A.6.2. Medidas basadas en la diferencia de medias

- Diferencia de media estandarizada DME: es la diferencia de medias de las poblaciones dividido entre la desviación estándar basada en una o ambas poblaciones.

- d de Cohen: usa la diferencia de medias estandarizada usando como desviación estándar la desviación estándar combinada (para dos muestras independientes) definida de la siguiente forma: $s = \sqrt{\frac{(n_1-1)*s_1^2+(n_2-1)*s_2^2}{n_1+n_2-2}}$, siendo $s^2$ la varianza muestral.

### A.6.3. Análisis del tamaño de efecto

Consideraciones del tamaño de efecto según Cohen y ampliado por Sawilowsky:

| | |
|---|---|
| Muy pequeña | 0.01 |
| Pequeño | 0.20 |
| Medio | 0.50 |
| Grande | 0.80 |
| Muy grande | 1.20 |
| Enorme | 2.0 |

## A.7. Diagrama de dispersión

Un **diagrama de dispersión** o gráfica de dispersión o gráfico de burbujas gráfico de bolas es un tipo de diagrama matemático que utiliza las coordenadas cartesianas para mostrar los valores de dos variables para un conjunto de datos.

El diagrama de dispersión nos permite ver el grado de correlación (no causalidad) entre dos variables.

# Anexos B

# Descripción Base de Datos

**DOMINIOS**

tpNombre = varchar(20);

tpNombreLong = varchar(30);

tpText = text;

tpIllumination = enum("high", "medium", "low");

tpMovement = enum("derecha", "izquierda", "arriba", "abajo", "centro", "nocentro");

tpSelectorType = enum("all", "min", "mean");

tpDataset = enum("fingers", "movement");

tpTraining = enum("0", "4", "8", "12", "16", "20", "derecha", "izquierda", "arriba", "abajo", "centro", "nocentro");

**ESQUEMAS DE RELACIÓN**

User(username: tpNombre *NOT NULL*;
password: tpNombre *NOT NULL*);

Collection(name: varchar(20) *NOT NULL*;
author: tpNombre *NOT NULL*, clave ajena de User;
camera: tpNombre, clave ajena de Camera;
illumination: tpIllumination;
description: tpText);

Camera(name: tpNombre *NOT NULL*;
description: tpText);

Share(collection: tpNombre *NOT NULL*, clave ajena de Collection;

username: tpNombre *NOT NULL*, clave ajena de User);

Search(id: int *NOT NULL* AUTO_INCREMENT;
videoSet: int *NOT NULL*, clave ajena de VideoSet;
search: tpText *NOT NULL*);
* Restricción: (id, search) únicos, no se puede especificar debido al campo tpText.

VideoSet(id: int *NOT NULL* AUTO_INCREMENT);

SearchVideo(videoSet: int *NOT NULL*, clave ajena de VideoSet;
video: int *NOT NULL*, clave ajena de Video);

Video(id: int *NOT NULL* AUTO_INCREMENT;
collection: tpNombre *NOT NULL*, clave ajena de Collection;
movement: tpMovement *NOT NULL*;
fingers: tpNombre *NOT NULL*;
name: tpNombre *NOT NULL*;
(collection, movement, fingers, name), UNIQUE);
* Atributo identificador alternativo: (collection, movement, fingers, name)

Selector(id: int *NOT NULL* AUTO_INCREMENT;
type: tpSelectorType *NOT NULL*;
nParts: int;
(type, nParts), UNIQUE);
* Atributo identificador alternativo: (type, nParts)

Selection(id: int *NOT NULL* AUTO_INCREMENT;
video: int *NOT NULL*, clave ajena de Video;
selector: int *NOT NULL*, clave ajena de Selector;
from_: int *NOT NULL*;
to_: int *NOT NULL*;
numFrameAnterior: int;
numFrameActual: int;
(video, selector, from_, to_), UNIQUE);
* Atributo identificador alternativo: (video, selector, from_, to_)

FingersData(selection: int *NOT NULL*, clave ajena de Selection;

paramsPulgar(22): float *NOT NULL*;
paramsIndice(22): float *NOT NULL*;
paramsCorazon(22): float *NOT NULL*;
paramsAnular(22): float *NOT NULL*;
paramsMeñique(22): float *NOT NULL*);

* paramsdedo(22) indica que tiene 22 columnas de características por cada dedo. Son las especificadas en la sección de elección de características *(**Angela:** Añadir referencia a la sección)*

MovementData(selection: int *NOT NULL*, clave ajena de Selection;
params(X): float *NOT NULL*);

* params(X) indica que tiene X columnas de características de movimiento. Son las especificadas en la sección de elección de características *(**Angela:** Añadir referencia a la sección)*

Dataset(id: int *NOT NULL* AUTO_INCREMENT;
videoSet: int *NOT NULL*, clave ajena de videoSet;
type_: tpDataset *NOT NULL*;
selector: int *NOT NULL*, clave ajena de selector;
multiclass: boolean *NOT NULL*;
sampling: tpNombreLong *NOT NULL*;
fs: tpNombreLong *NOT NULL*;
(videoSet, type_, selector, multiclass, sampling, fs), UNIQUE);

* Atributo identificador alternativo: (videoSet, type_, selector, multiclass, sampling, fs)

TrainingParams(id: int *NOT NULL* AUTO_INCREMENT;
dataset: int *NOT NULL*, clave ajena de Dataset;
type_: tpTraining *NOT NULL*;
prueba: int *NOT NULL*,
classifier: tpNombreLong *NOT NULL*;
gridSearch: boolean *NOT NULL*;
(dataset, type_, prueba, classifier, gridSearch), UNIQUE);

* Atributo identificador alternativo: (dataset, type_, prueba, classifier, gridSearch)

TrainingData(trainingParams: int *NOT NULL*, clave ajena de trainingParams;
trainingTime: float *NOT NULL*;

predictingTime: float *NOT NULL*;
testingTime: float *NOT NULL*;
trainingBestScore: float;
trainingBestParams: tpText;
confusionMatrix: tpText *NOT NULL*;
classes: tpText *NOT NULL*;
classificationReport: tpText *NOT NULL*;
trainScore: float *NOT NULL*;
testScore: float *NOT NULL*);

# Anexos C

# Carga y evolución del proyecto

La figura C.1 muestra el plan original de los módulos a desarrollar en el proyecto. El plan se dividía entre dos grandes grupos: los módulos software y los módulos hardware. El desarrollo de los módulos software son los que han tenido más peso en la carga de este proyecto con una ocupación estimada del 85 %. La carga en el diseño e implementación de los módulos hardware se ha llevado el resto de la ocupación con un valor estimado del 15 %. El proyecto se inició en Mayo del 2022 y se terminó en Julio del 2023, con un total de horas dedicadas de unas 500 horas.

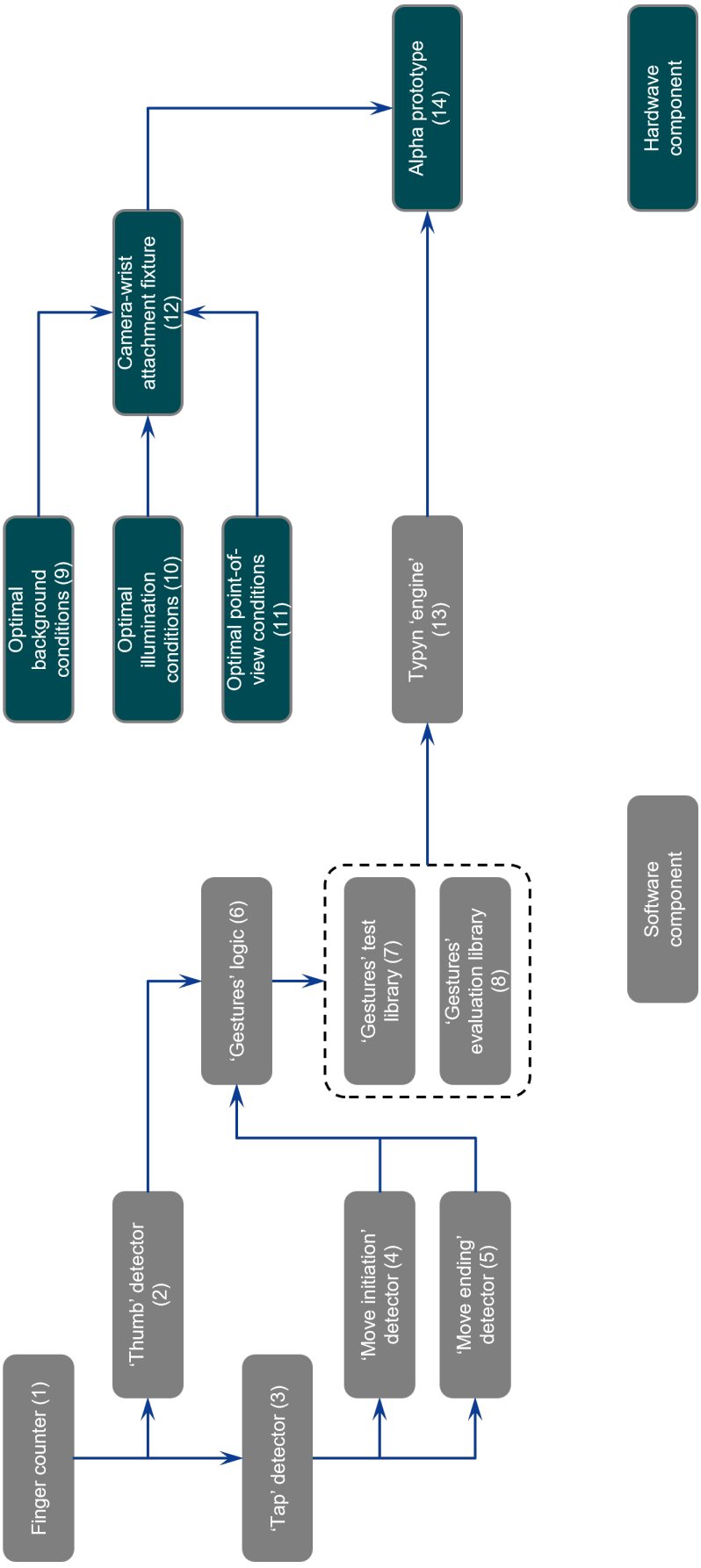


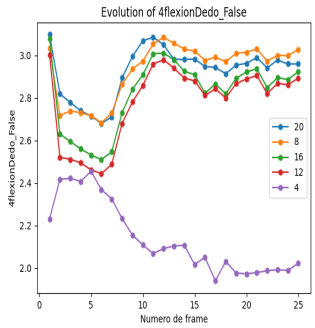

Figura C.1: Plan original de desarrollo del proyecto basado en módulos.

# Anexos D

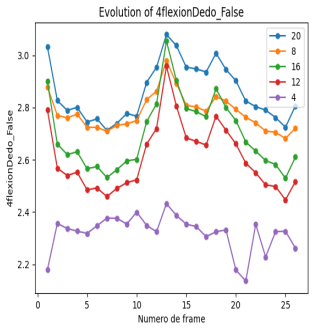
# Analisis

### Ángulo de la flexión del dedo

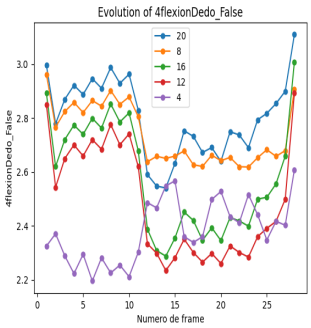
Observando la figura D.1 donde todos los dedos están estirados, se observa que, durante todo el vídeo y para todos los dedos, el ángulo formado es $>$ 2 radianes ($\approx 115°$) y su evolución puede variar bastante conforme avanza el movimiento, mientras que para los vídeos de la figura D.2 donde los dedos no están estirados (todos menos el índice a partir de un frame concreto), el ángulo formado en la mayoría de los casos está entorno a 0,5 radianes ($\approx 30°$) excepto para el pulgar que está entre 2 y 2,5 radianes



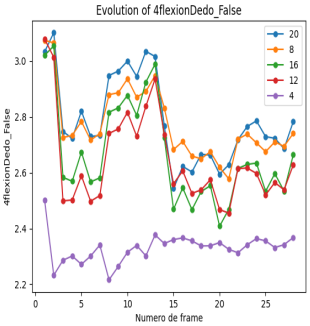

(a) Movimiento: Izquierda



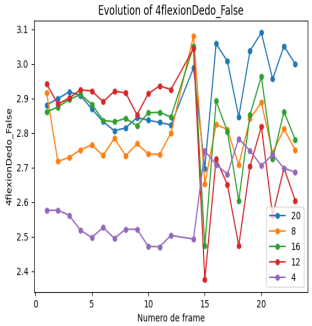

(b) Movimiento: Derecha




(c) Movimiento: Arriba

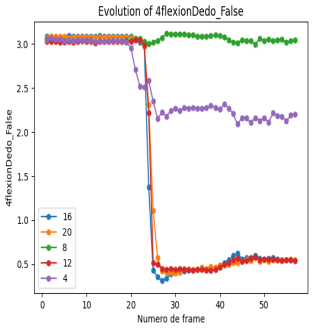



(d) Movimiento: Abajo

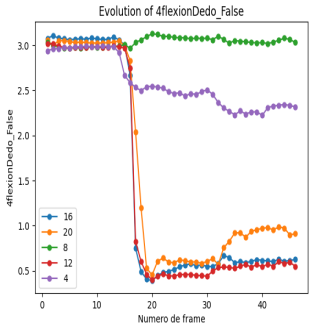



(e) Movimiento: Centro

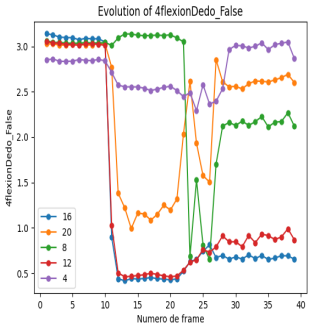
Figura D.1: Característica flexionDedo_False con todos los dedos estirados y sin iluminación



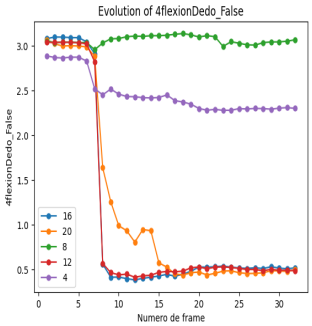

(a) Movimiento: Izquierda



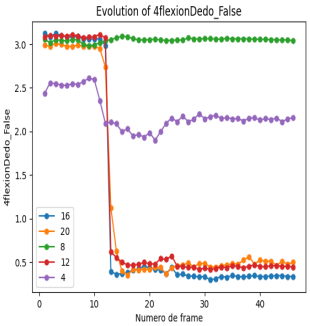

(b) Movimiento: Derecha




(c) Movimiento: Arriba




(d) Movimiento: Abajo




(e) Movimiento: Centro

Figura D.2: Característica flexionDedo_False con solo el índice estirado y con iluminación

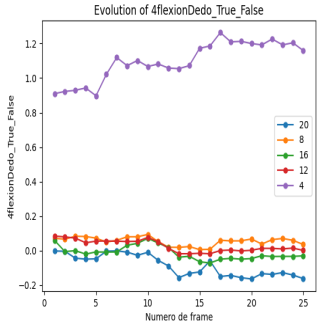
($\approx 115 - 145°$). Además se observa que el pulgar tiene un comportamiento diferente al del resto de los dedos, mientras que para el resto de los dedos esta característica puede parecer útil, para el pulgar puede parecer menos útil debido a la similitud en valores respecto al dedo "estirado" y "no estirado".

### Ángulo de la flexión del dedo en e plano XY

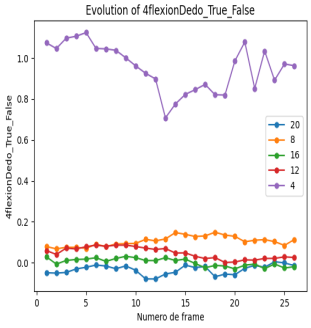
Observando las figuras D.3 y D.4 no se puede observar claramente que esta característica permita diferenciar entre "estirado" y "no estirado" aunque podría ser útil en combinación con otras características para diferenciar.

Observando las figuras D.5 se observa que todos los dedos menos el pulgar estando estirados presentan un valor superior a 2,5 radianes ($\approx 145°$) mientras que en la figura D.6 donde están no estirados presentan en su mayoría valores entorno a 0,5 radianes ($\approx 30°$) e inferiores a 1 radianes ($\approx 60°$). Esta característica parece útil para todos los dedos menos para el pulgar.

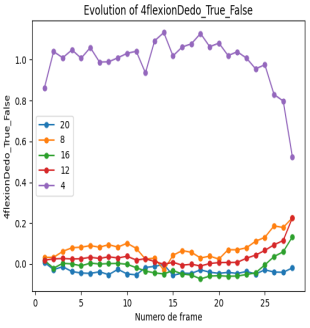
### Ángulos entre el dedo y el dedo índice



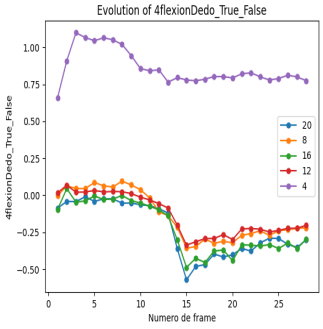

(a) Movimiento: Izquierda



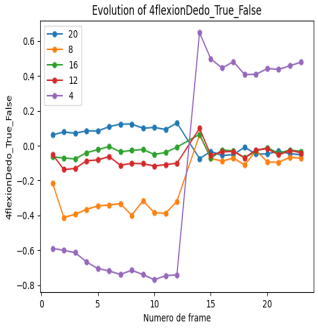

(b) Movimiento: Derecha




(c) Movimiento: Arriba

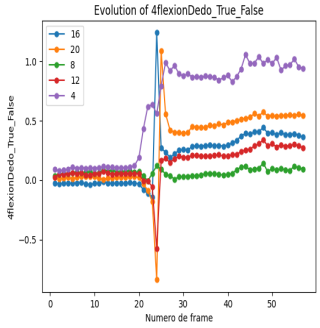



(d) Movimiento: Abajo

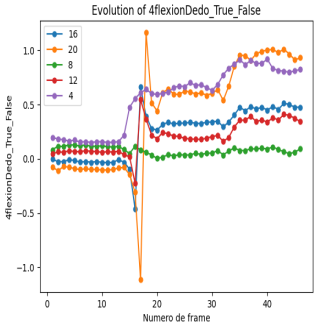



(e) Movimiento: Centro

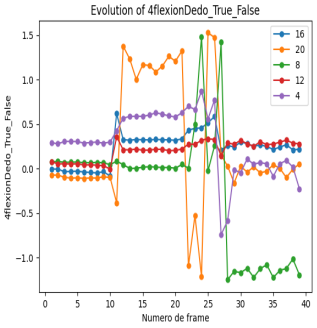
Figura D.3: Característica flexionDedo_True_False con todos los dedos estirados y sin iluminación



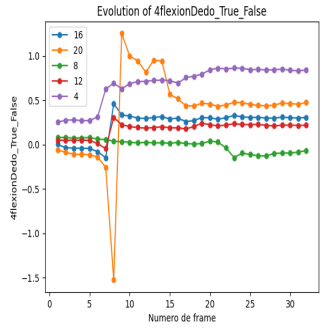

(a) Movimiento: Izquierda



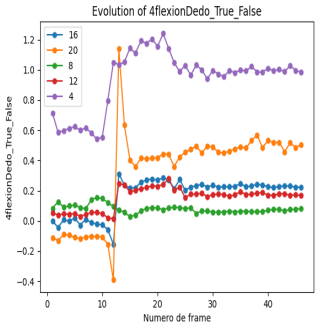

(b) Movimiento: Derecha




(c) Movimiento: Arriba

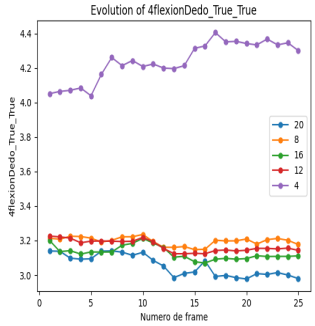



(d) Movimiento: Abajo

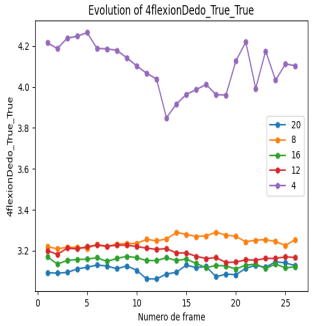



(e) Movimiento: Centro

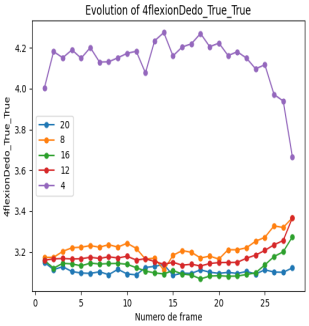
Figura D.4: Característica flexionDedo_True_False con solo el índice estirado y con iluminación



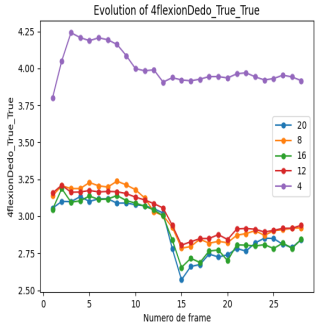

(a) Movimiento: Izquierda



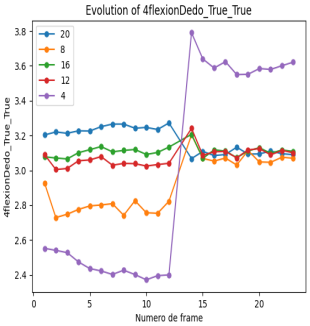

(b) Movimiento: Derecha




(c) Movimiento: Arriba

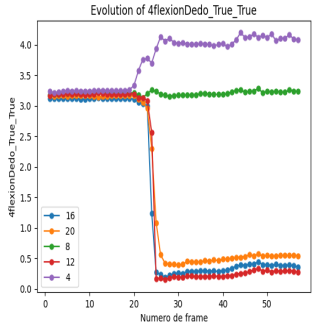



(d) Movimiento: Abajo

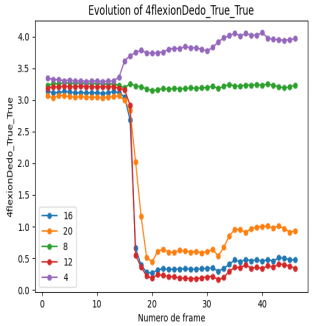



(e) Movimiento: Centro

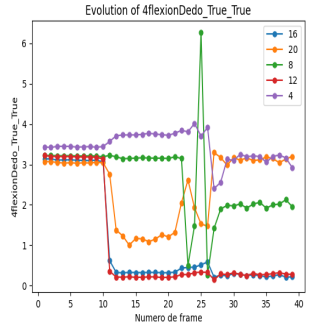
Figura D.5: Característica flexionDedo_True_True con todos los dedos estirados y sin iluminación



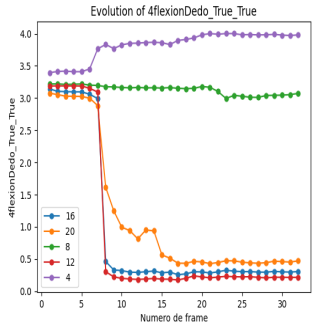

(a) Movimiento: Izquierda



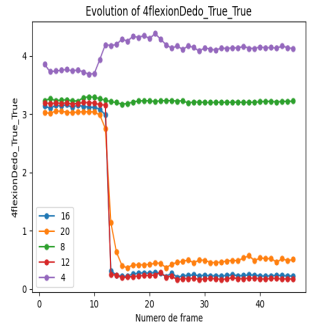

(b) Movimiento: Derecha




(c) Movimiento: Arriba

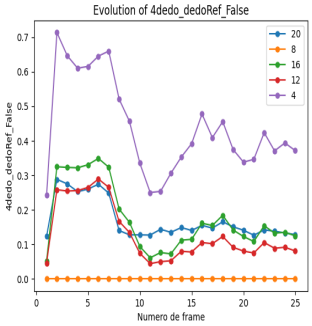



(d) Movimiento: Abajo

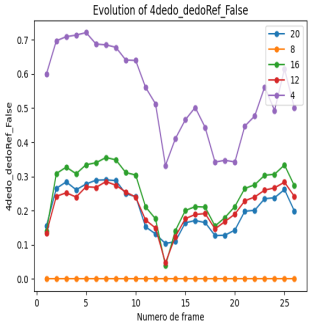



(e) Movimiento: Centro

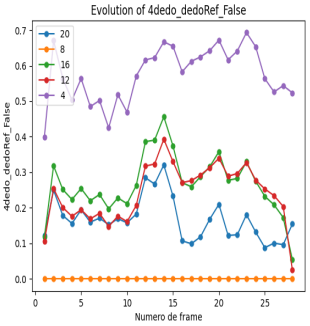
Figura D.6: Característica flexionDedo_True_True con solo el índice estirado y con iluminación



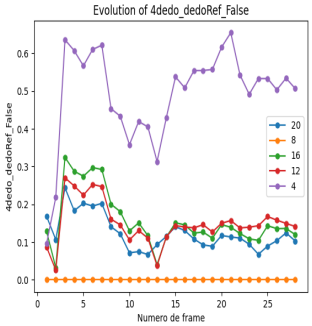

(a) Movimiento: Izquierda



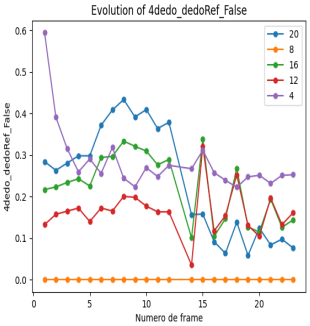

(b) Movimiento: Derecha




(c) Movimiento: Arriba

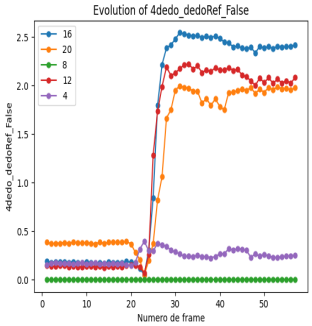



(d) Movimiento: Abajo

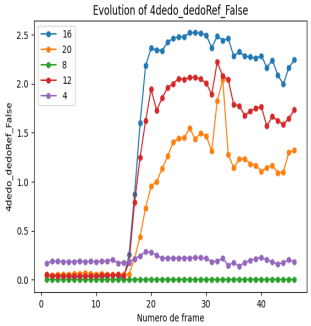



(e) Movimiento: Centro

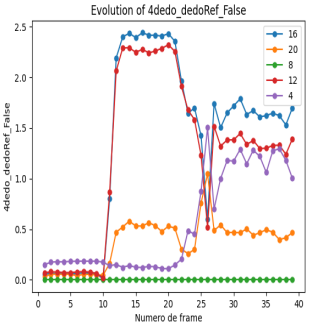
Figura D.7: Característica dedo_dedoRef_False con todos los dedos estirados y sin iluminación



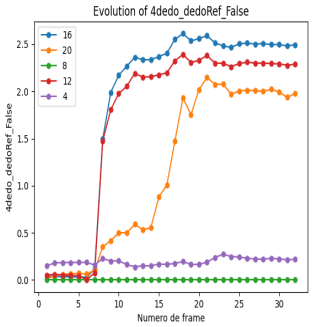

(a) Movimiento: Izquierda



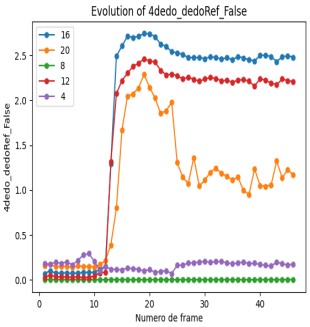

(b) Movimiento: Derecha




(c) Movimiento: Arriba




(d) Movimiento: Abajo




(e) Movimiento: Centro

Figura D.8: Característica dedo_dedoRef_False con solo el índice estirado y con iluminación

# Anexos E

# Pruebas

## E.1. Pulgar

### E.1.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-KNC | 0.98631 | 0.00476 |
| FS-FValue1-KNC | 0.98596 | 0.00429 |
| FS-MI1-KNC | 0.98596 | 0.00429 |
| FS-FValue0.5-KNC | 0.98596 | 0.00332 |
| FS-MI0.75-KNC | 0.98596 | 0.00430 |
| FS-MI0.5-KNC | 0.98525 | 0.00326 |
| ROS-FValue0.5-KNC | 0.98034 | 0.00071 |
| ROS-MI0.75-KNC | 0.98033 | 0.00491 |
| ROS-FValue1-KNC | 0.97963 | 0.00410 |
| ROS-MI1-KNC | 0.97963 | 0.00410 |
| ROS-FValue0.75-KNC | 0.97893 | 0.00315 |
| FS-MI0.25-KNC | 0.97858 | 0.00652 |
| ROS-MI0.5-KNC | 0.97717 | 0.00432 |
| ROS-MI0.25-KNC | 0.95541 | 0.00687 |
| FS-FValue0.25-KNC | 0.95363 | 0.02174 |
| RUS-MI0.75-KNC | 0.94522 | 0.00994 |
| FS-MI0.5-SGDC | 0.94453 | 0.01910 |
| ROS-FValue0.25-KNC | 0.94277 | 0.01410 |
| RUS-FValue0.75-KNC | 0.94171 | 0.00844 |
| RUS-FValue0.5-KNC | 0.94065 | 0.01465 |
| FS-FValue0.5-SGDC | 0.93997 | 0.01980 |
| FS-MI0.25-SGDC | 0.93995 | 0.02394 |
| RUS-MI0.5-KNC | 0.93960 | 0.00743 |
| FS-MI0.75-SGDC | 0.93960 | 0.02029 |
| RUS-FValue1-KNC | 0.93644 | 0.00876 |
| RUS-MI1-KNC | 0.93644 | 0.00876 |
| FS-FValue1-SGDC | 0.93609 | 0.01697 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-RC | 0.93574 | 0.00627 |
| FS-MI1-SGDC | 0.93539 | 0.02143 |
| FS-FValue1-RC | 0.93504 | 0.00678 |
| FS-MI1-RC | 0.93504 | 0.00678 |
| FS-MI0.75-RC | 0.93504 | 0.00912 |
| FS-FValue0.5-RC | 0.93434 | 0.00728 |
| FS-FValue0.75-SGDC | 0.93293 | 0.01180 |
| FS-MI0.5-RC | 0.92696 | 0.00608 |
| FS-FValue1-BNB | 0.92029 | 0.00942 |
| FS-MI1-BNB | 0.92029 | 0.00942 |
| FS-MI0.25-RC | 0.91327 | 0.00782 |
| ROS-MI0.75-SGDC | 0.91292 | 0.00626 |
| FS-FValue0.75-BNB | 0.91186 | 0.00916 |
| FS-MI0.75-BNB | 0.91081 | 0.00888 |
| FS-MI0.25-BNB | 0.90941 | 0.01100 |
| FS-FValue0.25-SGDC | 0.90519 | 0.02472 |
| FS-FValue0.25-RC | 0.90238 | 0.01290 |
| ROS-MI1-SGDC | 0.90028 | 0.01978 |
| RUS-MI0.25-KNC | 0.89501 | 0.02406 |
| ROS-FValue1-SGDC | 0.89221 | 0.02088 |
| FS-FValue0.5-BNB | 0.89186 | 0.01025 |
| FS-MI0.5-BNB | 0.89115 | 0.00715 |
| ROS-FValue1-BNB | 0.89080 | 0.00889 |
| ROS-MI1-BNB | 0.89080 | 0.00889 |
| ROS-MI0.25-SGDC | 0.88446 | 0.03908 |
| RUS-FValue0.25-KNC | 0.88413 | 0.01421 |
| RUS-FValue1-BNB | 0.88412 | 0.01258 |
| RUS-MI1-BNB | 0.88412 | 0.01258 |
| FS-FValue0.25-BNB | 0.88341 | 0.01907 |
| ROS-FValue0.75-SGDC | 0.88308 | 0.00886 |
| ROS-FValue1-NC | 0.88237 | 0.00934 |
| ROS-MI1-NC | 0.88237 | 0.00934 |
| ROS-MI0.75-RC | 0.88203 | 0.01344 |
| ROS-FValue0.75-RC | 0.88167 | 0.01567 |
| ROS-FValue1-RC | 0.88132 | 0.01310 |
| ROS-MI1-RC | 0.88132 | 0.01310 |
| ROS-FValue0.5-SGDC | 0.88062 | 0.01905 |
| ROS-FValue0.25-BNB | 0.87955 | 0.02383 |
| ROS-FValue0.75-NC | 0.87921 | 0.00974 |
| FS-FValue1-NC | 0.87921 | 0.00956 |
| FS-MI1-NC | 0.87921 | 0.00956 |
| ROS-MI0.5-SGDC | 0.87816 | 0.00975 |
| FS-FValue0.75-NC | 0.87781 | 0.01079 |
| RUS-FValue1-NC | 0.87780 | 0.01210 |
| RUS-MI1-NC | 0.87780 | 0.01210 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.75-NC | 0.87780 | 0.01195 |
| RUS-FValue1-RC | 0.87712 | 0.02042 |
| RUS-MI1-RC | 0.87712 | 0.02042 |
| RUS-MI0.75-BNB | 0.87570 | 0.00895 |
| RUS-MI0.75-SGDC | 0.87500 | 0.03371 |
| RUS-MI0.75-RC | 0.87115 | 0.01456 |
| RUS-FValue0.75-BNB | 0.87078 | 0.01205 |
| RUS-MI1-SGDC | 0.87046 | 0.04877 |
| RUS-MI0.75-NC | 0.86938 | 0.01239 |
| RUS-FValue0.75-RC | 0.86764 | 0.01833 |
| RUS-FValue0.75-SGDC | 0.86656 | 0.03510 |
| RUS-FValue0.5-SGDC | 0.86622 | 0.02043 |
| FS-MI0.75-NC | 0.86621 | 0.01531 |
| ROS-MI0.75-BNB | 0.86586 | 0.01545 |
| ROS-FValue0.5-RC | 0.86553 | 0.01701 |
| ROS-MI0.5-RC | 0.86202 | 0.02198 |
| FS-FValue0.5-NC | 0.86060 | 0.01081 |
| ROS-FValue0.75-BNB | 0.86060 | 0.01176 |
| RUS-FValue1-SGDC | 0.85743 | 0.03416 |
| RUS-MI0.5-BNB | 0.85567 | 0.03194 |
| RUS-MI0.5-RC | 0.85535 | 0.01692 |
| RUS-FValue0.5-RC | 0.85464 | 0.01544 |
| RUS-FValue0.5-BNB | 0.85463 | 0.01476 |
| ROS-MI0.75-NC | 0.85461 | 0.03092 |
| RUS-FValue0.5-NC | 0.85217 | 0.01539 |
| ROS-FValue0.5-NC | 0.85147 | 0.01333 |
| RUS-FValue0.25-BNB | 0.85147 | 0.03769 |
| ROS-MI0.25-RC | 0.85077 | 0.02108 |
| RUS-MI0.5-NC | 0.84866 | 0.01858 |
| FS-MI0.5-NC | 0.83989 | 0.01933 |
| ROS-MI0.5-NC | 0.83463 | 0.04789 |
| RUS-MI0.25-RC | 0.83428 | 0.02497 |
| ROS-MI0.5-BNB | 0.83043 | 0.04474 |
| FS-FValue0.25-NC | 0.82794 | 0.01989 |
| RUS-FValue0.25-NC | 0.82654 | 0.01701 |
| ROS-FValue0.25-RC | 0.82583 | 0.02757 |
| RUS-FValue0.25-SGDC | 0.82479 | 0.02178 |
| ROS-FValue0.25-NC | 0.82444 | 0.01167 |
| RUS-FValue0.25-RC | 0.82373 | 0.02800 |
| ROS-MI0.25-BNB | 0.82163 | 0.00699 |
| FS-MI0.25-NC | 0.81953 | 0.02172 |
| ROS-FValue0.5-BNB | 0.81529 | 0.05028 |
| ROS-FValue0.25-SGDC | 0.80793 | 0.03247 |
| RUS-MI0.5-SGDC | 0.80587 | 0.07047 |
| ROS-MI0.25-NC | 0.80233 | 0.01756 |
| RUS-MI0.25-BNB | 0.78687 | 0.01712 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.25-SGDC | 0.77846 | 0.10095 |
| RUS-MI0.25-NC | 0.77178 | 0.04050 |

### E.1.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.75-MLPC | 0.99017 | 0.00345 |
| ROS-MI1-MLPC | 0.98876 | 0.00179 |
| ROS-MI0.75-MLPC | 0.98876 | 0.00345 |
| FS-MI0.75-MLPC | 0.98806 | 0.00233 |
| ROS-MI0.75-RFC | 0.98806 | 0.00303 |
| FS-MI1-MLPC | 0.98771 | 0.00248 |
| ROS-FValue1-MLPC | 0.98736 | 0.00301 |
| ROS-FValue1-RFC | 0.98701 | 0.00179 |
| FS-FValue1-MLPC | 0.98666 | 0.00238 |
| FS-FValue1-RFC | 0.98631 | 0.00131 |
| FS-FValue0.75-MLPC | 0.98631 | 0.00281 |
| FS-MI0.75-RFC | 0.98595 | 0.00193 |
| ROS-MI1-RFC | 0.98595 | 0.00369 |
| FS-MI1-RFC | 0.98525 | 0.00180 |
| ROS-FValue1-SVC | 0.98490 | 0.00211 |
| ROS-MI1-SVC | 0.98490 | 0.00211 |
| ROS-MI0.75-SVC | 0.98490 | 0.00239 |
| ROS-MI0.5-RFC | 0.98455 | 0.00341 |
| ROS-FValue0.75-RFC | 0.98455 | 0.00407 |
| ROS-FValue0.75-SVC | 0.98455 | 0.00282 |
| ROS-FValue0.5-RFC | 0.98455 | 0.00451 |
| ROS-MI0.5-MLPC | 0.98454 | 0.00699 |
| FS-FValue1-SVC | 0.98420 | 0.00314 |
| FS-MI1-SVC | 0.98420 | 0.00314 |
| FS-FValue0.75-RFC | 0.98420 | 0.00248 |
| FS-MI0.75-SVC | 0.98385 | 0.00258 |
| FS-FValue0.75-SVC | 0.98385 | 0.00341 |
| FS-FValue0.5-SVC | 0.98350 | 0.00263 |
| FS-MI0.5-MLPC | 0.98350 | 0.00306 |
| FS-FValue0.5-MLPC | 0.98315 | 0.00238 |
| FS-FValue0.5-RFC | 0.98279 | 0.00436 |
| FS-MI0.75-GBC | 0.98244 | 0.00400 |
| FS-FValue1-GBC | 0.98209 | 0.00436 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI1-GBC | 0.98209 | 0.00436 |
| FS-MI0.5-RFC | 0.98209 | 0.00437 |
| ROS-FValue0.5-MLPC | 0.98104 | 0.00406 |
| FS-FValue0.75-GBC | 0.98034 | 0.00205 |
| ROS-MI1-BaggingClassifier | 0.98033 | 0.00550 |
| FS-MI0.5-SVC | 0.97998 | 0.00606 |
| ROS-FValue1-GBC | 0.97963 | 0.00946 |
| FS-FValue1-BaggingClassifier | 0.97928 | 0.00723 |
| ROS-MI1-GBC | 0.97928 | 0.00918 |
| ROS-MI0.75-GBC | 0.97928 | 0.01002 |
| FS-MI0.5-BaggingClassifier | 0.97893 | 0.00638 |
| ROS-FValue1-BaggingClassifier | 0.97893 | 0.00667 |
| FS-FValue0.5-BaggingClassifier | 0.97858 | 0.00697 |
| FS-MI0.5-GBC | 0.97858 | 0.00436 |
| FS-MI1-BaggingClassifier | 0.97858 | 0.00560 |
| ROS-FValue0.5-SVC | 0.97823 | 0.00239 |
| ROS-FValue0.75-GBC | 0.97823 | 0.00742 |
| ROS-MI0.5-SVC | 0.97823 | 0.00595 |
| ROS-MI0.75-BaggingClassifier | 0.97823 | 0.00529 |
| FS-MI0.75-DecisionTreeClassifier | 0.97647 | 0.00550 |
| FS-FValue0.5-GBC | 0.97577 | 0.00420 |
| FS-MI0.75-ABC | 0.97577 | 0.00475 |
| FS-MI0.75-BaggingClassifier | 0.97542 | 0.00497 |
| ROS-FValue0.75-BaggingClassifier | 0.97542 | 0.00963 |
| ROS-MI0.5-BaggingClassifier | 0.97472 | 0.00540 |
| FS-MI0.5-DecisionTreeClassifier | 0.97472 | 0.00454 |
| FS-FValue0.75-BaggingClassifier | 0.97437 | 0.00688 |
| FS-MI0.25-RFC | 0.97437 | 0.00572 |
| FS-MI1-DecisionTreeClassifier | 0.97331 | 0.00680 |
| FS-FValue1-ABC | 0.97296 | 0.00325 |
| FS-MI1-ABC | 0.97296 | 0.00325 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.5-DecisionTreeClassifier | 0.97261 | 0.00758 |
| ROS-MI0.5-GBC | 0.97261 | 0.00505 |
| FS-FValue1-DecisionTreeClassifier | 0.97226 | 0.00582 |
| ROS-FValue0.5-ExtraTreeClassifier | 0.97226 | 0.00952 |
| ROS-FValue0.5-BaggingClassifier | 0.97156 | 0.00924 |
| ROS-FValue1-DecisionTreeClassifier | 0.97155 | 0.00864 |
| FS-FValue0.75-ABC | 0.97121 | 0.01112 |
| ROS-FValue0.5-GBC | 0.97121 | 0.00827 |
| ROS-FValue0.75-DecisionTreeClassifier | 0.97120 | 0.01319 |
| ROS-MI0.5-DecisionTreeClassifier | 0.97051 | 0.00256 |
| FS-FValue0.75-DecisionTreeClassifier | 0.97050 | 0.00681 |
| ROS-MI1-DecisionTreeClassifier | 0.97015 | 0.01000 |
| FS-MI0.5-ABC | 0.96980 | 0.00612 |
| FS-MI1-ExtraTreeClassifier | 0.96980 | 0.00489 |
| ROS-MI0.75-DecisionTreeClassifier | 0.96980 | 0.01087 |
| FS-MI0.25-SVC | 0.96945 | 0.00790 |
| FS-FValue1-ExtraTreeClassifier | 0.96875 | 0.00449 |
| ROS-FValue0.75-ExtraTreeClassifier | 0.96805 | 0.00376 |
| FS-MI0.75-ExtraTreeClassifier | 0.96769 | 0.00838 |
| FS-FValue0.5-ExtraTreeClassifier | 0.96735 | 0.00892 |
| ROS-MI0.75-ExtraTreeClassifier | 0.96734 | 0.00814 |
| FS-MI0.25-GBC | 0.96734 | 0.00454 |
| FS-MI0.25-BaggingClassifier | 0.96734 | 0.00553 |
| FS-FValue0.75-ExtraTreeClassifier | 0.96700 | 0.00433 |
| ROS-FValue1-ExtraTreeClassifier | 0.96665 | 0.00291 |
| ROS-FValue1-ABC | 0.96594 | 0.00670 |
| ROS-MI1-ABC | 0.96594 | 0.00670 |
| ROS-FValue0.5-DecisionTreeClassifier | 0.96559 | 0.00560 |
| FS-MI0.25-DecisionTreeClassifier | 0.96524 | 0.00849 |
| ROS-FValue0.75-ABC | 0.96524 | 0.00680 |
| FS-MI0.5-ExtraTreeClassifier | 0.96454 | 0.01083 |
| ROS-MI0.75-ABC | 0.96454 | 0.00581 |
| ROS-MI1-ExtraTreeClassifier | 0.96453 | 0.00858 |
| FS-FValue0.5-ABC | 0.96349 | 0.01083 |
| FS-MI0.25-MLPC | 0.96313 | 0.00839 |
| RUS-FValue1-MLPC | 0.96173 | 0.00423 |
| ROS-MI0.5-ExtraTreeClassifier | 0.96067 | 0.00595 |
| ROS-MI0.25-RFC | 0.95961 | 0.01592 |
| ROS-MI0.25-BaggingClassifier | 0.95927 | 0.00795 |
| RUS-MI1-MLPC | 0.95856 | 0.00606 |
| ROS-MI0.25-DecisionTreeClassifier | 0.95752 | 0.00712 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.75-MLPC | 0.95716 | 0.00718 |
| RUS-MI0.75-MLPC | 0.95611 | 0.00498 |
| ROS-FValue0.25-RFC | 0.95611 | 0.00739 |
| ROS-MI0.25-ExtraTreeClassifier | 0.95576 | 0.00679 |
| FS-MI0.25-ExtraTreeClassifier | 0.95435 | 0.01494 |
| RUS-MI0.75-RFC | 0.95400 | 0.01130 |
| RUS-FValue1-RFC | 0.95225 | 0.00660 |
| ROS-MI0.25-GBC | 0.95190 | 0.00803 |
| RUS-FValue0.75-RFC | 0.95084 | 0.01005 |
| RUS-MI1-RFC | 0.95084 | 0.01082 |
| ROS-MI0.5-ABC | 0.95049 | 0.00927 |
| FS-FValue0.25-RFC | 0.95013 | 0.01884 |
| RUS-FValue0.5-MLPC | 0.94979 | 0.00894 |
| ROS-MI0.25-MLPC | 0.94874 | 0.00824 |
| FS-MI0.25-ABC | 0.94838 | 0.00308 |
| ROS-FValue0.25-BaggingClassifier | 0.94698 | 0.00906 |
| FS-FValue0.25-BaggingClassifier | 0.94556 | 0.01966 |
| ROS-FValue0.25-DecisionTreeClassifier | 0.94487 | 0.00954 |
| RUS-MI0.75-SVC | 0.94276 | 0.01191 |
| RUS-FValue1-SVC | 0.94136 | 0.00907 |
| RUS-MI1-SVC | 0.94136 | 0.00907 |
| RUS-MI0.5-MLPC | 0.94101 | 0.00597 |
| RUS-MI0.5-RFC | 0.94066 | 0.00802 |
| FS-FValue0.25-DecisionTreeClassifier | 0.94029 | 0.02525 |
| RUS-MI0.75-ABC | 0.93996 | 0.00910 |
| FS-FValue0.25-SVC | 0.93959 | 0.02070 |
| RUS-FValue0.5-SVC | 0.93925 | 0.01706 |
| ROS-FValue0.5-ABC | 0.93891 | 0.00659 |
| RUS-MI0.75-GBC | 0.93855 | 0.00999 |
| RUS-FValue0.75-GBC | 0.93820 | 0.01050 |
| FS-FValue0.25-GBC | 0.93818 | 0.02230 |
| FS-FValue0.25-MLPC | 0.93818 | 0.02688 |
| RUS-FValue0.75-SVC | 0.93785 | 0.00728 |
| RUS-MI1-GBC | 0.93714 | 0.01298 |
| RUS-FValue0.5-RFC | 0.93680 | 0.01435 |
| RUS-FValue1-ABC | 0.93645 | 0.00610 |
| RUS-MI1-ABC | 0.93645 | 0.00610 |
| RUS-FValue1-GBC | 0.93574 | 0.01256 |
| ROS-FValue0.25-GBC | 0.93188 | 0.01695 |
| RUS-FValue0.5-GBC | 0.93083 | 0.01724 |
| RUS-FValue0.75-ABC | 0.93048 | 0.00835 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.5-SVC | 0.92838 | 0.00859 |
| ROS-FValue0.25-MLPC | 0.92837 | 0.01210 |
| RUS-MI0.5-ABC | 0.92802 | 0.01298 |
| ROS-FValue0.25-ExtraTreeClassifier | 0.92732 | 0.01087 |
| RUS-MI0.75-BaggingClassifier | 0.92696 | 0.00818 |
| FS-FValue0.25-ExtraTreeClassifier | 0.92625 | 0.02549 |
| RUS-MI0.5-GBC | 0.92591 | 0.00340 |
| RUS-FValue0.75-BaggingClassifier | 0.92346 | 0.00925 |
| RUS-FValue1-BaggingClassifier | 0.92239 | 0.01008 |
| RUS-MI0.5-BaggingClassifier | 0.91959 | 0.00840 |
| FS-FValue0.25-ABC | 0.91852 | 0.02308 |
| RUS-FValue0.5-BaggingClassifier | 0.91819 | 0.00526 |
| RUS-FValue1-ExtraTreeClassifier | 0.91749 | 0.00530 |
| RUS-FValue1-DecisionTreeClassifier | 0.91502 | 0.02453 |
| RUS-FValue0.5-DecisionTreeClassifier | 0.91502 | 0.01586 |
| RUS-FValue0.5-ExtraTreeClassifier | 0.91363 | 0.01859 |
| ROS-FValue0.25-SVC | 0.91362 | 0.01380 |
| RUS-MI0.5-DecisionTreeClassifier | 0.91292 | 0.01853 |
| ROS-MI0.25-SVC | 0.91220 | 0.03189 |
| RUS-MI1-BaggingClassifier | 0.91151 | 0.01959 |
| RUS-FValue0.75-DecisionTreeClassifier | 0.91116 | 0.02557 |
| RUS-MI1-DecisionTreeClassifier | 0.91081 | 0.02145 |
| RUS-MI0.75-DecisionTreeClassifier | 0.91081 | 0.01936 |
| ROS-MI0.25-ABC | 0.91047 | 0.00723 |
| RUS-MI1-ExtraTreeClassifier | 0.91010 | 0.02458 |
| RUS-FValue0.75-ExtraTreeClassifier | 0.90976 | 0.01197 |
| RUS-MI0.5-ExtraTreeClassifier | 0.90941 | 0.01473 |
| RUS-FValue0.5-ABC | 0.90660 | 0.01341 |
| RUS-MI0.25-RFC | 0.90484 | 0.02065 |
| RUS-MI0.75-ExtraTreeClassifier | 0.90310 | 0.01044 |
| RUS-MI0.25-DecisionTreeClassifier | 0.90203 | 0.02318 |
| RUS-MI0.25-BaggingClassifier | 0.89993 | 0.01523 |
| ROS-FValue0.25-ABC | 0.89712 | 0.01270 |
| RUS-MI0.25-GBC | 0.89571 | 0.01826 |
| RUS-MI0.25-ABC | 0.89150 | 0.01536 |
| RUS-FValue0.25-RFC | 0.88974 | 0.02061 |
| RUS-FValue0.25-GBC | 0.88869 | 0.01765 |
| RUS-MI0.25-MLPC | 0.88589 | 0.01663 |
| RUS-MI0.25-ExtraTreeClassifier | 0.88202 | 0.02078 |
| RUS-FValue0.25-DecisionTreeClassifier | 0.88132 | 0.01912 |
| RUS-FValue0.25-BaggingClassifier | 0.87710 | 0.01036 |
| RUS-MI0.25-SVC | 0.87289 | 0.02213 |
| RUS-FValue0.25-MLPC | 0.87289 | 0.02317 |
| RUS-FValue0.25-ExtraTreeClassifier | 0.86868 | 0.02446 |
| RUS-FValue0.25-SVC | 0.86587 | 0.02080 |
| RUS-FValue0.25-ABC | 0.86131 | 0.01894 |

### E.1.3. Prueba 3

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.75-MLPC | 0.98841 | 0.00237 |
| ROS-SFS0.5-MLPC | 0.98841 | 0.00179 |
| ROS-SBS0.75-MLPC | 0.98806 | 0.00321 |
| ROS-SFS0.5-RFC | 0.98806 | 0.00561 |
| ROS-SBS1-MLPC | 0.98701 | 0.00210 |
| ROS-SFS1-MLPC | 0.98701 | 0.00086 |
| ROS-SBS1-RFC | 0.98701 | 0.00453 |
| ROS-SFS1-RFC | 0.98701 | 0.00379 |
| ROS-SFS0.75-RFC | 0.98666 | 0.00286 |
| ROS-SBS0.75-RFC | 0.98490 | 0.00345 |
| ROS-SFS0.5-KNC | 0.98174 | 0.00394 |
| ROS-SFS0.75-KNC | 0.98034 | 0.00282 |
| ROS-SFS0.75-GBC | 0.97999 | 0.00892 |
| ROS-SBS1-KNC | 0.97963 | 0.00410 |
| ROS-SFS1-KNC | 0.97963 | 0.00410 |
| ROS-SBS1-GBC | 0.97963 | 0.00946 |
| ROS-SFS1-GBC | 0.97963 | 0.00946 |
| ROS-SBS0.5-RFC | 0.97893 | 0.00577 |
| ROS-SBS0.75-GBC | 0.97823 | 0.00681 |
| ROS-SBS0.5-MLPC | 0.97753 | 0.00611 |
| ROS-SFS0.25-RFC | 0.97507 | 0.00725 |
| ROS-SBS0.75-KNC | 0.97472 | 0.00624 |
| ROS-SFS0.5-GBC | 0.97331 | 0.00764 |
| ROS-SBS0.5-KNC | 0.97227 | 0.00746 |
| ROS-SBS0.25-RFC | 0.96910 | 0.01142 |
| ROS-SFS0.25-KNC | 0.96769 | 0.01149 |
| ROS-SBS0.5-GBC | 0.96630 | 0.00721 |
| ROS-SFS0.25-MLPC | 0.96102 | 0.01294 |
| ROS-SFS0.25-GBC | 0.95435 | 0.01348 |
| ROS-SBS0.25-KNC | 0.95225 | 0.01204 |
| ROS-SBS0.25-GBC | 0.94172 | 0.01494 |
| ROS-SBS0.25-MLPC | 0.94101 | 0.02094 |

## E.2. Indice

### E.2.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-KNC | 0.96594 | 0.00927 |
| FS-MI0.5-KNC | 0.95927 | 0.00951 |
| ROS-FValue0.75-KNC | 0.95927 | 0.01130 |
| FS-FValue0.5-KNC | 0.95822 | 0.01027 |
| FS-MI0.75-KNC | 0.95365 | 0.01148 |
| ROS-MI0.75-KNC | 0.95259 | 0.01367 |
| FS-FValue1-KNC | 0.94944 | 0.01120 |
| FS-MI1-KNC | 0.94944 | 0.01120 |
| ROS-MI0.5-KNC | 0.94839 | 0.00906 |
| ROS-FValue0.5-KNC | 0.94803 | 0.01318 |
| RUS-FValue0.75-KNC | 0.94628 | 0.01154 |
| RUS-MI0.5-KNC | 0.93961 | 0.01302 |
| RUS-FValue0.5-KNC | 0.93961 | 0.01136 |
| RUS-MI0.75-KNC | 0.93680 | 0.01176 |
| RUS-MI0.25-KNC | 0.93610 | 0.00689 |
| ROS-FValue1-KNC | 0.93398 | 0.01463 |
| ROS-MI1-KNC | 0.93398 | 0.01463 |
| ROS-MI0.25-KNC | 0.93083 | 0.01794 |
| RUS-FValue1-KNC | 0.92942 | 0.00651 |
| RUS-MI1-KNC | 0.92942 | 0.00651 |
| FS-FValue0.25-KNC | 0.92802 | 0.00608 |
| FS-MI0.25-KNC | 0.92767 | 0.01467 |
| RUS-FValue0.25-KNC | 0.92452 | 0.00943 |
| FS-FValue0.75-RC | 0.92135 | 0.01245 |
| ROS-FValue0.25-KNC | 0.92100 | 0.01187 |
| FS-FValue1-RC | 0.92030 | 0.01373 |
| FS-MI1-RC | 0.92030 | 0.01373 |
| FS-MI0.75-RC | 0.92030 | 0.01160 |
| FS-MI1-SGDC | 0.91678 | 0.01145 |
| FS-FValue0.5-BNB | 0.91328 | 0.01235 |
| FS-MI0.5-BNB | 0.91293 | 0.01323 |
| FS-FValue0.25-BNB | 0.91258 | 0.01524 |
| FS-MI0.75-SGDC | 0.91187 | 0.01984 |
| FS-MI0.25-BNB | 0.91152 | 0.01069 |
| FS-FValue0.25-NC | 0.91118 | 0.02189 |
| FS-FValue1-SGDC | 0.91012 | 0.00985 |
| FS-MI0.5-NC | 0.90977 | 0.01189 |
| ROS-MI0.75-SGDC | 0.90837 | 0.01727 |
| FS-MI0.75-BNB | 0.90661 | 0.01673 |
| FS-FValue0.75-BNB | 0.90661 | 0.01492 |
| FS-FValue1-BNB | 0.90626 | 0.01648 |
| FS-MI1-BNB | 0.90626 | 0.01648 |
| FS-FValue0.5-NC | 0.90591 | 0.01853 |
| ROS-MI0.75-BNB | 0.90591 | 0.01445 |
| ROS-FValue1-BNB | 0.90591 | 0.01420 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI1-BNB | 0.90591 | 0.01420 |
| RUS-FValue1-BNB | 0.90556 | 0.01482 |
| RUS-MI1-BNB | 0.90556 | 0.01482 |
| ROS-MI0.5-NC | 0.90555 | 0.01410 |
| FS-MI0.75-NC | 0.90485 | 0.01710 |
| RUS-FValue1-RC | 0.90450 | 0.01577 |
| RUS-MI1-RC | 0.90450 | 0.01577 |
| RUS-MI0.75-BNB | 0.90415 | 0.01381 |
| FS-FValue0.25-RC | 0.90346 | 0.02988 |
| RUS-MI0.5-NC | 0.90345 | 0.01555 |
| ROS-MI0.5-BNB | 0.90345 | 0.00788 |
| ROS-FValue0.75-BNB | 0.90310 | 0.01344 |
| ROS-FValue0.5-NC | 0.90275 | 0.01765 |
| RUS-FValue0.5-NC | 0.90275 | 0.01668 |
| RUS-MI0.75-RC | 0.90274 | 0.01578 |
| FS-FValue1-NC | 0.90240 | 0.01822 |
| FS-MI1-NC | 0.90240 | 0.01822 |
| RUS-FValue0.75-BNB | 0.90240 | 0.01504 |
| RUS-FValue0.5-BNB | 0.90204 | 0.01401 |
| ROS-FValue0.5-BNB | 0.90204 | 0.01388 |
| ROS-MI0.25-BNB | 0.90169 | 0.01156 |
| RUS-MI0.75-NC | 0.90134 | 0.01945 |
| FS-FValue0.75-NC | 0.90134 | 0.01885 |
| RUS-MI0.5-BNB | 0.90134 | 0.00797 |
| ROS-MI0.75-NC | 0.90099 | 0.01943 |
| ROS-FValue1-RC | 0.89994 | 0.01642 |
| ROS-MI1-RC | 0.89994 | 0.01642 |
| RUS-FValue0.75-RC | 0.89993 | 0.01499 |
| ROS-FValue0.75-SGDC | 0.89924 | 0.02279 |
| ROS-MI0.75-RC | 0.89888 | 0.01782 |
| FS-FValue0.75-SGDC | 0.89888 | 0.00878 |
| FS-FValue0.5-RC | 0.89853 | 0.01618 |
| ROS-FValue1-SGDC | 0.89783 | 0.00892 |
| RUS-FValue0.75-NC | 0.89748 | 0.01927 |
| RUS-FValue1-NC | 0.89748 | 0.01865 |
| RUS-MI1-NC | 0.89748 | 0.01865 |
| ROS-FValue0.75-NC | 0.89643 | 0.01922 |
| ROS-MI0.25-NC | 0.89642 | 0.01255 |
| ROS-FValue1-NC | 0.89608 | 0.01933 |
| ROS-MI1-NC | 0.89608 | 0.01933 |
| ROS-FValue0.75-RC | 0.89572 | 0.01552 |
| ROS-MI1-SGDC | 0.89571 | 0.01937 |
| FS-MI0.5-RC | 0.89467 | 0.01805 |
| RUS-MI1-SGDC | 0.89291 | 0.01819 |
| RUS-FValue0.75-SGDC | 0.89257 | 0.02080 |
| ROS-MI0.5-SGDC | 0.89116 | 0.02379 |
| FS-FValue0.5-SGDC | 0.88906 | 0.02136 |
| ROS-MI0.25-SGDC | 0.88870 | 0.02454 |
| RUS-MI0.25-NC | 0.88800 | 0.01353 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.25-SGDC | 0.88660 | 0.02054 |
| RUS-FValue1-SGDC | 0.88554 | 0.01673 |
| FS-MI0.5-SGDC | 0.88412 | 0.01604 |
| RUS-MI0.25-BNB | 0.88344 | 0.03178 |
| ROS-FValue0.5-RC | 0.88344 | 0.01886 |
| RUS-FValue0.5-RC | 0.88274 | 0.01996 |
| FS-MI0.25-NC | 0.88132 | 0.00576 |
| ROS-FValue0.5-SGDC | 0.88097 | 0.01442 |
| ROS-MI0.5-RC | 0.87958 | 0.02049 |
| RUS-MI0.5-RC | 0.87958 | 0.02125 |
| RUS-MI0.5-SGDC | 0.87922 | 0.01226 |
| RUS-FValue0.25-SGDC | 0.87676 | 0.01188 |
| RUS-MI0.25-RC | 0.87430 | 0.01701 |
| ROS-MI0.25-RC | 0.87396 | 0.02188 |
| RUS-FValue0.5-SGDC | 0.87360 | 0.01821 |
| FS-MI0.25-SGDC | 0.87359 | 0.00999 |
| RUS-FValue0.25-RC | 0.87290 | 0.01978 |
| RUS-FValue0.25-NC | 0.87290 | 0.01567 |
| ROS-FValue0.25-RC | 0.86834 | 0.02186 |
| ROS-FValue0.25-SGDC | 0.86624 | 0.02774 |
| ROS-FValue0.25-NC | 0.86623 | 0.02148 |
| RUS-MI0.75-SGDC | 0.86305 | 0.03903 |
| FS-FValue0.25-SGDC | 0.85708 | 0.01776 |
| FS-MI0.25-RC | 0.85393 | 0.00653 |
| RUS-FValue0.25-BNB | 0.84446 | 0.02025 |
| ROS-FValue0.25-BNB | 0.83427 | 0.02224 |

### E.2.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0,75-MLPC | 0.96980 | 0.00582 |
| ROS-FValue0,75-RFC | 0.96769 | 0.01121 |
| ROS-FValue0,75-MLPC | 0.96594 | 0.00552 |
| FS-FValue0,75-RFC | 0.96524 | 0.00933 |
| ROS-FValue1-MLPC | 0.96488 | 0.00911 |
| FS-MI1-MLPC | 0.96418 | 0.00866 |
| FS-MI1-RFC | 0.96418 | 0.00974 |
| ROS-MI1-RFC | 0.96418 | 0.01159 |
| FS-FValue1-MLPC | 0.96278 | 0.00643 |
| ROS-MI0,75-RFC | 0.96243 | 0.00743 |
| FS-MI0,75-MLPC | 0.96243 | 0.01148 |
| FS-MI0,75-RFC | 0.96208 | 0.01115 |
| ROS-FValue1-RFC | 0.96207 | 0.01453 |
| ROS-FValue0,75-GBC | 0.96137 | 0.01330 |
| FS-FValue1-RFC | 0.96137 | 0.01129 |
| FS-FValue0,75-GBC | 0.96102 | 0.00623 |
| ROS-MI0,75-MLPC | 0.96067 | 0.01185 |
| ROS-FValue0,75-BaggingClassifier | 0.96067 | 0.00999 |
| ROS-MI1-MLPC | 0.95997 | 0.00759 |
| FS-MI1-GBC | 0.95927 | 0.00748 |
| ROS-MI0,5-RFC | 0.95927 | 0.00978 |
| FS-FValue0,75-BaggingClassifier | 0.95892 | 0.01136 |
| FS-FValue1-GBC | 0.95892 | 0.00742 |
| ROS-MI1-GBC | 0.95892 | 0.01241 |
| FS-MI0,5-RFC | 0.95857 | 0.00985 |
| ROS-FValue1-GBC | 0.95857 | 0.01241 |
| ROS-FValue0,5-RFC | 0.95751 | 0.00886 |
| ROS-MI1-BaggingClassifier | 0.95716 | 0.01743 |
| FS-FValue1-BaggingClassifier | 0.95681 | 0.01455 |
| FS-MI1-BaggingClassifier | 0.95646 | 0.01057 |
| ROS-FValue0,5-BaggingClassifier | 0.95611 | 0.00961 |
| FS-FValue0,5-BaggingClassifier | 0.95576 | 0.00937 |
| FS-FValue0,5-RFC | 0.95576 | 0.00891 |
| ROS-FValue0,75-SVC | 0.95576 | 0.01204 |
| ROS-FValue1-BaggingClassifier | 0.95575 | 0.01551 |
| ROS-MI0,75-GBC | 0.95541 | 0.01251 |
| FS-FValue0,75-SVC | 0.95506 | 0.01096 |
| ROS-MI0,5-MLPC | 0.95506 | 0.00892 |
| FS-FValue0,5-GBC | 0.95471 | 0.00989 |
| FS-MI0,75-BaggingClassifier | 0.95470 | 0.01382 |
| FS-MI0,75-GBC | 0.95400 | 0.00972 |
| FS-FValue0,5-MLPC | 0.95400 | 0.00952 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0,75-RFC | 0.95400 | 0.00997 |
| ROS-FValue0,5-MLPC | 0.95330 | 0.00740 |
| RUS-MI1-RFC | 0.95295 | 0.01067 |
| ROS-MI0,75-BaggingClassifier | 0.95295 | 0.01623 |
| ROS-MI0,5-GBC | 0.95260 | 0.01397 |
| ROS-MI0,5-BaggingClassifier | 0.95224 | 0.01205 |
| RUS-FValue0,75-MLPC | 0.95224 | 0.01190 |
| FS-MI0,5-BaggingClassifier | 0.95190 | 0.01147 |
| FS-MI0,5-GBC | 0.95155 | 0.01044 |
| RUS-FValue1-RFC | 0.95154 | 0.01189 |
| ROS-FValue0,5-GBC | 0.95120 | 0.01287 |
| FS-MI0,5-MLPC | 0.95084 | 0.01180 |
| RUS-MI0,75-BaggingClassifier | 0.95084 | 0.01129 |
| ROS-MI0,25-RFC | 0.95049 | 0.01495 |
| FS-FValue0,75-ExtraTreeClassifier | 0.95049 | 0.00372 |
| RUS-MI0,75-GBC | 0.95049 | 0.00849 |
| RUS-FValue1-GBC | 0.95014 | 0.00954 |
| RUS-MI0,75-RFC | 0.94979 | 0.01080 |
| RUS-FValue0,75-GBC | 0.94979 | 0.00973 |
| RUS-MI1-GBC | 0.94944 | 0.01092 |
| FS-MI0,75-ExtraTreeClassifier | 0.94909 | 0.00674 |
| RUS-FValue0,75-BaggingClassifier | 0.94874 | 0.01072 |
| RUS-FValue0,5-RFC | 0.94874 | 0.00697 |
| RUS-MI0,5-RFC | 0.94839 | 0.00939 |
| ROS-MI0,75-SVC | 0.94803 | 0.01070 |
| RUS-MI1-BaggingClassifier | 0.94768 | 0.01307 |
| RUS-FValue0,5-BaggingClassifier | 0.94733 | 0.00809 |
| RUS-MI0,75-MLPC | 0.94733 | 0.01191 |
| FS-FValue1-SVC | 0.94628 | 0.00966 |
| FS-MI1-SVC | 0.94628 | 0.00966 |
| ROS-FValue0,75-DecisionTreeClassifier | 0.94593 | 0.01453 |
| ROS-FValue1-SVC | 0.94557 | 0.01645 |
| ROS-MI1-SVC | 0.94557 | 0.01645 |
| FS-MI0,75-SVC | 0.94523 | 0.01073 |
| RUS-MI0,5-BaggingClassifier | 0.94452 | 0.00750 |
| RUS-FValue0,5-GBC | 0.94452 | 0.01331 |
| ROS-MI0,25-BaggingClassifier | 0.94418 | 0.01481 |
| RUS-MI0,5-GBC | 0.94417 | 0.01141 |
| ROS-MI0,5-DecisionTreeClassifier | 0.94417 | 0.00913 |
| FS-FValue0,75-DecisionTreeClassifier | 0.94347 | 0.00729 |
| RUS-FValue1-BaggingClassifier | 0.94242 | 0.01112 |
| RUS-FValue1-MLPC | 0.94241 | 0.01464 |
| ROS-FValue0,75-ExtraTreeClassifier | 0.94207 | 0.01301 |
| FS-FValue0,5-SVC | 0.94207 | 0.01663 |
| ROS-FValue0,5-DecisionTreeClassifier | 0.94171 | 0.00642 |
| ROS-FValue0,25-RFC | 0.94137 | 0.01166 |
| RUS-FValue0,75-SVC | 0.94137 | 0.01584 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI1-ExtraTreeClassifier | 0.94136 | 0.00905 |
| RUS-MI1-MLPC | 0.94136 | 0.01230 |
| FS-FValue0,5-ExtraTreeClassifier | 0.94136 | 0.00824 |
| ROS-FValue1-DecisionTreeClassifier | 0.94102 | 0.01696 |
| RUS-MI0,5-DecisionTreeClassifier | 0.94066 | 0.00677 |
| FS-FValue0,5-DecisionTreeClassifier | 0.94032 | 0.01102 |
| ROS-MI0,25-GBC | 0.94031 | 0.01484 |
| ROS-MI1-DecisionTreeClassifier | 0.94031 | 0.01800 |
| RUS-MI0,25-RFC | 0.94031 | 0.00910 |
| FS-MI0,5-SVC | 0.93996 | 0.01765 |
| RUS-MI0,75-SVC | 0.93996 | 0.01655 |
| ROS-FValue0,75-ABC | 0.93961 | 0.00862 |
| RUS-MI0,25-GBC | 0.93960 | 0.01104 |
| FS-FValue1-ABC | 0.93856 | 0.01090 |
| FS-MI1-ABC | 0.93856 | 0.01090 |
| ROS-MI0,75-ExtraTreeClassifier | 0.93855 | 0.01430 |
| ROS-FValue0,5-ExtraTreeClassifier | 0.93820 | 0.00797 |
| RUS-FValue1-SVC | 0.93786 | 0.01507 |
| RUS-MI1-SVC | 0.93786 | 0.01507 |
| RUS-MI0,25-BaggingClassifier | 0.93785 | 0.00752 |
| ROS-FValue0,25-BaggingClassifier | 0.93751 | 0.01481 |
| FS-MI0,75-DecisionTreeClassifier | 0.93750 | 0.00912 |
| FS-MI0,5-DecisionTreeClassifier | 0.93750 | 0.01513 |
| FS-MI0,5-ExtraTreeClassifier | 0.93750 | 0.01046 |
| FS-FValue0,75-ABC | 0.93716 | 0.01471 |
| FS-FValue1-ExtraTreeClassifier | 0.93715 | 0.00917 |
| ROS-FValue1-ExtraTreeClassifier | 0.93715 | 0.00613 |
| FS-MI0,75-ABC | 0.93680 | 0.00871 |
| FS-FValue1-DecisionTreeClassifier | 0.93645 | 0.01094 |
| ROS-FValue1-ABC | 0.93575 | 0.00925 |
| ROS-MI1-ABC | 0.93575 | 0.00925 |
| FS-MI1-DecisionTreeClassifier | 0.93574 | 0.00750 |
| ROS-MI1-ExtraTreeClassifier | 0.93574 | 0.00479 |
| ROS-MI0,5-SVC | 0.93540 | 0.01573 |
| RUS-MI0,5-MLPC | 0.93469 | 0.01267 |
| RUS-FValue1-ABC | 0.93399 | 0.01426 |
| RUS-MI1-ABC | 0.93399 | 0.01426 |
| ROS-MI0,75-ABC | 0.93399 | 0.00476 |
| ROS-MI0,25-DecisionTreeClassifier | 0.93364 | 0.01804 |
| FS-MI0,25-RFC | 0.93364 | 0.01000 |
| RUS-FValue0,75-DecisionTreeClassifier | 0.93329 | 0.01289 |
| ROS-FValue0,5-SVC | 0.93294 | 0.01017 |
| RUS-MI0,5-ABC | 0.93294 | 0.00612 |
| ROS-FValue0,25-DecisionTreeClassifier | 0.93224 | 0.01439 |
| RUS-FValue0,75-ABC | 0.93188 | 0.01032 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0,5-DecisionTreeClassifier | 0.93188 | 0.00717 |
| ROS-MI0,25-ExtraTreeClassifier | 0.93119 | 0.02382 |
| RUS-MI0,75-ABC | 0.93083 | 0.00625 |
| ROS-MI0,75-DecisionTreeClassifier | 0.93048 | 0.01372 |
| FS-FValue0,25-RFC | 0.92907 | 0.01083 |
| FS-MI0,5-ABC | 0.92873 | 0.01604 |
| RUS-MI0,75-DecisionTreeClassifier | 0.92837 | 0.00994 |
| FS-MI0,25-GBC | 0.92802 | 0.01046 |
| ROS-MI0,5-ABC | 0.92802 | 0.00859 |
| ROS-MI0,25-MLPC | 0.92767 | 0.01797 |
| FS-FValue0,25-GBC | 0.92767 | 0.01127 |
| RUS-MI1-DecisionTreeClassifier | 0.92697 | 0.01041 |
| RUS-FValue0,5-MLPC | 0.92662 | 0.01368 |
| RUS-MI1-ExtraTreeClassifier | 0.92626 | 0.01152 |
| RUS-MI0,25-SVC | 0.92592 | 0.01317 |
| RUS-MI0,75-ExtraTreeClassifier | 0.92592 | 0.01309 |
| FS-FValue0,5-ABC | 0.92592 | 0.01386 |
| RUS-FValue1-DecisionTreeClassifier | 0.92556 | 0.01269 |
| ROS-MI0,5-ExtraTreeClassifier | 0.92521 | 0.00236 |
| FS-FValue0,25-BaggingClassifier | 0.92521 | 0.00805 |
| ROS-FValue0,5-ABC | 0.92451 | 0.00907 |
| ROS-MI0,25-SVC | 0.92417 | 0.01969 |
| RUS-MI0,5-SVC | 0.92416 | 0.01806 |
| RUS-FValue0,75-ExtraTreeClassifier | 0.92416 | 0.00960 |
| RUS-FValue0,5-ABC | 0.92416 | 0.00639 |
| RUS-FValue0,5-SVC | 0.92346 | 0.01657 |
| FS-MI0,25-SVC | 0.92346 | 0.00903 |
| FS-MI0,25-BaggingClassifier | 0.92311 | 0.01525 |
| RUS-FValue0,5-ExtraTreeClassifier | 0.92276 | 0.00528 |
| RUS-MI0,5-ExtraTreeClassifier | 0.92206 | 0.01099 |
| FS-FValue0,25-SVC | 0.92171 | 0.01557 |
| FS-MI0,25-MLPC | 0.92135 | 0.00692 |
| FS-FValue0,25-MLPC | 0.92100 | 0.01532 |
| RUS-MI0,25-ABC | 0.92064 | 0.00776 |
| RUS-FValue1-ExtraTreeClassifier | 0.91855 | 0.01450 |
| RUS-FValue0,25-BaggingClassifier | 0.91855 | 0.01302 |
| ROS-FValue0,25-GBC | 0.91784 | 0.01783 |
| RUS-FValue0,25-RFC | 0.91749 | 0.01060 |
| RUS-MI0,25-DecisionTreeClassifier | 0.91748 | 0.01560 |
| ROS-MI0,25-ABC | 0.91574 | 0.01208 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0,25-ExtraTreeClassifier | 0.91503 | 0.00740 |
| FS-FValue0,25-ABC | 0.91292 | 0.00883 |
| RUS-FValue0,25-GBC | 0.91082 | 0.01700 |
| RUS-MI0,25-MLPC | 0.90976 | 0.01955 |
| FS-MI0,25-ABC | 0.90907 | 0.01102 |
| RUS-FValue0,25-ABC | 0.90906 | 0.00578 |
| RUS-MI0,25-ExtraTreeClassifier | 0.90871 | 0.01697 |
| ROS-FValue0,25-ABC | 0.90836 | 0.01703 |
| FS-FValue0,25-DecisionTreeClassifier | 0.90765 | 0.01136 |
| RUS-FValue0,25-DecisionTreeClassifier | 0.90696 | 0.01871 |
| ROS-FValue0,25-MLPC | 0.90379 | 0.01356 |
| FS-MI0,25-ExtraTreeClassifier | 0.90099 | 0.01680 |
| FS-FValue0,25-ExtraTreeClassifier | 0.90028 | 0.00722 |
| FS-MI0,25-DecisionTreeClassifier | 0.89923 | 0.02164 |
| RUS-FValue0,25-ExtraTreeClassifier | 0.89783 | 0.01520 |
| RUS-FValue0,25-SVC | 0.89046 | 0.01493 |
| RUS-FValue0,25-MLPC | 0.88660 | 0.01111 |
| ROS-FValue0,25-SVC | 0.87149 | 0.02133 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS0.75-RFC | 0.96804 | 0.01004 |
| ROS-SBS1-RFC | 0.96523 | 0.01195 |
| ROS-SFS0.75-RFC | 0.96453 | 0.00953 |
| ROS-SFS1-RFC | 0.96383 | 0.01333 |
| ROS-SFS1-MLPC | 0.96278 | 0.00798 |
| ROS-SBS1-MLPC | 0.96278 | 0.01016 |
| ROS-SBS0.75-MLPC | 0.96102 | 0.00947 |
| ROS-SFS0.75-MLPC | 0.95962 | 0.01066 |
| ROS-SFS0.75-GBC | 0.95962 | 0.00730 |
| ROS-SBS1-GBC | 0.95892 | 0.01241 |
| ROS-SFS1-GBC | 0.95892 | 0.01241 |
| ROS-SBS0.5-RFC | 0.95786 | 0.01162 |
| ROS-SBS0.75-GBC | 0.95541 | 0.00798 |
| ROS-SFS0.5-RFC | 0.95435 | 0.01222 |
| ROS-SBS0.5-GBC | 0.95048 | 0.01088 |
| ROS-SFS0.75-KNC | 0.94698 | 0.00873 |
| ROS-SBS0.5-MLPC | 0.94487 | 0.01307 |
| ROS-SFS0.5-GBC | 0.94277 | 0.01407 |
| ROS-SBS0.5-KNC | 0.94171 | 0.00626 |
| ROS-SBS0.75-KNC | 0.94171 | 0.01017 |
| ROS-SFS0.25-RFC | 0.93680 | 0.01510 |
| ROS-SBS1-KNC | 0.93398 | 0.01463 |
| ROS-SFS1-KNC | 0.93398 | 0.01463 |
| ROS-SFS0.25-GBC | 0.93153 | 0.01522 |
| ROS-SBS0.25-GBC | 0.92907 | 0.00652 |
| ROS-SBS0.25-RFC | 0.92803 | 0.01007 |
| ROS-SFS0.5-MLPC | 0.92733 | 0.02157 |
| ROS-SFS0.5-KNC | 0.92521 | 0.01407 |
| ROS-SFS0.25-MLPC | 0.92381 | 0.00438 |
| ROS-SBS0.25-MLPC | 0.92205 | 0.00631 |
| ROS-SFS0.25-KNC | 0.91889 | 0.01467 |
| ROS-SBS0.25-KNC | 0.91573 | 0.00818 |

### E.2.3. Prueba 3

## E.3. Corazon

### E.3.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.5-KNC | 0.97753 | 0.00561 |
| FS-FValue1-KNC | 0.97647 | 0.00344 |
| FS-MI1-KNC | 0.97647 | 0.00344 |
| FS-MI0.75-KNC | 0.97542 | 0.00485 |
| FS-FValue0.75-KNC | 0.97331 | 0.00706 |
| ROS-FValue0.5-KNC | 0.97261 | 0.00344 |
| ROS-MI0.75-KNC | 0.97261 | 0.00879 |
| ROS-FValue0.75-KNC | 0.96980 | 0.00825 |
| ROS-MI0.5-KNC | 0.96770 | 0.01020 |
| FS-MI0.5-KNC | 0.96699 | 0.00741 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.25-KNC | 0.94452 | 0.01160 |
| ROS-FValue0.25-KNC | 0.94346 | 0.01481 |
| ROS-MI0.25-KNC | 0.93540 | 0.01425 |
| RUS-MI0.25-KNC | 0.93257 | 0.01125 |
| FS-MI0.25-RC | 0.93083 | 0.00752 |
| RUS-FValue0.25-RC | 0.93012 | 0.00829 |
| RUS-MI0.25-SGDC | 0.92977 | 0.00787 |
| RUS-MI0.25-RC | 0.92977 | 0.00870 |
| ROS-FValue0.25-RC | 0.92942 | 0.00798 |
| FS-FValue0.5-RC | 0.92907 | 0.00800 |
| ROS-MI0.25-SGDC | 0.92907 | 0.01088 |
| FS-MI0.25-SGDC | 0.92837 | 0.00914 |
| FS-FValue0.25-NC | 0.92837 | 0.00900 |
| FS-MI0.5-SGDC | 0.92837 | 0.00927 |
| ROS-FValue0.5-SGDC | 0.92802 | 0.00861 |
| FS-FValue0.25-SGDC | 0.92802 | 0.00976 |
| FS-FValue0.25-RC | 0.92802 | 0.00925 |
| FS-MI0.5-RC | 0.92802 | 0.00679 |
| FS-FValue0.5-SGDC | 0.92767 | 0.00698 |
| ROS-FValue0.25-SGDC | 0.92767 | 0.00900 |
| FS-FValue0.75-RC | 0.92732 | 0.00836 |
| FS-MI0.75-RC | 0.92731 | 0.00902 |
| RUS-FValue0.25-SGDC | 0.92731 | 0.00867 |
| ROS-MI0.25-RC | 0.92731 | 0.00875 |
| RUS-MI0.5-RC | 0.92661 | 0.00871 |
| RUS-FValue0.75-RC | 0.92661 | 0.00972 |
| ROS-FValue0.75-RC | 0.92626 | 0.00962 |
| ROS-MI0.25-NC | 0.92626 | 0.00970 |
| ROS-FValue0.5-RC | 0.92591 | 0.00871 |
| FS-FValue1-RC | 0.92591 | 0.00886 |
| FS-MI1-RC | 0.92591 | 0.00886 |
| ROS-MI0.5-SGDC | 0.92556 | 0.01010 |
| ROS-MI0.5-RC | 0.92556 | 0.01083 |
| FS-FValue0.75-SGDC | 0.92521 | 0.00949 |
| RUS-FValue0.5-RC | 0.92486 | 0.00892 |
| RUS-MI0.5-SGDC | 0.92381 | 0.00740 |
| ROS-FValue1-RC | 0.92381 | 0.00814 |
| ROS-MI1-RC | 0.92381 | 0.00814 |
| FS-MI0.25-NC | 0.92345 | 0.01338 |
| ROS-FValue1-NC | 0.92310 | 0.00941 |
| ROS-MI1-NC | 0.92310 | 0.00941 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-NC | 0.92240 | 0.00966 |
| FS-FValue1-NC | 0.92240 | 0.00966 |
| FS-MI1-NC | 0.92240 | 0.00966 |
| RUS-FValue1-NC | 0.92205 | 0.00921 |
| RUS-MI1-NC | 0.92205 | 0.00921 |
| FS-FValue0.5-NC | 0.92205 | 0.01170 |
| ROS-FValue0.75-SGDC | 0.92170 | 0.01526 |
| ROS-FValue0.75-NC | 0.92170 | 0.00955 |
| ROS-MI0.75-RC | 0.92135 | 0.01215 |
| RUS-FValue0.75-NC | 0.92135 | 0.00953 |
| RUS-MI0.75-NC | 0.92135 | 0.01011 |
| ROS-FValue0.5-NC | 0.92100 | 0.01182 |
| RUS-FValue0.5-NC | 0.92100 | 0.01182 |
| FS-MI0.75-NC | 0.92099 | 0.01032 |
| ROS-MI0.75-NC | 0.92099 | 0.01032 |
| RUS-MI0.75-RC | 0.92099 | 0.01131 |
| ROS-FValue0.25-NC | 0.92065 | 0.01162 |
| RUS-MI0.25-NC | 0.92064 | 0.01245 |
| RUS-FValue1-RC | 0.92029 | 0.00922 |
| RUS-MI1-RC | 0.92029 | 0.00922 |
| FS-MI0.5-NC | 0.91994 | 0.01159 |
| RUS-FValue0.25-NC | 0.91959 | 0.01045 |
| ROS-MI0.25-BNB | 0.91889 | 0.01251 |
| RUS-FValue0.25-BNB | 0.91854 | 0.01266 |
| RUS-MI0.25-BNB | 0.91819 | 0.01305 |
| ROS-FValue0.25-BNB | 0.91819 | 0.01231 |
| RUS-FValue1-BNB | 0.91784 | 0.01313 |
| RUS-MI1-BNB | 0.91784 | 0.01313 |
| RUS-MI0.75-BNB | 0.91784 | 0.01270 |
| RUS-MI0.5-NC | 0.91783 | 0.01373 |
| ROS-FValue1-BNB | 0.91783 | 0.01265 |
| ROS-MI1-BNB | 0.91783 | 0.01265 |
| ROS-MI0.75-BNB | 0.91748 | 0.01228 |
| FS-MI0.75-SGDC | 0.91679 | 0.01182 |
| ROS-MI0.5-NC | 0.91678 | 0.01304 |
| RUS-FValue0.5-BNB | 0.91678 | 0.01355 |
| RUS-MI0.5-BNB | 0.91678 | 0.01397 |
| ROS-FValue0.5-BNB | 0.91643 | 0.01319 |
| RUS-FValue0.75-BNB | 0.91608 | 0.01381 |
| ROS-FValue0.75-BNB | 0.91608 | 0.01322 |
| FS-MI1-SGDC | 0.91574 | 0.01539 |
| ROS-MI0.5-BNB | 0.91082 | 0.01696 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue1-SGDC | 0.90695 | 0.02401 |
| ROS-MI0.75-SGDC | 0.90379 | 0.03178 |
| ROS-FValue1-SGDC | 0.90239 | 0.02183 |
| RUS-MI1-SGDC | 0.89570 | 0.02128 |
| FS-FValue1-BNB | 0.89537 | 0.01393 |
| FS-MI1-BNB | 0.89537 | 0.01393 |
| FS-MI0.75-BNB | 0.89537 | 0.01416 |
| ROS-MI1-SGDC | 0.89257 | 0.05607 |
| FS-FValue0.75-BNB | 0.89045 | 0.01357 |
| FS-FValue0.5-BNB | 0.88554 | 0.01126 |
| FS-MI0.5-BNB | 0.88413 | 0.01456 |
| FS-FValue0.25-BNB | 0.88202 | 0.00865 |
| FS-MI0.25-BNB | 0.87605 | 0.01093 |
| RUS-FValue0.5-SGDC | 0.86721 | 0.12435 |
| RUS-MI0.75-SGDC | 0.86659 | 0.04957 |
| RUS-FValue1-SGDC | 0.85288 | 0.03157 |
| RUS-FValue0.75-SGDC | 0.84729 | 0.12475 |

### E.3.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98596 | 0.00430 |
| ROS-MI1-RFC | 0.98595 | 0.00401 |
| FS-FValue1-RFC | 0.98279 | 0.00132 |
| RUS-MI1-RFC | 0.98209 | 0.00489 |
| FS-MI1-RFC | 0.98209 | 0.00358 |
| RUS-FValue1-RFC | 0.98104 | 0.00581 |
| RUS-FValue0.75-RFC | 0.98034 | 0.00640 |
| ROS-MI0.75-RFC | 0.98034 | 0.00591 |
| ROS-FValue1-GBC | 0.98034 | 0.00390 |
| ROS-MI1-GBC | 0.97999 | 0.00361 |
| FS-MI0.75-RFC | 0.97964 | 0.00343 |
| RUS-FValue1-GBC | 0.97928 | 0.00623 |
| ROS-FValue0.75-RFC | 0.97893 | 0.00776 |
| FS-FValue0.75-RFC | 0.97893 | 0.00736 |
| RUS-MI1-GBC | 0.97858 | 0.00661 |
| ROS-FValue0.5-RFC | 0.97718 | 0.00575 |
| RUS-MI0.75-RFC | 0.97718 | 0.00532 |
| RUS-FValue0.75-GBC | 0.97718 | 0.00293 |
| ROS-MI0.75-GBC | 0.97648 | 0.00623 |
| FS-FValue0.5-RFC | 0.97577 | 0.00622 |
| ROS-FValue1-MLPC | 0.97577 | 0.00407 |
| RUS-FValue0.5-RFC | 0.97543 | 0.00954 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.75-GBC | 0.97507 | 0.00301 |
| FS-MI0.5-RFC | 0.97472 | 0.00706 |
| FS-MI1-MLPC | 0.97437 | 0.00603 |
| FS-FValue1-MLPC | 0.97437 | 0.00708 |
| ROS-MI0.5-RFC | 0.97402 | 0.00803 |
| ROS-FValue0.75-GBC | 0.97367 | 0.00726 |
| ROS-FValue1-BaggingClassifier | 0.97367 | 0.00894 |
| FS-FValue1-GBC | 0.97367 | 0.00761 |
| FS-FValue1-ABC | 0.97332 | 0.00524 |
| FS-MI1-ABC | 0.97332 | 0.00524 |
| RUS-MI0.5-RFC | 0.97297 | 0.00932 |
| FS-MI1-GBC | 0.97261 | 0.00798 |
| RUS-FValue0.75-BaggingClassifier | 0.97226 | 0.00464 |
| ROS-MI1-BaggingClassifier | 0.97226 | 0.00303 |
| ROS-FValue1-ABC | 0.97226 | 0.00951 |
| ROS-MI1-ABC | 0.97226 | 0.00951 |
| RUS-FValue1-BaggingClassifier | 0.97191 | 0.00533 |
| ROS-MI1-MLPC | 0.97191 | 0.00639 |
| ROS-MI0.5-BaggingClassifier | 0.97156 | 0.00924 |
| ROS-MI0.5-GBC | 0.97156 | 0.00818 |
| RUS-MI1-BaggingClassifier | 0.97121 | 0.00515 |
| FS-MI0.75-GBC | 0.97086 | 0.00878 |
| RUS-FValue1-ABC | 0.97086 | 0.00503 |
| RUS-MI1-ABC | 0.97086 | 0.00503 |
| FS-MI0.75-BaggingClassifier | 0.97016 | 0.00665 |
| ROS-FValue0.75-BaggingClassifier | 0.97016 | 0.00744 |
| FS-MI1-BaggingClassifier | 0.96946 | 0.00592 |
| FS-FValue1-BaggingClassifier | 0.96945 | 0.00716 |
| ROS-FValue0.5-GBC | 0.96805 | 0.00825 |
| FS-FValue0.75-GBC | 0.96805 | 0.00581 |
| ROS-MI0.75-BaggingClassifier | 0.96805 | 0.00661 |
| RUS-FValue0.5-GBC | 0.96770 | 0.00670 |
| FS-MI0.5-BaggingClassifier | 0.96770 | 0.00539 |
| FS-FValue0.5-BaggingClassifier | 0.96770 | 0.00539 |
| FS-FValue0.75-ABC | 0.96769 | 0.00759 |
| FS-FValue0.75-BaggingClassifier | 0.96734 | 0.00562 |
| FS-FValue0.5-GBC | 0.96734 | 0.00614 |
| ROS-MI0.75-ABC | 0.96699 | 0.00780 |
| ROS-FValue0.5-BaggingClassifier | 0.96664 | 0.00628 |
| RUS-MI0.5-BaggingClassifier | 0.96629 | 0.00903 |
| ROS-FValue0.75-ABC | 0.96629 | 0.00795 |
| FS-MI0.5-GBC | 0.96524 | 0.00463 |
| RUS-MI0.75-BaggingClassifier | 0.96523 | 0.00759 |
| RUS-FValue0.75-ABC | 0.96489 | 0.00458 |
| RUS-MI0.5-GBC | 0.96489 | 0.00599 |
| ROS-MI0.5-DecisionTreeClassifier | 0.96454 | 0.00747 |
| FS-FValue1-ExtraTreeClassifier | 0.96419 | 0.00502 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.5-BaggingClassifier | 0.96419 | 0.01134 |
| RUS-MI0.75-ABC | 0.96313 | 0.00587 |
| ROS-MI0.5-ABC | 0.96313 | 0.01041 |
| ROS-MI1-ExtraTreeClassifier | 0.96313 | 0.00761 |
| FS-MI0.75-ABC | 0.96313 | 0.00648 |
| ROS-FValue0.75-MLPC | 0.96243 | 0.00362 |
| FS-MI1-ExtraTreeClassifier | 0.96208 | 0.00596 |
| FS-MI1-DecisionTreeClassifier | 0.96173 | 0.00918 |
| FS-FValue0.5-ABC | 0.96103 | 0.00591 |
| ROS-FValue0.5-ExtraTreeClassifier | 0.96103 | 0.00280 |
| RUS-FValue0.5-ABC | 0.95997 | 0.00995 |
| FS-MI0.75-DecisionTreeClassifier | 0.95927 | 0.00963 |
| RUS-FValue1-MLPC | 0.95892 | 0.00743 |
| ROS-FValue1-DecisionTreeClassifier | 0.95857 | 0.01032 |
| ROS-FValue0.5-ABC | 0.95857 | 0.00919 |
| FS-FValue0.5-DecisionTreeClassifier | 0.95752 | 0.00846 |
| ROS-MI0.75-DecisionTreeClassifier | 0.95751 | 0.00865 |
| FS-MI0.5-ABC | 0.95751 | 0.01265 |
| ROS-FValue1-ExtraTreeClassifier | 0.95716 | 0.00613 |
| ROS-FValue0.75-ExtraTreeClassifier | 0.95716 | 0.00671 |
| FS-FValue0.75-DecisionTreeClassifier | 0.95716 | 0.00879 |
| FS-MI0.75-ExtraTreeClassifier | 0.95716 | 0.00774 |
| FS-FValue1-DecisionTreeClassifier | 0.95682 | 0.00983 |
| ROS-FValue0.75-DecisionTreeClassifier | 0.95646 | 0.00995 |
| ROS-MI0.75-ExtraTreeClassifier | 0.95646 | 0.00730 |
| FS-MI0.25-RFC | 0.95646 | 0.00644 |
| RUS-MI0.25-RFC | 0.95576 | 0.00205 |
| ROS-MI0.25-RFC | 0.95541 | 0.01291 |
| ROS-FValue0.5-MLPC | 0.95540 | 0.00455 |
| FS-FValue0.5-ExtraTreeClassifier | 0.95505 | 0.00576 |
| ROS-MI1-DecisionTreeClassifier | 0.95471 | 0.01377 |
| FS-FValue0.75-ExtraTreeClassifier | 0.95471 | 0.00731 |
| ROS-FValue0.5-DecisionTreeClassifier | 0.95436 | 0.01532 |
| ROS-FValue0.25-GBC | 0.95400 | 0.00696 |
| ROS-FValue0.25-RFC | 0.95365 | 0.00907 |
| RUS-MI0.5-DecisionTreeClassifier | 0.95330 | 0.00706 |
| RUS-MI0.5-ABC | 0.95330 | 0.00615 |
| RUS-MI1-MLPC | 0.95294 | 0.01231 |
| RUS-FValue1-DecisionTreeClassifier | 0.95260 | 0.01104 |
| RUS-MI0.25-BaggingClassifier | 0.95260 | 0.00354 |
| FS-MI0.5-ExtraTreeClassifier | 0.95225 | 0.00660 |
| ROS-MI0.25-BaggingClassifier | 0.95190 | 0.01020 |
| RUS-MI1-DecisionTreeClassifier | 0.95190 | 0.00972 |
| ROS-FValue0.25-BaggingClassifier | 0.95190 | 0.01198 |
| RUS-MI0.75-DecisionTreeClassifier | 0.95155 | 0.01211 |
| FS-MI0.5-DecisionTreeClassifier | 0.95154 | 0.01206 |
| RUS-FValue0.5-DecisionTreeClassifier | 0.95085 | 0.00991 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI0.25-BaggingClassifier | 0.95049 | 0.00796 |
| ROS-MI0.5-ExtraTreeClassifier | 0.95014 | 0.00549 |
| RUS-FValue0.25-RFC | 0.95014 | 0.01163 |
| ROS-MI0.25-GBC | 0.94979 | 0.01538 |
| FS-MI0.25-GBC | 0.94979 | 0.00595 |
| RUS-FValue0.75-DecisionTreeClassifier | 0.94909 | 0.01529 |
| RUS-FValue0.75-ExtraTreeClassifier | 0.94839 | 0.00932 |
| RUS-FValue0.5-ExtraTreeClassifier | 0.94663 | 0.00975 |
| FS-FValue0.75-MLPC | 0.94592 | 0.00985 |
| RUS-MI0.25-GBC | 0.94558 | 0.00290 |
| FS-MI0.25-DecisionTreeClassifier | 0.94558 | 0.01191 |
| FS-FValue0.25-RFC | 0.94522 | 0.00748 |
| RUS-MI0.75-ExtraTreeClassifier | 0.94488 | 0.00875 |
| RUS-MI1-ExtraTreeClassifier | 0.94452 | 0.00820 |
| FS-FValue0.25-BaggingClassifier | 0.94382 | 0.01013 |
| FS-MI0.25-ABC | 0.94311 | 0.00890 |
| FS-FValue0.25-GBC | 0.94277 | 0.00560 |
| FS-MI0.75-MLPC | 0.94241 | 0.00725 |
| ROS-MI0.75-MLPC | 0.94206 | 0.01373 |
| RUS-FValue0.25-GBC | 0.94171 | 0.01287 |
| RUS-FValue0.25-BaggingClassifier | 0.94101 | 0.01475 |
| ROS-FValue1-SVC | 0.94065 | 0.01212 |
| ROS-MI1-SVC | 0.94065 | 0.01212 |
| ROS-MI0.25-DecisionTreeClassifier | 0.93961 | 0.00997 |
| RUS-MI0.5-ExtraTreeClassifier | 0.93891 | 0.01102 |
| ROS-FValue0.25-ExtraTreeClassifier | 0.93890 | 0.01310 |
| RUS-FValue0.75-MLPC | 0.93855 | 0.00694 |
| FS-MI0.25-ExtraTreeClassifier | 0.93820 | 0.00863 |
| ROS-FValue0.25-DecisionTreeClassifier | 0.93785 | 0.01616 |
| RUS-FValue1-ExtraTreeClassifier | 0.93749 | 0.01560 |
| RUS-FValue0.5-MLPC | 0.93714 | 0.01098 |
| ROS-MI0.25-ExtraTreeClassifier | 0.93681 | 0.02192 |
| FS-MI0.5-MLPC | 0.93539 | 0.00828 |
| ROS-FValue0.5-SVC | 0.93504 | 0.00911 |
| RUS-FValue0.5-SVC | 0.93504 | 0.00911 |
| FS-FValue0.5-MLPC | 0.93504 | 0.00870 |
| FS-MI0.25-MLPC | 0.93434 | 0.00807 |
| FS-FValue0.5-SVC | 0.93434 | 0.00887 |
| FS-FValue1-SVC | 0.93434 | 0.00955 |
| FS-MI1-SVC | 0.93434 | 0.00955 |
| RUS-MI0.5-SVC | 0.93399 | 0.00719 |
| RUS-MI0.25-SVC | 0.93363 | 0.00927 |
| FS-MI0.25-SVC | 0.93363 | 0.00865 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-SVC | 0.93363 | 0.01004 |
| RUS-FValue0.25-DecisionTreeClassifier | 0.93329 | 0.01924 |
| FS-MI0.75-SVC | 0.93328 | 0.00897 |
| RUS-MI0.25-MLPC | 0.93328 | 0.00848 |
| ROS-MI0.25-SVC | 0.93328 | 0.00911 |
| ROS-FValue0.75-SVC | 0.93328 | 0.01096 |
| FS-MI0.5-SVC | 0.93293 | 0.00843 |
| RUS-FValue0.25-SVC | 0.93223 | 0.00751 |
| FS-FValue0.25-MLPC | 0.93223 | 0.00800 |
| FS-FValue0.25-SVC | 0.93223 | 0.00800 |
| RUS-FValue0.75-SVC | 0.93188 | 0.00865 |
| ROS-FValue0.25-SVC | 0.93188 | 0.00783 |
| RUS-FValue0.25-MLPC | 0.93153 | 0.00779 |
| ROS-FValue0.25-MLPC | 0.93153 | 0.00870 |
| RUS-MI0.75-SVC | 0.93153 | 0.00958 |
| ROS-MI0.5-MLPC | 0.93083 | 0.00847 |
| FS-FValue0.25-ExtraTreeClassifier | 0.93083 | 0.00591 |
| ROS-MI0.25-ABC | 0.92978 | 0.01681 |
| RUS-MI0.5-MLPC | 0.92977 | 0.01232 |
| FS-FValue0.25-ABC | 0.92803 | 0.00963 |
| ROS-MI0.25-MLPC | 0.92767 | 0.00807 |
| RUS-FValue1-SVC | 0.92766 | 0.01457 |
| RUS-MI1-SVC | 0.92766 | 0.01457 |
| ROS-MI0.5-SVC | 0.92766 | 0.01718 |
| RUS-MI0.25-ABC | 0.92626 | 0.01062 |
| FS-FValue0.25-DecisionTreeClassifier | 0.92556 | 0.00676 |
| ROS-FValue0.25-ABC | 0.92486 | 0.01059 |
| ROS-MI0.75-SVC | 0.92345 | 0.01631 |
| RUS-MI0.25-DecisionTreeClassifier | 0.92275 | 0.00993 |
| RUS-MI0.75-MLPC | 0.92099 | 0.01542 |
| RUS-FValue0.25-ABC | 0.91924 | 0.01053 |
| RUS-MI0.25-ExtraTreeClassifier | 0.91081 | 0.01685 |
| RUS-FValue0.25-ExtraTreeClassifier | 0.90800 | 0.01294 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS1-RFC | 0.98525 | 0.00593 |
| ROS-SBS1-RFC | 0.98245 | 0.00618 |
| ROS-SFS0.75-RFC | 0.98245 | 0.00313 |
| ROS-SFS0.75-GBC | 0.98069 | 0.00429 |
| ROS-SBS1-GBC | 0.98069 | 0.00430 |
| ROS-SFS1-GBC | 0.98034 | 0.00488 |
| ROS-SBS0.75-RFC | 0.98034 | 0.00787 |
| ROS-SFS0.75-MLPC | 0.97964 | 0.00593 |
| ROS-SBS0.75-GBC | 0.97753 | 0.00687 |
| ROS-SBS1-MLPC | 0.97753 | 0.00206 |
| ROS-SFS0.75-KNC | 0.97718 | 0.00248 |
| ROS-SFS0.5-RFC | 0.97612 | 0.00481 |
| ROS-SBS0.75-MLPC | 0.97437 | 0.00899 |
| ROS-SFS1-MLPC | 0.97437 | 0.00691 |
| ROS-SFS0.5-KNC | 0.97331 | 0.00205 |
| ROS-SBS0.5-RFC | 0.97192 | 0.00979 |
| ROS-SBS0.75-KNC | 0.97121 | 0.00789 |
| ROS-SFS0.5-MLPC | 0.97121 | 0.00324 |
| ROS-SFS0.5-GBC | 0.97086 | 0.00178 |
| ROS-SBS0.5-GBC | 0.96840 | 0.01004 |
| ROS-SBS1-KNC | 0.96629 | 0.00408 |
| ROS-SFS1-KNC | 0.96629 | 0.00408 |
| ROS-SFS0.25-RFC | 0.96348 | 0.00516 |
| ROS-SBS0.5-KNC | 0.95611 | 0.00799 |
| ROS-SFS0.25-GBC | 0.95400 | 0.00725 |
| ROS-SBS0.25-RFC | 0.94874 | 0.01459 |
| ROS-SFS0.25-KNC | 0.94558 | 0.00400 |
| ROS-SFS0.25-MLPC | 0.94417 | 0.01182 |
| ROS-SBS0.5-MLPC | 0.93995 | 0.00799 |
| ROS-SBS0.25-GBC | 0.93926 | 0.01313 |
| ROS-SBS0.25-MLPC | 0.93328 | 0.01357 |
| ROS-SBS0.25-KNC | 0.92767 | 0.00513 |

### E.3.3. Prueba 3

## E.4. Anular

### E.4.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.75-KNC | 0.98069 | 0.00556 |
| ROS-FValue0.75-KNC | 0.97999 | 0.00505 |
| FS-FValue0.75-KNC | 0.97998 | 0.00552 |
| ROS-MI0.75-KNC | 0.97823 | 0.00805 |
| FS-MI0.75-KNC | 0.97753 | 0.00877 |
| RUS-MI0.75-KNC | 0.97718 | 0.00745 |
| ROS-FValue1-KNC | 0.97613 | 0.00750 |
| ROS-MI1-KNC | 0.97613 | 0.00750 |
| RUS-MI0.5-KNC | 0.97542 | 0.01006 |
| RUS-FValue1-KNC | 0.97507 | 0.00788 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.5-RC | 0.85710 | 0.01735 |
| FS-MI0.5-RC | 0.85570 | 0.01816 |
| ROS-FValue0.25-SGDC | 0.85254 | 0.01767 |
| RUS-FValue1-NC | 0.85149 | 0.01899 |
| RUS-MI1-NC | 0.85149 | 0.01899 |
| ROS-FValue0.25-RC | 0.85149 | 0.01726 |
| RUS-MI0.25-RC | 0.85149 | 0.01559 |
| FS-FValue1-NC | 0.85114 | 0.01955 |
| FS-MI1-NC | 0.85114 | 0.01955 |
| ROS-FValue1-NC | 0.85114 | 0.01872 |
| ROS-MI1-NC | 0.85114 | 0.01872 |
| RUS-MI0.25-SGDC | 0.85114 | 0.01775 |
| RUS-FValue0.25-SGDC | 0.85114 | 0.01823 |
| ROS-MI0.25-RC | 0.85113 | 0.01569 |
| RUS-FValue0.25-RC | 0.85079 | 0.01967 |
| FS-FValue0.25-SGDC | 0.85079 | 0.01804 |
| FS-MI0.25-SGDC | 0.85078 | 0.01731 |
| FS-FValue0.25-RC | 0.85008 | 0.01729 |
| ROS-MI0.25-SGDC | 0.85008 | 0.01758 |
| FS-FValue0.75-NC | 0.84973 | 0.01993 |
| ROS-FValue0.75-NC | 0.84973 | 0.01993 |
| FS-FValue0.25-NC | 0.84973 | 0.01764 |
| ROS-FValue0.25-NC | 0.84973 | 0.01764 |
| RUS-FValue0.25-NC | 0.84903 | 0.01840 |
| FS-MI0.25-RC | 0.84903 | 0.01857 |
| RUS-FValue0.75-NC | 0.84868 | 0.02023 |
| ROS-FValue0.75-RC | 0.84833 | 0.01783 |
| RUS-FValue0.75-RC | 0.84832 | 0.01662 |
| RUS-MI0.75-RC | 0.84798 | 0.02015 |
| RUS-MI0.25-NC | 0.84798 | 0.01764 |
| FS-MI0.25-NC | 0.84763 | 0.01818 |
| ROS-MI0.25-NC | 0.84763 | 0.01818 |
| FS-MI0.75-RC | 0.84728 | 0.02177 |
| FS-FValue0.75-RC | 0.84692 | 0.01646 |
| ROS-MI0.75-RC | 0.84657 | 0.02287 |
| RUS-FValue0.5-SGDC | 0.84622 | 0.01215 |
| ROS-FValue1-BNB | 0.84587 | 0.01995 |
| ROS-MI1-BNB | 0.84587 | 0.01995 |
| RUS-FValue1-BNB | 0.84587 | 0.01995 |
| RUS-MI1-BNB | 0.84587 | 0.01995 |
| RUS-MI0.5-NC | 0.84587 | 0.02032 |
| FS-FValue1-BNB | 0.84587 | 0.01986 |
| FS-MI1-BNB | 0.84587 | 0.01986 |
| RUS-FValue1-RC | 0.84587 | 0.01877 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI1-RC | 0.84587 | 0.01877 |
| FS-FValue0.75-BNB | 0.84552 | 0.01978 |
| FS-MI0.75-NC | 0.84517 | 0.02172 |
| ROS-MI0.75-NC | 0.84517 | 0.02172 |
| RUS-MI0.75-NC | 0.84517 | 0.02172 |
| ROS-MI0.5-NC | 0.84517 | 0.02085 |
| ROS-FValue1-RC | 0.84516 | 0.01893 |
| ROS-MI1-RC | 0.84516 | 0.01893 |
| FS-MI0.5-NC | 0.84482 | 0.02135 |
| FS-FValue0.5-NC | 0.84376 | 0.02125 |
| ROS-FValue0.5-NC | 0.84376 | 0.02125 |
| RUS-FValue0.5-NC | 0.84376 | 0.02125 |
| FS-MI0.75-BNB | 0.84376 | 0.01981 |
| ROS-FValue0.75-BNB | 0.84376 | 0.01902 |
| RUS-FValue0.75-BNB | 0.84376 | 0.01902 |
| RUS-MI0.75-BNB | 0.84306 | 0.01974 |
| ROS-MI0.75-BNB | 0.84271 | 0.01970 |
| FS-FValue0.25-BNB | 0.84236 | 0.02010 |
| FS-MI0.5-BNB | 0.84201 | 0.02145 |
| RUS-FValue0.5-BNB | 0.84201 | 0.01996 |
| FS-MI0.25-BNB | 0.84201 | 0.01984 |
| FS-FValue1-RC | 0.84200 | 0.01577 |
| FS-MI1-RC | 0.84200 | 0.01577 |
| ROS-MI0.5-BNB | 0.84166 | 0.02140 |
| RUS-MI0.5-BNB | 0.84166 | 0.02140 |
| ROS-FValue0.5-BNB | 0.84166 | 0.01963 |
| FS-FValue0.5-BNB | 0.84131 | 0.02073 |
| FS-FValue0.5-RC | 0.83990 | 0.02208 |
| ROS-FValue0.25-BNB | 0.83990 | 0.02128 |
| RUS-FValue0.25-BNB | 0.83990 | 0.02052 |
| ROS-MI0.25-BNB | 0.83955 | 0.02103 |
| RUS-MI0.25-BNB | 0.83955 | 0.02029 |
| RUS-FValue0.5-RC | 0.83885 | 0.02108 |
| ROS-FValue0.5-RC | 0.83850 | 0.02162 |

### E.4.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue1-RFC | 0.98560 | 0.00406 |
| RUS-MI1-MLPC | 0.98560 | 0.00773 |
| FS-FValue1-RFC | 0.98525 | 0.00286 |
| ROS-FValue1-MLPC | 0.98490 | 0.00562 |
| FS-MI1-MLPC | 0.98490 | 0.00782 |
| FS-FValue1-MLPC | 0.98455 | 0.00780 |
| ROS-MI1-MLPC | 0.98455 | 0.00715 |
| RUS-FValue1-MLPC | 0.98455 | 0.00757 |
| ROS-FValue0.75-RFC | 0.98420 | 0.00647 |
| ROS-MI1-RFC | 0.98420 | 0.00532 |
| FS-MI1-RFC | 0.98385 | 0.00525 |
| RUS-MI1-RFC | 0.98385 | 0.00421 |
| FS-FValue0.75-RFC | 0.98350 | 0.00479 |
| RUS-FValue1-RFC | 0.98315 | 0.00550 |
| RUS-FValue0.75-RFC | 0.98280 | 0.00651 |
| FS-FValue0.75-GBC | 0.98209 | 0.00449 |
| FS-FValue1-GBC | 0.98209 | 0.00679 |
| FS-MI1-GBC | 0.98209 | 0.00679 |
| ROS-MI0.75-RFC | 0.98174 | 0.00504 |
| ROS-FValue1-GBC | 0.98174 | 0.00708 |
| ROS-MI1-GBC | 0.98174 | 0.00634 |
| FS-MI0.75-RFC | 0.98139 | 0.00439 |
| RUS-MI1-GBC | 0.98069 | 0.00617 |
| ROS-FValue0.75-GBC | 0.98069 | 0.00293 |
| RUS-FValue1-GBC | 0.98034 | 0.00621 |
| FS-FValue1-ABC | 0.98034 | 0.00537 |
| FS-MI1-ABC | 0.98034 | 0.00537 |
| FS-MI0.75-GBC | 0.97964 | 0.00634 |
| FS-FValue0.75-BaggingClassifier | 0.97893 | 0.00823 |
| FS-MI0.75-BaggingClassifier | 0.97893 | 0.00711 |
| ROS-MI1-BaggingClassifier | 0.97893 | 0.00657 |
| ROS-FValue1-ABC | 0.97893 | 0.00416 |
| ROS-MI1-ABC | 0.97893 | 0.00416 |
| RUS-FValue0.75-GBC | 0.97858 | 0.00679 |
| RUS-MI0.75-GBC | 0.97823 | 0.00742 |
| FS-MI1-BaggingClassifier | 0.97823 | 0.00813 |
| RUS-MI0.75-RFC | 0.97788 | 0.00663 |
| ROS-FValue1-BaggingClassifier | 0.97753 | 0.00897 |
| RUS-FValue1-BaggingClassifier | 0.97753 | 0.00855 |
| FS-FValue1-BaggingClassifier | 0.97753 | 0.00764 |
| ROS-MI0.75-GBC | 0.97753 | 0.00592 |
| ROS-FValue0.75-ABC | 0.97753 | 0.00679 |
| RUS-MI0.5-RFC | 0.97753 | 0.00463 |
| RUS-FValue1-ABC | 0.97753 | 0.00233 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI1-ABC | 0.97753 | 0.00233 |
| ROS-FValue0.75-BaggingClassifier | 0.97718 | 0.00859 |
| FS-FValue0.75-ABC | 0.97718 | 0.00333 |
| RUS-MI1-BaggingClassifier | 0.97648 | 0.00990 |
| ROS-FValue0.5-RFC | 0.97542 | 0.00555 |
| FS-MI0.5-GBC | 0.97507 | 0.00623 |
| FS-MI0.5-RFC | 0.97472 | 0.00594 |
| ROS-MI0.75-BaggingClassifier | 0.97472 | 0.00690 |
| RUS-FValue0.75-BaggingClassifier | 0.97437 | 0.00741 |
| RUS-MI0.75-BaggingClassifier | 0.97437 | 0.00652 |
| RUS-FValue0.75-ABC | 0.97437 | 0.00492 |
| FS-FValue0.5-RFC | 0.97437 | 0.00439 |
| ROS-MI0.5-RFC | 0.97402 | 0.00559 |
| FS-FValue0.5-GBC | 0.97401 | 0.01015 |
| ROS-MI0.5-GBC | 0.97367 | 0.00675 |
| ROS-FValue0.5-GBC | 0.97296 | 0.00742 |
| FS-MI0.75-ABC | 0.97261 | 0.00634 |
| RUS-MI0.5-GBC | 0.97261 | 0.00750 |
| RUS-FValue0.5-GBC | 0.97191 | 0.00974 |
| RUS-FValue0.5-RFC | 0.97191 | 0.00657 |
| RUS-MI0.75-ABC | 0.97191 | 0.00510 |
| ROS-MI0.75-ABC | 0.97156 | 0.00477 |
| FS-MI0.75-MLPC | 0.97121 | 0.01168 |
| FS-FValue0.5-BaggingClassifier | 0.97121 | 0.00758 |
| RUS-MI0.5-BaggingClassifier | 0.97051 | 0.00811 |
| FS-MI1-DecisionTreeClassifier | 0.97050 | 0.00977 |
| RUS-MI1-DecisionTreeClassifier | 0.96980 | 0.01004 |
| ROS-FValue0.5-BaggingClassifier | 0.96945 | 0.00708 |
| ROS-FValue1-DecisionTreeClassifier | 0.96945 | 0.01147 |
| RUS-FValue0.75-MLPC | 0.96875 | 0.01015 |
| ROS-MI0.5-BaggingClassifier | 0.96840 | 0.01210 |
| FS-MI0.5-BaggingClassifier | 0.96805 | 0.00904 |
| FS-FValue0.75-MLPC | 0.96805 | 0.00932 |
| FS-FValue0.5-DecisionTreeClassifier | 0.96805 | 0.00990 |
| ROS-MI1-DecisionTreeClassifier | 0.96770 | 0.00871 |
| RUS-FValue0.5-BaggingClassifier | 0.96770 | 0.00699 |
| ROS-FValue0.5-DecisionTreeClassifier | 0.96770 | 0.00614 |
| FS-FValue1-DecisionTreeClassifier | 0.96770 | 0.00681 |
| FS-FValue0.75-DecisionTreeClassifier | 0.96735 | 0.00765 |
| ROS-FValue0.75-MLPC | 0.96734 | 0.01265 |
| ROS-MI0.75-MLPC | 0.96664 | 0.01392 |
| RUS-FValue0.5-DecisionTreeClassifier | 0.96664 | 0.00638 |
| ROS-FValue0.75-DecisionTreeClassifier | 0.96664 | 0.01280 |
| ROS-FValue1-ExtraTreeClassifier | 0.96664 | 0.00770 |
| RUS-FValue0.75-DecisionTreeClassifier | 0.96664 | 0.01330 |
| RUS-FValue1-DecisionTreeClassifier | 0.96664 | 0.00911 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-MI0.75-MLPC | 0.96629 | 0.01049 |
| ROS-MI1-ExtraTreeClassifier | 0.96594 | 0.00573 |
| FS-FValue0.75-ExtraTreeClassifier | 0.96489 | 0.00456 |
| FS-FValue1-ExtraTreeClassifier | 0.96489 | 0.00648 |
| RUS-MI0.5-DecisionTreeClassifier | 0.96489 | 0.01182 |
| ROS-FValue1-SVC | 0.96454 | 0.00989 |
| ROS-MI1-SVC | 0.96454 | 0.00989 |
| RUS-MI0.5-ABC | 0.96453 | 0.00914 |
| FS-FValue1-SVC | 0.96418 | 0.01319 |
| FS-MI1-SVC | 0.96418 | 0.01319 |
| ROS-MI0.75-ExtraTreeClassifier | 0.96384 | 0.01390 |
| RUS-MI0.5-MLPC | 0.96384 | 0.01381 |
| FS-MI0.5-DecisionTreeClassifier | 0.96383 | 0.01033 |
| FS-MI0.75-DecisionTreeClassifier | 0.96383 | 0.01259 |
| RUS-MI0.75-DecisionTreeClassifier | 0.96383 | 0.01110 |
| FS-FValue0.25-RFC | 0.96278 | 0.00696 |
| ROS-FValue0.75-ExtraTreeClassifier | 0.96278 | 0.00749 |
| ROS-MI0.5-MLPC | 0.96243 | 0.01931 |
| ROS-MI0.75-DecisionTreeClassifier | 0.96243 | 0.00749 |
| RUS-MI1-ExtraTreeClassifier | 0.96138 | 0.00587 |
| RUS-FValue1-SVC | 0.96067 | 0.00920 |
| RUS-MI1-SVC | 0.96067 | 0.00920 |
| FS-MI1-ExtraTreeClassifier | 0.96067 | 0.00776 |
| RUS-FValue0.25-RFC | 0.95998 | 0.01175 |
| RUS-MI0.5-ExtraTreeClassifier | 0.95997 | 0.00448 |
| FS-MI0.5-MLPC | 0.95963 | 0.01434 |
| ROS-FValue0.25-RFC | 0.95962 | 0.00909 |
| ROS-MI0.5-ABC | 0.95927 | 0.01465 |
| FS-MI0.75-ExtraTreeClassifier | 0.95892 | 0.00452 |
| FS-FValue0.25-BaggingClassifier | 0.95857 | 0.00594 |
| FS-MI0.5-ExtraTreeClassifier | 0.95822 | 0.00883 |
| RUS-FValue1-ExtraTreeClassifier | 0.95821 | 0.00893 |
| ROS-MI0.5-ExtraTreeClassifier | 0.95752 | 0.01007 |
| RUS-FValue0.75-ExtraTreeClassifier | 0.95717 | 0.00747 |
| ROS-MI0.5-DecisionTreeClassifier | 0.95717 | 0.01243 |
| ROS-FValue0.5-ExtraTreeClassifier | 0.95716 | 0.00603 |
| FS-MI0.5-ABC | 0.95681 | 0.01214 |
| ROS-FValue0.25-BaggingClassifier | 0.95611 | 0.00726 |
| FS-FValue0.5-ABC | 0.95541 | 0.00632 |
| RUS-FValue0.5-ABC | 0.95330 | 0.00358 |
| ROS-FValue0.5-ABC | 0.95225 | 0.00695 |
| RUS-FValue0.25-BaggingClassifier | 0.95225 | 0.00579 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.25-GBC | 0.95225 | 0.00678 |
| RUS-FValue0.25-GBC | 0.95085 | 0.01009 |
| ROS-MI0.25-RFC | 0.95085 | 0.00915 |
| RUS-MI0.25-RFC | 0.95049 | 0.00834 |
| FS-MI0.25-RFC | 0.95049 | 0.00680 |
| ROS-FValue0.5-MLPC | 0.95049 | 0.01129 |
| RUS-FValue0.5-ExtraTreeClassifier | 0.95014 | 0.01514 |
| ROS-FValue0.25-GBC | 0.94944 | 0.00523 |
| FS-FValue0.5-ExtraTreeClassifier | 0.94944 | 0.00474 |
| RUS-MI0.75-ExtraTreeClassifier | 0.94839 | 0.00723 |
| FS-MI0.25-BaggingClassifier | 0.94804 | 0.00959 |
| RUS-FValue0.5-MLPC | 0.94768 | 0.01028 |
| ROS-MI0.25-GBC | 0.94734 | 0.01040 |
| FS-FValue0.5-MLPC | 0.94733 | 0.00949 |
| FS-MI0.25-GBC | 0.94628 | 0.01144 |
| RUS-MI0.25-GBC | 0.94593 | 0.01169 |
| RUS-MI0.25-BaggingClassifier | 0.94593 | 0.00778 |
| ROS-FValue0.75-SVC | 0.94558 | 0.01451 |
| ROS-MI0.25-BaggingClassifier | 0.94558 | 0.00674 |
| FS-FValue0.75-SVC | 0.94417 | 0.01520 |
| ROS-FValue0.25-ExtraTreeClassifier | 0.94312 | 0.00764 |
| RUS-FValue0.75-SVC | 0.94277 | 0.01292 |
| RUS-FValue0.25-DecisionTreeClassifier | 0.94241 | 0.01381 |
| FS-FValue0.25-DecisionTreeClassifier | 0.94136 | 0.00953 |
| ROS-FValue0.25-DecisionTreeClassifier | 0.94136 | 0.01205 |
| RUS-MI0.25-ExtraTreeClassifier | 0.93504 | 0.00780 |
| RUS-MI0.25-ABC | 0.93435 | 0.01494 |
| FS-MI0.25-MLPC | 0.93434 | 0.01199 |
| RUS-FValue0.25-ExtraTreeClassifier | 0.93364 | 0.01502 |
| RUS-MI0.25-DecisionTreeClassifier | 0.93329 | 0.00654 |
| ROS-MI0.25-MLPC | 0.93329 | 0.01481 |
| ROS-MI0.25-ABC | 0.93294 | 0.01391 |
| FS-MI0.25-ABC | 0.93259 | 0.01531 |
| FS-MI0.25-ExtraTreeClassifier | 0.93259 | 0.00599 |
| RUS-MI0.25-MLPC | 0.93188 | 0.01388 |
| FS-FValue0.25-ExtraTreeClassifier | 0.93083 | 0.00967 |
| ROS-MI0.25-DecisionTreeClassifier | 0.92978 | 0.01019 |
| FS-MI0.25-DecisionTreeClassifier | 0.92943 | 0.00794 |
| ROS-MI0.25-ExtraTreeClassifier | 0.92767 | 0.00729 |
| ROS-FValue0.25-ABC | 0.92662 | 0.01081 |
| FS-FValue0.25-ABC | 0.92627 | 0.00842 |
| RUS-FValue0.25-ABC | 0.92592 | 0.00567 |
| FS-FValue0.5-SVC | 0.91994 | 0.01472 |
| FS-MI0.75-SVC | 0.91889 | 0.01902 |
| ROS-MI0.75-SVC | 0.90697 | 0.03122 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI0.25-SVC | 0.90660 | 0.02358 |
| ROS-FValue0.5-SVC | 0.90416 | 0.03800 |
| RUS-MI0.75-SVC | 0.89889 | 0.03063 |
| FS-MI0.5-SVC | 0.89853 | 0.01817 |
| FS-FValue0.25-MLPC | 0.88765 | 0.01849 |
| RUS-FValue0.25-MLPC | 0.88694 | 0.00919 |
| ROS-MI0.5-SVC | 0.88380 | 0.04135 |
| ROS-FValue0.25-MLPC | 0.88238 | 0.01745 |
| RUS-FValue0.5-SVC | 0.88203 | 0.02921 |
| RUS-MI0.5-SVC | 0.88167 | 0.02997 |
| ROS-MI0.25-SVC | 0.86905 | 0.03035 |
| RUS-MI0.25-SVC | 0.85781 | 0.02149 |
| FS-FValue0.25-SVC | 0.85465 | 0.01734 |
| ROS-FValue0.25-SVC | 0.85465 | 0.01734 |
| RUS-FValue0.25-SVC | 0.85429 | 0.01809 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SFS0.75-RFC | 0.98806 | 0.00602 |
| ROS-SFS1-MLPC | 0.98595 | 0.00676 |
| ROS-SFS1-RFC | 0.98525 | 0.00466 |
| ROS-SFS0.5-RFC | 0.98490 | 0.00325 |
| ROS-SBS1-MLPC | 0.98490 | 0.00644 |
| ROS-SBS1-RFC | 0.98420 | 0.00399 |
| ROS-SFS0.75-MLPC | 0.98420 | 0.00599 |
| ROS-SFS0.75-GBC | 0.98314 | 0.00615 |
| ROS-SBS0.75-RFC | 0.98210 | 0.00559 |
| ROS-SBS1-GBC | 0.98174 | 0.00708 |
| ROS-SFS1-GBC | 0.98174 | 0.00708 |
| ROS-SFS0.75-KNC | 0.98069 | 0.00598 |
| ROS-SFS0.5-KNC | 0.98069 | 0.00444 |
| ROS-SBS0.75-GBC | 0.97999 | 0.00504 |
| ROS-SBS0.5-RFC | 0.97999 | 0.00634 |
| ROS-SFS0.5-MLPC | 0.97963 | 0.00663 |
| ROS-SFS0.5-GBC | 0.97788 | 0.00516 |
| ROS-SBS1-KNC | 0.97613 | 0.00750 |
| ROS-SFS1-KNC | 0.97613 | 0.00750 |
| ROS-SBS0.5-GBC | 0.97507 | 0.00731 |
| ROS-SBS0.75-KNC | 0.97297 | 0.00806 |
| ROS-SBS0.25-RFC | 0.97296 | 0.00540 |
| ROS-SBS0.5-KNC | 0.97050 | 0.01484 |
| ROS-SBS0.75-MLPC | 0.96700 | 0.01106 |
| ROS-SBS0.25-KNC | 0.96559 | 0.00691 |
| ROS-SBS0.25-GBC | 0.96524 | 0.00475 |
| ROS-SBS0.5-MLPC | 0.96384 | 0.01188 |
| ROS-SFS0.25-RFC | 0.95436 | 0.01809 |
| ROS-SFS0.25-KNC | 0.94839 | 0.02117 |
| ROS-SFS0.25-GBC | 0.94734 | 0.01761 |
| ROS-SBS0.25-MLPC | 0.94453 | 0.02034 |
| ROS-SFS0.25-MLPC | 0.93997 | 0.02194 |

### E.4.3. Prueba 3

## E.5. Meñique

### E.5.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-MI0.75-KNC | 0.98420 | 0.00638 |
| RUS-FValue0.75-KNC | 0.98420 | 0.00793 |
| ROS-FValue0.75-KNC | 0.98385 | 0.00697 |
| FS-MI0.75-KNC | 0.98315 | 0.00594 |
| FS-FValue0.75-KNC | 0.98314 | 0.00644 |
| RUS-MI0.5-KNC | 0.98279 | 0.00642 |
| RUS-MI0.75-KNC | 0.98244 | 0.00618 |
| FS-FValue0.5-KNC | 0.98244 | 0.00295 |
| ROS-FValue0.5-KNC | 0.98244 | 0.00370 |
| RUS-FValue0.5-KNC | 0.98244 | 0.00472 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI0.75-SGDC | 0.85570 | 0.02077 |
| FS-FValue0.25-NC | 0.85570 | 0.01223 |
| ROS-FValue0.25-NC | 0.85570 | 0.01223 |
| RUS-FValue0.25-NC | 0.85570 | 0.01223 |
| ROS-MI0.75-NC | 0.85499 | 0.01199 |
| ROS-MI0.75-BNB | 0.85465 | 0.01477 |
| RUS-MI0.75-BNB | 0.85465 | 0.01477 |
| RUS-MI0.75-NC | 0.85464 | 0.01167 |
| FS-FValue0.75-SGDC | 0.85464 | 0.01738 |
| FS-FValue1-BNB | 0.85429 | 0.01353 |
| FS-MI1-BNB | 0.85429 | 0.01353 |
| FS-MI0.75-NC | 0.85429 | 0.01270 |
| ROS-FValue1-RC | 0.85394 | 0.01672 |
| ROS-MI1-RC | 0.85394 | 0.01672 |
| FS-FValue1-RC | 0.85359 | 0.01684 |
| FS-MI1-RC | 0.85359 | 0.01684 |
| RUS-MI0.75-RC | 0.85359 | 0.01601 |
| ROS-MI0.5-RC | 0.85359 | 0.01587 |
| ROS-FValue0.75-RC | 0.85359 | 0.01421 |
| ROS-MI0.75-RC | 0.85324 | 0.01578 |
| ROS-MI0.25-RC | 0.85289 | 0.01642 |
| RUS-FValue0.75-RC | 0.85289 | 0.01394 |
| RUS-MI0.5-RC | 0.85254 | 0.01514 |
| FS-MI0.5-SGDC | 0.85254 | 0.01149 |
| RUS-FValue1-RC | 0.85219 | 0.01627 |
| RUS-MI1-RC | 0.85219 | 0.01627 |
| RUS-FValue0.5-BNB | 0.85219 | 0.01425 |
| FS-MI0.5-RC | 0.85219 | 0.01632 |
| FS-MI0.75-RC | 0.85219 | 0.01531 |
| ROS-FValue0.5-BNB | 0.85184 | 0.01383 |
| ROS-FValue1-BNB | 0.85149 | 0.01646 |
| ROS-MI1-BNB | 0.85149 | 0.01646 |
| RUS-FValue1-BNB | 0.85149 | 0.01646 |
| RUS-MI1-BNB | 0.85149 | 0.01646 |
| FS-FValue0.75-BNB | 0.85149 | 0.01489 |
| RUS-MI0.5-NC | 0.85148 | 0.01152 |
| FS-FValue0.25-SGDC | 0.85148 | 0.00882 |
| ROS-MI0.5-BNB | 0.85113 | 0.01364 |
| ROS-MI0.25-NC | 0.85079 | 0.01800 |
| FS-MI0.5-BNB | 0.85079 | 0.01617 |
| FS-FValue0.75-RC | 0.85078 | 0.01378 |
| RUS-FValue0.25-SGDC | 0.85078 | 0.01117 |
| ROS-FValue0.25-SGDC | 0.85078 | 0.01095 |
| RUS-MI0.5-BNB | 0.85008 | 0.01248 |
| RUS-MI0.25-RC | 0.85008 | 0.01204 |
| ROS-FValue0.5-SGDC | 0.85008 | 0.01551 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| RUS-FValue0.75-BNB | 0.84973 | 0.01585 |
| ROS-FValue0.75-BNB | 0.84973 | 0.01489 |
| FS-FValue0.5-BNB | 0.84938 | 0.01552 |
| ROS-MI0.25-BNB | 0.84868 | 0.01662 |
| FS-FValue0.25-BNB | 0.84833 | 0.01387 |
| RUS-FValue0.5-RC | 0.84832 | 0.01370 |
| FS-MI0.5-NC | 0.84832 | 0.01173 |
| FS-FValue0.5-RC | 0.84797 | 0.01370 |
| ROS-FValue0.5-RC | 0.84797 | 0.01315 |
| RUS-FValue0.25-BNB | 0.84762 | 0.01390 |
| ROS-FValue0.25-BNB | 0.84727 | 0.01339 |
| FS-MI0.25-SGDC | 0.84657 | 0.01145 |
| FS-FValue1-NC | 0.84622 | 0.01119 |
| FS-MI1-NC | 0.84622 | 0.01119 |
| ROS-MI0.5-NC | 0.84621 | 0.00988 |
| ROS-FValue1-NC | 0.84587 | 0.01202 |
| ROS-MI1-NC | 0.84587 | 0.01202 |
| RUS-MI0.75-SGDC | 0.84552 | 0.02023 |
| FS-MI0.25-BNB | 0.84517 | 0.01425 |
| RUS-FValue1-NC | 0.84516 | 0.01271 |
| RUS-MI1-NC | 0.84516 | 0.01271 |
| RUS-FValue0.75-SGDC | 0.84482 | 0.02533 |
| FS-MI0.25-NC | 0.84481 | 0.01286 |
| RUS-MI0.25-BNB | 0.84446 | 0.01439 |
| ROS-MI0.25-SGDC | 0.84342 | 0.02140 |
| RUS-MI0.25-NC | 0.84271 | 0.01870 |
| FS-MI0.25-RC | 0.84235 | 0.00818 |
| RUS-MI0.25-SGDC | 0.84130 | 0.01441 |
| FS-FValue0.5-SGDC | 0.83743 | 0.01597 |
| FS-FValue0.75-NC | 0.83709 | 0.00990 |
| ROS-FValue0.75-NC | 0.83673 | 0.01048 |
| RUS-FValue0.75-NC | 0.83533 | 0.01150 |
| RUS-FValue0.5-SGDC | 0.83428 | 0.02403 |

### E.5.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI1-RFC | 0.99122 | 0.00415 |
| FS-FValue1-RFC | 0.99017 | 0.00452 |
| FS-FValue1-ABC | 0.98982 | 0.00301 |
| FS-MI1-ABC | 0.98982 | 0.00301 |
| ROS-FValue1-RFC | 0.98911 | 0.00436 |
| FS-FValue1-GBC | 0.98911 | 0.00340 |
| FS-MI1-GBC | 0.98911 | 0.00358 |
| ROS-FValue1-GBC | 0.98876 | 0.00263 |
| ROS-MI1-GBC | 0.98876 | 0.00263 |
| ROS-MI1-RFC | 0.98876 | 0.00439 |
| FS-MI0.75-GBC | 0.98841 | 0.00361 |
| FS-MI0.75-RFC | 0.98806 | 0.00449 |
| RUS-FValue1-RFC | 0.98806 | 0.00450 |
| RUS-FValue1-GBC | 0.98806 | 0.00340 |
| RUS-MI1-GBC | 0.98806 | 0.00340 |
| ROS-FValue0.75-RFC | 0.98806 | 0.00514 |
| FS-FValue0.5-GBC | 0.98806 | 0.00407 |
| RUS-MI1-RFC | 0.98771 | 0.00430 |
| ROS-MI0.75-GBC | 0.98771 | 0.00385 |
| FS-FValue1-MLPC | 0.98771 | 0.00248 |
| FS-FValue0.75-GBC | 0.98736 | 0.00489 |
| RUS-FValue1-ABC | 0.98736 | 0.00421 |
| RUS-MI1-ABC | 0.98736 | 0.00421 |
| ROS-FValue0.75-GBC | 0.98701 | 0.00516 |
| FS-FValue0.75-RFC | 0.98701 | 0.00466 |
| ROS-FValue1-MLPC | 0.98701 | 0.00263 |
| ROS-MI1-MLPC | 0.98666 | 0.00326 |
| FS-FValue0.5-RFC | 0.98631 | 0.00696 |
| ROS-MI0.75-RFC | 0.98631 | 0.00357 |
| FS-FValue1-BaggingClassifier | 0.98631 | 0.00435 |
| ROS-FValue0.5-GBC | 0.98631 | 0.00514 |
| RUS-FValue0.75-GBC | 0.98631 | 0.00450 |
| ROS-FValue1-ABC | 0.98631 | 0.00258 |
| ROS-MI1-ABC | 0.98631 | 0.00258 |
| ROS-MI1-BaggingClassifier | 0.98596 | 0.00532 |
| ROS-FValue1-BaggingClassifier | 0.98596 | 0.00332 |
| RUS-MI1-MLPC | 0.98596 | 0.00248 |
| RUS-FValue1-MLPC | 0.98560 | 0.00281 |
| RUS-FValue0.75-RFC | 0.98560 | 0.00303 |
| ROS-FValue0.5-RFC | 0.98525 | 0.00452 |
| RUS-MI0.75-GBC | 0.98490 | 0.00439 |
| RUS-FValue0.5-RFC | 0.98455 | 0.00301 |
| RUS-FValue0.5-GBC | 0.98455 | 0.00341 |
| ROS-FValue0.75-MLPC | 0.98455 | 0.00622 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.75-BaggingClassifier | 0.98420 | 0.00429 |
| RUS-MI0.75-RFC | 0.98420 | 0.00368 |
| FS-MI1-BaggingClassifier | 0.98420 | 0.00351 |
| FS-MI1-MLPC | 0.98420 | 0.00484 |
| ROS-MI0.75-BaggingClassifier | 0.98385 | 0.00340 |
| RUS-MI1-BaggingClassifier | 0.98385 | 0.00513 |
| FS-MI0.75-ABC | 0.98385 | 0.00561 |
| ROS-FValue0.75-BaggingClassifier | 0.98280 | 0.00475 |
| FS-MI0.75-BaggingClassifier | 0.98280 | 0.00559 |
| FS-MI0.5-RFC | 0.98244 | 0.00544 |
| RUS-FValue1-BaggingClassifier | 0.98244 | 0.00294 |
| ROS-MI0.75-MLPC | 0.98244 | 0.00648 |
| RUS-MI0.75-BaggingClassifier | 0.98209 | 0.00463 |
| FS-FValue0.5-ABC | 0.98209 | 0.00322 |
| ROS-FValue0.5-BaggingClassifier | 0.98209 | 0.00172 |
| RUS-FValue0.5-BaggingClassifier | 0.98174 | 0.00326 |
| FS-FValue0.75-MLPC | 0.98174 | 0.00717 |
| RUS-MI0.75-MLPC | 0.98174 | 0.00635 |
| FS-FValue0.5-BaggingClassifier | 0.98139 | 0.00262 |
| FS-MI0.5-GBC | 0.98139 | 0.00774 |
| ROS-MI0.75-ABC | 0.98139 | 0.00439 |
| RUS-MI0.5-RFC | 0.98139 | 0.00529 |
| ROS-MI0.5-RFC | 0.98139 | 0.00791 |
| FS-MI0.75-MLPC | 0.98104 | 0.00732 |
| RUS-FValue0.75-BaggingClassifier | 0.98069 | 0.00457 |
| RUS-FValue0.75-MLPC | 0.98069 | 0.00648 |
| ROS-MI0.5-GBC | 0.98034 | 0.00475 |
| RUS-MI0.5-GBC | 0.97999 | 0.00551 |
| RUS-MI0.75-ABC | 0.97999 | 0.00635 |
| ROS-MI0.5-BaggingClassifier | 0.97998 | 0.00363 |
| FS-MI0.5-BaggingClassifier | 0.97963 | 0.00583 |
| RUS-MI0.5-BaggingClassifier | 0.97963 | 0.00326 |
| ROS-MI1-DecisionTreeClassifier | 0.97963 | 0.00865 |
| FS-FValue0.75-ABC | 0.97963 | 0.00725 |
| ROS-FValue0.75-ABC | 0.97963 | 0.00691 |
| ROS-MI0.75-DecisionTreeClassifier | 0.97928 | 0.00406 |
| RUS-MI0.5-ABC | 0.97858 | 0.00582 |
| RUS-FValue0.75-ABC | 0.97823 | 0.00681 |
| FS-FValue1-DecisionTreeClassifier | 0.97823 | 0.00439 |
| ROS-FValue0.5-ABC | 0.97823 | 0.00529 |
| RUS-MI1-DecisionTreeClassifier | 0.97753 | 0.00810 |
| FS-FValue0.5-MLPC | 0.97753 | 0.00515 |
| FS-MI0.5-ABC | 0.97718 | 0.00701 |
| FS-MI0.75-DecisionTreeClassifier | 0.97718 | 0.00430 |
| RUS-FValue0.5-ABC | 0.97683 | 0.00623 |
| ROS-FValue1-DecisionTreeClassifier | 0.97682 | 0.00661 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI1-DecisionTreeClassifier | 0.97682 | 0.00538 |
| RUS-FValue0.75-DecisionTreeClassifier | 0.97612 | 0.00263 |
| ROS-FValue0.75-DecisionTreeClassifier | 0.97577 | 0.00537 |
| FS-FValue0.75-DecisionTreeClassifier | 0.97577 | 0.00341 |
| ROS-MI0.25-RFC | 0.97543 | 0.01604 |
| RUS-FValue1-DecisionTreeClassifier | 0.97542 | 0.00710 |
| RUS-MI0.5-DecisionTreeClassifier | 0.97507 | 0.00408 |
| ROS-FValue0.5-MLPC | 0.97472 | 0.00493 |
| RUS-FValue0.5-DecisionTreeClassifier | 0.97471 | 0.00626 |
| ROS-MI0.5-ABC | 0.97437 | 0.00361 |
| ROS-FValue0.5-DecisionTreeClassifier | 0.97401 | 0.00408 |
| RUS-MI0.75-DecisionTreeClassifier | 0.97332 | 0.00435 |
| RUS-MI0.5-MLPC | 0.97331 | 0.00733 |
| ROS-MI0.5-DecisionTreeClassifier | 0.97296 | 0.00424 |
| RUS-FValue0.5-MLPC | 0.97261 | 0.00575 |
| FS-FValue0.5-DecisionTreeClassifier | 0.97156 | 0.00652 |
| FS-MI0.5-DecisionTreeClassifier | 0.97121 | 0.00724 |
| ROS-MI0.5-MLPC | 0.97085 | 0.00852 |
| RUS-MI0.25-RFC | 0.97016 | 0.01458 |
| ROS-MI0.25-GBC | 0.97016 | 0.01879 |
| ROS-FValue1-ExtraTreeClassifier | 0.97016 | 0.00761 |
| ROS-MI0.25-BaggingClassifier | 0.96981 | 0.01290 |
| FS-MI0.5-MLPC | 0.96629 | 0.01813 |
| ROS-FValue1-SVC | 0.96594 | 0.00654 |
| ROS-MI1-SVC | 0.96594 | 0.00654 |
| FS-FValue1-SVC | 0.96559 | 0.00873 |
| FS-MI1-SVC | 0.96559 | 0.00873 |
| FS-FValue1-ExtraTreeClassifier | 0.96489 | 0.00331 |
| RUS-FValue1-SVC | 0.96348 | 0.00680 |
| RUS-MI1-SVC | 0.96348 | 0.00680 |
| FS-MI1-ExtraTreeClassifier | 0.96348 | 0.00436 |
| ROS-MI1-ExtraTreeClassifier | 0.96243 | 0.00452 |
| ROS-MI0.5-ExtraTreeClassifier | 0.96137 | 0.00975 |
| ROS-FValue0.5-ExtraTreeClassifier | 0.96137 | 0.00511 |
| RUS-MI0.25-GBC | 0.96034 | 0.02217 |
| FS-MI0.25-RFC | 0.96033 | 0.01965 |
| FS-FValue0.5-ExtraTreeClassifier | 0.96032 | 0.00393 |
| RUS-FValue0.5-ExtraTreeClassifier | 0.96032 | 0.00561 |
| ROS-MI0.25-DecisionTreeClassifier | 0.95998 | 0.01542 |
| RUS-MI0.5-ExtraTreeClassifier | 0.95926 | 0.01216 |
| RUS-FValue1-ExtraTreeClassifier | 0.95857 | 0.00876 |
| RUS-MI0.25-BaggingClassifier | 0.95857 | 0.01533 |
| FS-MI0.25-BaggingClassifier | 0.95787 | 0.01800 |
| RUS-MI1-ExtraTreeClassifier | 0.95611 | 0.00729 |
| FS-MI0.5-ExtraTreeClassifier | 0.95611 | 0.00446 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI0.75-ExtraTreeClassifier | 0.95575 | 0.00551 |
| ROS-FValue0.75-ExtraTreeClassifier | 0.95575 | 0.01081 |
| ROS-MI0.25-ABC | 0.95472 | 0.03116 |
| ROS-MI0.75-ExtraTreeClassifier | 0.95366 | 0.00747 |
| ROS-MI0.25-ExtraTreeClassifier | 0.95261 | 0.02122 |
| RUS-FValue0.75-ExtraTreeClassifier | 0.95224 | 0.01234 |
| FS-MI0.25-GBC | 0.95120 | 0.02144 |
| RUS-MI0.75-ExtraTreeClassifier | 0.95049 | 0.00870 |
| ROS-FValue0.25-RFC | 0.94979 | 0.00910 |
| FS-FValue0.25-RFC | 0.94979 | 0.00678 |
| RUS-MI0.25-DecisionTreeClassifier | 0.94944 | 0.01656 |
| RUS-FValue0.25-RFC | 0.94804 | 0.00677 |
| FS-FValue0.75-ExtraTreeClassifier | 0.94768 | 0.00649 |
| RUS-FValue0.25-GBC | 0.94453 | 0.00650 |
| FS-MI0.25-DecisionTreeClassifier | 0.94452 | 0.01069 |
| FS-FValue0.25-GBC | 0.94382 | 0.00365 |
| ROS-FValue0.25-GBC | 0.94312 | 0.00670 |
| RUS-FValue0.25-BaggingClassifier | 0.94277 | 0.00677 |
| RUS-MI0.25-ABC | 0.94244 | 0.03527 |
| ROS-FValue0.25-BaggingClassifier | 0.94207 | 0.01182 |
| RUS-MI0.25-ExtraTreeClassifier | 0.94172 | 0.02693 |
| FS-MI0.25-ExtraTreeClassifier | 0.93997 | 0.01963 |
| FS-FValue0.25-BaggingClassifier | 0.93821 | 0.00921 |
| ROS-MI0.25-MLPC | 0.93717 | 0.04775 |
| RUS-FValue0.25-DecisionTreeClassifier | 0.93329 | 0.01007 |
| FS-FValue0.25-DecisionTreeClassifier | 0.93119 | 0.00978 |
| ROS-FValue0.25-DecisionTreeClassifier | 0.92838 | 0.01613 |
| FS-FValue0.5-SVC | 0.92696 | 0.02194 |
| FS-MI0.25-ABC | 0.92452 | 0.03841 |
| ROS-FValue0.5-SVC | 0.92171 | 0.03088 |
| FS-FValue0.25-ExtraTreeClassifier | 0.92171 | 0.00851 |
| ROS-FValue0.25-ABC | 0.92170 | 0.00261 |
| FS-FValue0.25-ABC | 0.92135 | 0.00278 |
| RUS-FValue0.25-ABC | 0.91995 | 0.00580 |
| FS-MI0.5-SVC | 0.91926 | 0.03998 |
| ROS-FValue0.25-ExtraTreeClassifier | 0.91889 | 0.01224 |
| ROS-MI0.25-SVC | 0.91575 | 0.04200 |
| RUS-FValue0.25-ExtraTreeClassifier | 0.91082 | 0.01270 |
| FS-FValue0.75-SVC | 0.91012 | 0.02715 |
| RUS-MI0.25-MLPC | 0.90910 | 0.06339 |
| RUS-FValue0.5-SVC | 0.89362 | 0.02828 |
| RUS-MI0.25-SVC | 0.88662 | 0.04626 |
| FS-MI0.75-SVC | 0.88484 | 0.02215 |
| FS-MI0.25-SVC | 0.88168 | 0.03912 |
| FS-MI0.25-MLPC | 0.87852 | 0.06340 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-FValue0.75-SVC | 0.87783 | 0.02557 |
| RUS-MI0.5-SVC | 0.87502 | 0.03726 |
| ROS-MI0.75-SVC | 0.87326 | 0.02608 |
| RUS-FValue0.75-SVC | 0.87221 | 0.02102 |
| RUS-MI0.75-SVC | 0.87080 | 0.02864 |
| FS-FValue0.25-MLPC | 0.86553 | 0.01508 |
| ROS-MI0.5-SVC | 0.86237 | 0.01535 |
| FS-FValue0.25-SVC | 0.85991 | 0.01300 |
| ROS-FValue0.25-SVC | 0.85956 | 0.01354 |
| RUS-FValue0.25-SVC | 0.85921 | 0.01421 |
| ROS-FValue0.25-MLPC | 0.85780 | 0.01198 |
| RUS-FValue0.25-MLPC | 0.85570 | 0.01254 |

| | Mean accuracy | Std accuracy |
|---|---|---|
| ROS-SBS1-RFC | 0.99017 | 0.00465 |
| ROS-SFS1-RFC | 0.99017 | 0.00452 |
| ROS-SFS1-GBC | 0.98912 | 0.00281 |
| ROS-SBS1-GBC | 0.98876 | 0.00361 |
| ROS-SBS0.75-RFC | 0.98771 | 0.00657 |
| ROS-SFS0.75-GBC | 0.98771 | 0.00294 |
| ROS-SBS0.75-GBC | 0.98701 | 0.00516 |
| ROS-SFS1-MLPC | 0.98701 | 0.00238 |
| ROS-SBS1-MLPC | 0.98666 | 0.00344 |
| ROS-SFS0.75-RFC | 0.98560 | 0.00463 |
| ROS-SBS0.5-RFC | 0.98385 | 0.00392 |
| ROS-SFS0.75-MLPC | 0.98350 | 0.00540 |
| ROS-SBS0.5-GBC | 0.98279 | 0.00342 |
| ROS-SBS0.5-KNC | 0.98244 | 0.00445 |
| ROS-SBS0.75-MLPC | 0.98033 | 0.00716 |
| ROS-SBS0.75-KNC | 0.97998 | 0.01200 |
| ROS-SFS0.5-RFC | 0.97963 | 0.00927 |
| ROS-SBS1-KNC | 0.97858 | 0.01027 |
| ROS-SFS1-KNC | 0.97858 | 0.01027 |
| ROS-SFS0.75-KNC | 0.97823 | 0.00880 |
| ROS-SFS0.5-GBC | 0.97612 | 0.01688 |
| ROS-SBS0.25-RFC | 0.97367 | 0.00674 |
| ROS-SFS0.5-KNC | 0.97260 | 0.01160 |
| ROS-SBS0.5-MLPC | 0.97226 | 0.00539 |
| ROS-SBS0.25-KNC | 0.97156 | 0.00405 |
| ROS-SFS0.5-MLPC | 0.96733 | 0.02091 |
| ROS-SBS0.25-GBC | 0.96630 | 0.00936 |
| ROS-SFS0.25-RFC | 0.96172 | 0.02341 |
| ROS-SFS0.25-KNC | 0.96065 | 0.03199 |
| ROS-SBS0.25-MLPC | 0.95611 | 0.00961 |
| ROS-SFS0.25-GBC | 0.95259 | 0.02842 |
| ROS-SFS0.25-MLPC | 0.94099 | 0.04242 |

### E.5.3. Prueba 3

## E.6. Movimiento versión 1

### E.6.1. Prueba 1

| | Mean precision | Std precision |
|---|---|---|
| ROS-MI0.5-KNC | 0.77467 | 0.00738 |
| FS-MI0.5-KNC | 0.77416 | 0.00525 |
| RUS-MI0.5-KNC | 0.76583 | 0.00589 |
| FS-MI0.25-KNC | 0.75761 | 0.00224 |
| ROS-MI0.25-KNC | 0.75219 | 0.00420 |
| RUS-MI0.25-KNC | 0.74708 | 0.00285 |
| FS-FValue0.5-KNC | 0.72891 | 0.00711 |
| ROS-FValue0.5-KNC | 0.72495 | 0.00964 |
| RUS-FValue0.5-KNC | 0.72042 | 0.00880 |
| FS-MI0.75-KNC | 0.71207 | 0.00637 |

### E.6.2. Prueba 2

| | Mean precision | Std precision |
|---|---|---|
| ROS-MI0.75-RFC | 0.83532 | 0.00280 |
| FS-MI0.75-RFC | 0.83425 | 0.00552 |
| ROS-FValue0.75-RFC | 0.82852 | 0.00383 |
| FS-FValue0.75-RFC | 0.82829 | 0.00636 |
| ROS-FValue1-RFC | 0.82687 | 0.00572 |
| ROS-MI1-RFC | 0.82639 | 0.00558 |
| ROS-MI0.5-RFC | 0.82490 | 0.00616 |
| FS-FValue1-RFC | 0.82480 | 0.00703 |
| RUS-MI0.75-RFC | 0.82439 | 0.00877 |
| FS-MI1-RFC | 0.82423 | 0.00403 |
| FS-MI0.5-RFC | 0.82283 | 0.00778 |
| RUS-FValue0.75-RFC | 0.81955 | 0.00553 |
| RUS-MI1-RFC | 0.81497 | 0.00339 |
| RUS-MI0.5-RFC | 0.81485 | 0.00601 |
| RUS-FValue1-RFC | 0.81362 | 0.00312 |
| ROS-MI0.75-BaggingClassifier | 0.80466 | 0.00719 |
| FS-MI0.75-BaggingClassifier | 0.80268 | 0.00485 |
| ROS-FValue0.75-BaggingClassifier | 0.80147 | 0.00641 |
| FS-FValue0.75-BaggingClassifier | 0.79842 | 0.00582 |
| ROS-MI1-BaggingClassifier | 0.79708 | 0.00393 |
| ROS-FValue1-BaggingClassifier | 0.79691 | 0.00580 |
| FS-MI1-BaggingClassifier | 0.79611 | 0.00582 |
| FS-FValue1-BaggingClassifier | 0.79522 | 0.00517 |
| ROS-MI0.5-BaggingClassifier | 0.79481 | 0.00710 |
| FS-MI0.5-BaggingClassifier | 0.79274 | 0.00693 |
| RUS-FValue0.75-BaggingClassifier | 0.79003 | 0.00468 |
| RUS-FValue1-BaggingClassifier | 0.78809 | 0.00179 |
| RUS-MI0.75-BaggingClassifier | 0.78732 | 0.00644 |
| RUS-MI0.5-BaggingClassifier | 0.78609 | 0.00363 |
| RUS-MI1-BaggingClassifier | 0.78208 | 0.00520 |
| FS-FValue0.5-RFC | 0.77848 | 0.00351 |
| ROS-FValue0.5-RFC | 0.77673 | 0.00445 |
| FS-MI0.25-RFC | 0.77006 | 0.00414 |
| ROS-MI0.25-RFC | 0.76777 | 0.00170 |
| RUS-FValue0.5-RFC | 0.76676 | 0.00550 |
| RUS-MI0.25-RFC | 0.75956 | 0.00524 |
| FS-FValue0.5-BaggingClassifier | 0.75246 | 0.00338 |
| FS-MI0.75-MLPC | 0.74927 | 0.00788 |
| FS-MI0.25-BaggingClassifier | 0.74889 | 0.00409 |
| ROS-FValue0.5-BaggingClassifier | 0.74888 | 0.00468 |
| ROS-MI0.75-GBC | 0.74866 | 0.00480 |
| ROS-MI0.75-MLPC | 0.74836 | 0.00365 |
| ROS-FValue1-GBC | 0.74726 | 0.00519 |
| ROS-MI1-GBC | 0.74717 | 0.00519 |
| FS-FValue0.75-MLPC | 0.74635 | 0.00579 |
| FS-MI0.75-GBC | 0.74552 | 0.00412 |
| RUS-MI0.75-GBC | 0.74503 | 0.00400 |
| ROS-FValue0.75-MLPC | 0.74497 | 0.00496 |

## E.7. Movimiento versión 2

### E.7.1. Prueba 1

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-FValue0.25-KNC | 0.88979 | 0.01925 |
| RUS-FValue0.25-KNC | 0.88710 | 0.02136 |
| ROS-FValue0.25-KNC | 0.87770 | 0.01144 |
| FS-FValue0.5-KNC | 0.86827 | 0.02841 |
| RUS-FValue0.5-KNC | 0.86555 | 0.02961 |
| ROS-FValue0.5-KNC | 0.86289 | 0.03496 |
| FS-MI0.25-KNC | 0.86020 | 0.02770 |
| ROS-MI0.25-KNC | 0.85754 | 0.02334 |
| RUS-MI0.25-KNC | 0.85482 | 0.03090 |
| FS-FValue0.75-KNC | 0.83067 | 0.02077 |
| ROS-FValue0.75-KNC | 0.82934 | 0.02401 |
| RUS-FValue0.75-KNC | 0.82799 | 0.02881 |
| FS-MI0.5-KNC | 0.82794 | 0.01264 |
| RUS-MI0.5-KNC | 0.81184 | 0.01188 |
| ROS-MI0.5-KNC | 0.79436 | 0.02493 |
| FS-MI0.75-KNC | 0.79170 | 0.02760 |
| RUS-MI0.75-KNC | 0.77690 | 0.02034 |
| ROS-MI0.75-KNC | 0.73924 | 0.03301 |
| FS-MI0.75-RC | 0.73118 | 0.03627 |
| ROS-FValue1-RC | 0.72850 | 0.04503 |
| ROS-MI1-RC | 0.72850 | 0.04503 |
| FS-MI0.5-RC | 0.72850 | 0.02895 |
| RUS-FValue1-RC | 0.72715 | 0.04318 |
| RUS-MI1-RC | 0.72715 | 0.04318 |
| ROS-MI0.75-RC | 0.72714 | 0.04507 |
| ROS-MI0.5-RC | 0.72714 | 0.04160 |
| FS-FValue1-RC | 0.72445 | 0.03659 |
| FS-MI1-RC | 0.72445 | 0.03659 |
| ROS-FValue0.75-RC | 0.72313 | 0.04514 |
| RUS-MI0.5-RC | 0.72313 | 0.04029 |
| RUS-MI0.75-RC | 0.72179 | 0.03959 |
| FS-FValue0.75-RC | 0.72043 | 0.03408 |
| RUS-FValue0.75-RC | 0.71641 | 0.04474 |
| ROS-MI0.25-NC | 0.71233 | 0.03132 |
| RUS-MI0.25-NC | 0.71099 | 0.02815 |
| FS-MI0.25-NC | 0.71098 | 0.02627 |
| ROS-FValue0.25-RC | 0.70830 | 0.02995 |
| FS-FValue0.25-RC | 0.70830 | 0.02539 |
| FS-MI0.25-RC | 0.70703 | 0.04103 |
| RUS-MI0.75-SGDC | 0.70700 | 0.04862 |
| FS-FValue1-SGDC | 0.70561 | 0.03273 |
| RUS-FValue0.25-RC | 0.70560 | 0.02996 |
| FS-MI0.5-SGDC | 0.70558 | 0.03842 |
| FS-MI0.75-SGDC | 0.70298 | 0.01086 |
| ROS-MI0.25-RC | 0.70297 | 0.04690 |
| ROS-FValue1-SGDC | 0.70153 | 0.04890 |

### E.7.2. Prueba 2

| | Mean accuracy | Std accuracy |
|---|---|---|
| FS-MI0.75-RFC | 0.93416 | 0.02761 |
| ROS-MI0.5-RFC | 0.93146 | 0.02178 |
| RUS-MI0.5-RFC | 0.92744 | 0.03403 |
| FS-MI0.5-RFC | 0.92744 | 0.01912 |
| ROS-MI0.75-RFC | 0.92478 | 0.03153 |
| ROS-FValue0.75-RFC | 0.92475 | 0.02411 |
| FS-FValue0.75-RFC | 0.92342 | 0.01870 |
| FS-MI1-RFC | 0.92341 | 0.02830 |
| RUS-MI1-RFC | 0.92340 | 0.03247 |
| ROS-FValue1-RFC | 0.92340 | 0.02144 |
| ROS-MI1-RFC | 0.92207 | 0.02985 |
| FS-FValue1-RFC | 0.91939 | 0.03416 |
| FS-MI0.25-RFC | 0.91935 | 0.01123 |
| RUS-MI0.75-RFC | 0.91800 | 0.02734 |
| RUS-FValue0.75-RFC | 0.91533 | 0.03640 |
| ROS-MI0.25-RFC | 0.91533 | 0.01727 |
| RUS-FValue1-RFC | 0.91401 | 0.04184 |
| RUS-MI0.75-GBC | 0.91130 | 0.02218 |
| FS-MI0.5-GBC | 0.91130 | 0.01661 |
| ROS-FValue0.75-GBC | 0.90995 | 0.01380 |
| ROS-MI0.5-GBC | 0.90995 | 0.01088 |
| FS-FValue1-GBC | 0.90863 | 0.01366 |
| RUS-MI0.25-RFC | 0.90862 | 0.02265 |
| FS-MI0.75-GBC | 0.90862 | 0.01080 |
| RUS-MI0.5-GBC | 0.90860 | 0.01928 |
| FS-MI1-GBC | 0.90728 | 0.00967 |
| ROS-FValue1-GBC | 0.90727 | 0.01424 |
| RUS-FValue0.75-GBC | 0.90727 | 0.01711 |
| ROS-MI0.25-GBC | 0.90726 | 0.02136 |
| FS-MI0.25-GBC | 0.90592 | 0.02032 |
| ROS-FValue1-BaggingClassifier | 0.90459 | 0.01909 |
| RUS-MI0.25-GBC | 0.90459 | 0.02692 |
| ROS-MI0.75-GBC | 0.90324 | 0.01558 |
| FS-FValue0.75-GBC | 0.90323 | 0.01242 |
| RUS-FValue1-GBC | 0.90056 | 0.02588 |
| ROS-MI1-GBC | 0.90056 | 0.01810 |
| RUS-MI1-GBC | 0.90055 | 0.01910 |
| FS-FValue0.5-RFC | 0.89922 | 0.03571 |
| ROS-FValue0.5-RFC | 0.89921 | 0.03283 |
| ROS-MI1-BaggingClassifier | 0.89654 | 0.02176 |
| ROS-MI0.75-BaggingClassifier | 0.89654 | 0.01490 |
| ROS-FValue0.75-BaggingClassifier | 0.89651 | 0.00907 |
| RUS-FValue0.25-RFC | 0.89518 | 0.01916 |
| FS-FValue0.25-RFC | 0.89383 | 0.01956 |
| RUS-MI0.25-BaggingClassifier | 0.89382 | 0.01866 |
| ROS-FValue0.25-RFC | 0.89250 | 0.01630 |
| RUS-FValue0.5-RFC | 0.89249 | 0.02810 |
| ROS-MI0.5-BaggingClassifier | 0.89116 | 0.01807 |

| | Mean accuracy | Std accuracy |
|---|---|---|

| | Mean accuracy | Std accuracy |
|---|---|---|