UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA ELECTRÓNICA Y AUTOMÁTICA INDUSTRIAL

"DESARROLLO DE UN MODELO DE MACHINE LEARNING PARA CLASIFICACIÓN DE TAREAS DURANTE LA MARCHA"

TRABAJO FIN DE GRADO

Junio -2024

AUTOR: Merche Fernández Terol

DIRECTOR/ES: Nicolás M. García Aracil

David Martínez Pascual

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

RESUMEN

El proyecto se centra en el análisis de la marcha humana y el desarrollo de un clasificador para identificar las distintas actividades durante la marcha, como andar en plano, subir rampa, bajar rampa y subir escaleras. Para ello, se empleó Machine Learning con el fin de crear un modelo que utilice los ángulos de las articulaciones de los miembros inferiores como datos de entrada. Este enfoque permitirá diseñar un sistema capaz de reconocer y clasificar de manera precisa las acciones realizadas durante la marcha. El objetivo es entrenar y evaluar el modelo, analizando su precisión y la importancia de las características de entrada. El trabajo detallará el proceso de desarrollo del modelo, los resultados obtenidos y las conclusiones de la investigación.

Palabras clave:

Análisis de la marcha, clasificación de actividades, Machine Learning, ángulos de las articulaciones de los miembros inferiores

Biblioteca
UNIVERSITAS Miguel Hernández

Capítulo 1

1. INTRODUCCIÓN

El análisis de la marcha es el estudio detallado de los movimientos y patrones biomecánicos presentes durante la marcha humana. Este campo analiza la coordinación y funcionamiento de los diferentes sistemas del cuerpo, como los músculos, las articulaciones y el sistema nervioso, para entender cómo se produce y controla el movimiento de manera eficiente y efectiva.

Al examinar la marcha de un individuo, los expertos en biomecánica pueden identificar patrones anormales o desequilibrios en el movimiento. Comprender estos aspectos permite el desarrollo de dispositivos personalizados que se adaptan a las necesidades individuales, mejorando así la funcionalidad del individuo.

El ciclo de la marcha se divide en diferentes fases, incluyendo la fase de apoyo y la fase de balanceo. Cada fase involucra movimientos específicos de los miembros inferiores y superiores, y la biomecánica de la marcha se encarga de analizar estos patrones.

Este análisis es fundamental para comprender el rango de movimiento (ROM) de las articulaciones de las extremidades inferiores, lo que contribuye a prevenir lesiones o trastornos en la marcha. Además, es útil en diagnósticos y evaluaciones de progreso durante las terapias de rehabilitación. La disminución del ROM en estas articulaciones puede dificultar la realización de las actividades de la vida diaria (AVD), como caminar, subir o bajar rampa y subir escaleras.

La estimación de los ángulos de las articulaciones de las extremidades inferiores se puede lograr mediante el uso de unidades de medición inercial portátiles (IMU), las cuales se colocan en cada segmento principal de las extremidades inferiores. De esta forma, la información proporcionada por cada IMU permite conocer el valor angular de las articulaciones, siendo importante conocer las AVD que se están realizando durante la marcha para un análisis preciso de los movimientos de las extremidades inferiores.

Para analizar los ángulos de flexión/extensión de las articulaciones de las extremidades inferiores, el sistema debe detectar el inicio y el final de cada paso. La clasificación de las actividades de la marcha, mediante técnicas de aprendizaje automático, implica la

extracción de características temporales y de frecuencia de las señales de la IMU, permitiendo identificar patrones específicos de la marcha.

La clasificación precisa de las actividades de la marcha no solo tiene aplicaciones en la adaptación de la asistencia proporcionada por dispositivos como exoesqueletos o prótesis, sino que también está vinculada a la seguridad. La detección de cambios en la actividad es importante para prevenir posibles caídas o situaciones peligrosas, ya que el sistema debe anticiparse a los movimientos del usuario.

La Inteligencia Artificial, al aprender y adaptarse con el tiempo, posibilita la mejora continua de los modelos de análisis de la marcha. Además, la automatización a través de la IA permite procesar grandes cantidades de datos de manera simultánea y escalable.

Este trabajo, tiene como objetivo desarrollar un clasificador que sea capaz de distinguir entre diferentes AVD de la marcha, como andar en plano, subir rampa, bajar rampa y subir escaleras. La detección de estos cambios es útil para adaptar la asistencia de un exoesqueleto, ya que la biomecánica varía según la actividad que se está realizando. En esta tarea, se emplearán modelos de Machine Learning para clasificar las distintas AVD de la marcha utilizando la información de los ángulos de las articulaciones de las extremidades inferiores humanas. [5]

1.1. PROYECTO EXOEPI

El proyecto ExoEpi (Ankle Exoskeleton for Emergency Personnel Assistance) se trata de un proyecto desarrollado por el grupo de investigación de Robótica e Inteligencia Artificial del Instituto de Bioingeniería de la Universidad Miguel Hernández de Elche junto con las empresas INESCOP y PANTER.

El objetivo del proyecto es desarrollar un equipo de protección individual de calzado para actividades de emergencia de larga duración, integrando un exoesqueleto que ayude al usuario a desarrollar los movimientos de la marcha disminuyendo su fatiga y potenciando su rendimiento.

Este proyecto implica el desarrollo e implementación de un exoesqueleto de tobillo integrado en unas botas de intervención que ofrece una asistencia continuada durante la marcha. De esta forma, se pretende lograr una reducción del coste energético en los trabajos de extinción de incendios proporcionando la asistencia adecuada durante los múltiples desplazamientos del Personal Especialista de Extinción de Incendios Forestales (PEEIF) en el incendio y/o hasta llegar al lugar del incendio. [4]

1.1.1. DISEÑO Y COMPONENTES

En cuanto al diseño y los componentes, se ha creado un prototipo de exoesqueleto inferior de tobillo en el proyecto ExoEpi, utilizando un sistema biela-manivela para transformar el movimiento rotativo de un motor eléctrico en movimiento lineal. El diseño consta de dos partes: una unida rígidamente a la bota en la parte inferior y la otra fija a la pierna por debajo de la rodilla, conectadas mediante un punto de rotación con un eje mecanizado.

- **Motor:**
  - Motor Cube Mars AK 80-9 con un par nominal de 9 N·m y un pico de par de 18 N·m.
  - Se ubica en la posición central-superior, conectado a una estructura de aluminio que sirve como bastidor principal.
  - Un brazo de aluminio en la salida del motor articula con la biela.

- **Ortesis:**
  - Fijada en la parte superior de la estructura principal, debajo de la rodilla y encima del gemelo con una cinta de velcro.
  - Evita restricciones en los movimientos de la rodilla y el gemelo, proporcionando agarre suficientemente rígido para contrarrestar el par generado.
  - Incorpora una unidad inercial magnética (IMU) modelo XSens Dot para obtener información sobre la fase del paso.

- **Sensores y Seguridad:**
    - Dos sensores de final de carrera con roldana alrededor del motor evitan exceder límites físicos y garantizan seguridad.
    - Dos chapas de acero inoxidable en la parte inferior actúan como anclaje a la bota, con la superior proporcionando unión de rotación, ajuste de altura y anclaje a la biela. Además, alberga otra IMU XSens Dot.
    - La otra chapa inferior, también de acero inoxidable, garantiza una fijación rígida a la bota, minimizando la flexión para transmitir eficientemente el par ejercido.

- **Unión a la Bota:**
    - El conjunto inferior se fija a la bota mediante tres pernos de M5 roscados.

A partir de esta información, se podría decir que este diseño integral busca mejorar la movilidad y reducir la fatiga en el tobillo durante la marcha, ya que con este sistema está aportando un soporte activo, el cual se controla a través del motor y los sensores incorporados. [4]

Biblioteca
UNIVERSITAS Miguel Hernández

1.1.2. CONTROL DEL EXOESQUELETO

El control del exoesqueleto se realiza mediante la gestión del par del motor AK80-9. Es esencial enviar el par deseado durante la fase de apoyo del pie, evitando hacerlo durante la fase de balanceo para prevenir posibles lesiones musculoesqueléticas. El proceso de control se centrará en dos aspectos clave: la identificación del estado actual del paso y la determinación precisa del par necesario en cada instante. [4]

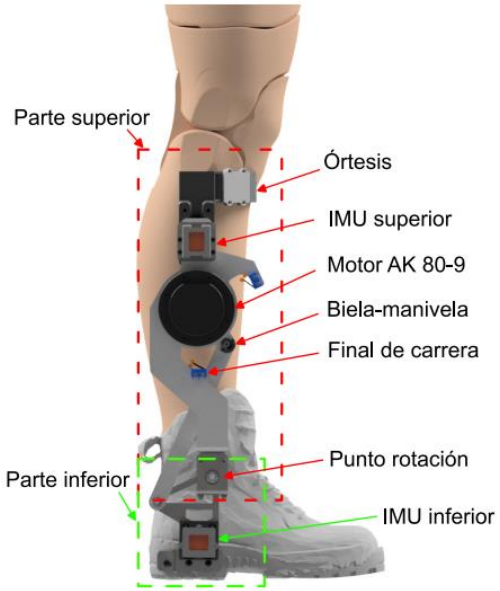

Figura 1: Prototipo exoesqueleto de tobillo ExoEpi

Biblioteca
UNIVERSITAS Miguel Hernández

1.2. OBJETIVOS

Una vez presentado la problemática y el propósito del proyecto ExoEpi, se procederá a establecer los objetivos que se cumplirán durante el desarrollo del presente Trabajo de Fin de Grado.

La finalidad de este trabajo será desarrollar un modelo que sea capaz de clasificar las distintas actividades que se realizan durante la marcha, para posteriormente adaptar la asistencia para cada uno de los usuarios.

Con el fin de alcanzar los objetivos generales del trabajo, se buscará alcanzar los siguientes puntos:

- Entrenar un modelo que sea capaz de clasificar las distintas actividades que se realizan durante la marcha
- Evaluación de la precisión del modelo mediante los conjuntos de validación y test
- Analizar los resultados de los conjuntos de entrenamiento, validación y test, mediante matrices de confusión

- Comparar la predicción de varios modelos para los conjuntos de entrenamiento y validación
- Análisis de la importancia de las características de entrada

A partir de los objetivos mencionados anteriormente, se irá realizando el desarrollo de este Trabajo de Fin de Grado, explicando en que consiste y cómo se ha ido investigando. A su vez, a través de los resultados obtenidos durante la investigación, se sacarán las soluciones, las cuales se irán detallando a lo largo de este proyecto.

Biblioteca
UNIVERSITAS Miguel Hernández

Capítulo 2

2. ANTECEDENTES

En los últimos años, los exoesqueletos de miembros inferiores han sido una innovación importante en la rehabilitación y el apoyo de las personas con problemas de movilidad. Estos dispositivos ofrecen ayuda mecánica para mejorar el movimiento y la vida cotidiana de individuos con diferentes condiciones médicas, como pueden ser: lesiones en la médula espinal o parálisis cerebral. Sin embargo, el gran desafío de este proyecto se encuentra en su diseño y uso, ya que han de asegurarse que se adapten correctamente a las necesidades específicas de cada persona.

La investigación en este campo se enfoca en hacer que estos dispositivos sean más efectivos, seguros y adaptables a las necesidades individuales de cada paciente. Esto implica entender cómo funcionan las articulaciones y cómo diseñar exoesqueletos que se ajusten de manera segura y cómoda al cuerpo humano, permitiendo una rehabilitación más efectiva y una mejora en la calidad de vida de los usuarios.

Este capítulo se enfoca en revisar los estudios que desarrollan o evalúan exoesqueletos de miembro inferior, analizando los métodos para poder adaptar la asistencia de los exoesqueletos a un miembro inferior. A lo largo del capítulo, se realizará una revisión al Estado del Arte, examinando algunos experimentos previos con el fin de aplicar sus enfoques al diseño de la experimentación presentada en este Trabajo de Fin de Grado.

2.1. EXOESQUELETOS DE MIEMBRO INFERIOR PARA REHABILITACIÓN

Las enfermedades como la lesión en la médula espinal, la parálisis cerebral o la apoplejía, pueden afectar seriamente a la movilidad humana, causando así una pérdida parcial o total

de la capacidad para caminar. Por ello, entre los diferentes dispositivos de asistencia y rehabilitación, los exoesqueletos se están destacando como una opción muy prometedora para mejorar la calidad de vida de estos pacientes y brindarles la oportunidad de volver a caminar.

Al diseñar un exoesqueleto para los miembros inferiores, es crucial considerar los grados de libertad DOF (Degrees Of Freedom) de las articulaciones, y cómo influyen en el movimiento durante la marcha. Además, la seguridad durante el proceso de rehabilitación debe ser una prioridad para evitar posibles caídas y lesiones secundarias. [2]

Por tanto, la selección del robot exoesqueleto adecuado para la rehabilitación de miembros inferiores puede variar según la capacidad de equilibrio del paciente durante el ejercicio. En la actualidad, los robots exoesqueletos de rehabilitación de miembros inferiores estándar en el mercado son:

1) **LOKOMAT:** Permite que los pacientes realicen movimientos autónomos en una cinta de correr con plataforma, utilizando un exoesqueleto para las extremidades inferiores y correas protectoras. También, puede ajustar su nivel de apoyo según la fuerza de las piernas del paciente. [3]
2) **REX:** Este robot de exoesqueleto para miembros inferiores tiene capacidad de autoequilibrio y brinda apoyo externo, para ayudar a los pacientes a ponerse de pie. Su movimiento puede ser controlado mediante un joystick diseñado para pacientes con accidente cerebrovascular, y también, se está investigando su control a través de interfaces cerebro-computadora utilizando EEG. [3]
3) **ReWalk:** Un exoesqueleto portátil para las extremidades inferiores que incorpora muletas y requiere un cierto grado de movilidad y equilibrio del paciente. Los sensores en las articulaciones de la cadera y en el exoesqueleto impulsan el movimiento a través de actuadores, estos simulan el movimiento natural de las piernas humanas para facilitar la movilidad de los pacientes con accidente cerebrovascular. [3]

Los exoesqueletos de miembros inferiores, como LOKOMAT, REX y ReWalk se muestran a continuación en la *Figura 2*, ofrecen diferentes niveles de soporte y funcionalidad para la rehabilitación de supervivientes de accidentes cerebrovasculares. [3]

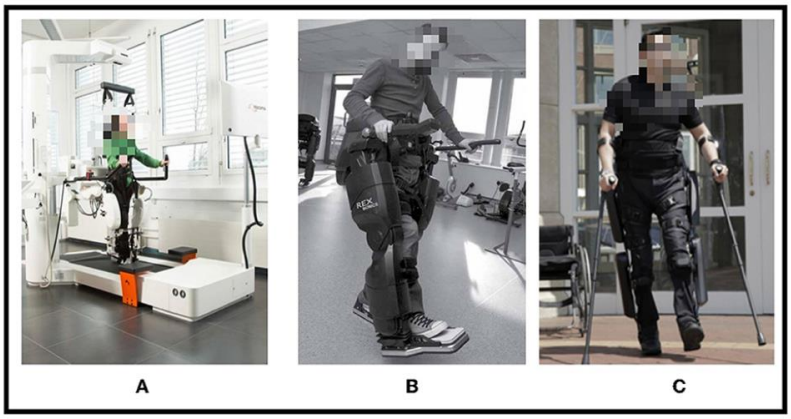

Figura 2: Tres tipos de exoesqueletos de miembros inferiores, A) Lokomat, B) Rex, C) ReWalk

En este caso, LOKOMAT se destaca por su seguridad mejorada con correas protectoras y el uso de sensores IMU, para detectar la intención de movimiento del paciente, promoviendo una participación activa en la terapia. Por lo que, se considera una terapia prometedora para mejorar la marcha y las habilidades motoras en estos pacientes. La efectividad de estos dispositivos, también dependen de la calidad de los actuadores y los algoritmos de control utilizados. [3]

La iniciativa EksoWalker surge de la necesidad de ayudar a pacientes con diversas enfermedades que afectan a su movilidad. Este método consiste en proporcionar un soporte estructural rígido de tipo ortesis, que se ajusta al cuerpo sin limitar los grados de libertad más importantes para la marcha, especialmente a aquellos que ocurren en el plano sagital (*Figura 3*). [2]

Figura 3: Grados de libertad que permite la estructura en el plano sagital en el ciclo de marcha

Para el diseño del exoesqueleto es importante estudiar el cuerpo humano, su composición y el funcionamiento mecánico de sus articulaciones, centrándose especialmente en las articulaciones involucradas en el movimiento deseado. En el caso de un exoesqueleto de

rodilla, es necesario modelar todo el miembro inferior y analizar los grados de libertad, para determinar qué movimientos son fundamentales y cuáles se pueden restringir.

Este análisis, se obtuvo utilizando un sistema de captura de movimiento BTS, y un protocolo específico para estudiar los miembros inferiores, como el protocolo de Davis-Heel (*Figura 4*). Este enfoque, permite resaltar cada elemento relevante durante el movimiento de las articulaciones, proporcionando así información crucial para el diseño y la optimización del exoesqueleto.

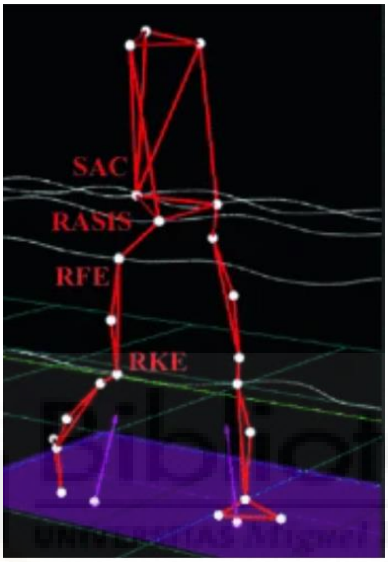

Figura 4: Sistema de captura de movimiento BTS

Las extremidades se modelan como eslabones rígidos conectados en puntos específicos llamados Centros Articulares de Rotación (CAR). Estos puntos, permiten movimientos en tres direcciones: abducción-aducción, flexión-extensión y rotación en el mismo eje. A la hora de diseñar el exoesqueleto, se utiliza esta información para asegurar que cada articulación del dispositivo se alinee correctamente con los puntos de rotación naturales del cuerpo humano. Esto garantiza que el exoesqueleto siga los movimientos del cuerpo, sin causar microdesalineamientos, lo que podría ser perjudicial para el usuario. [2]

2.2. EXOESQUELETOS DE MIEMBRO INFERIOR PARA ASISTENCIA

En cuanto a los exoesqueletos de miembro inferior para asistencia, se podría decir que este tipo de exoesqueletos están diseñados para personas que han perdido parcial o totalmente la movilidad en sus piernas, así como para aquellos con dificultades para caminar, como personas mayores. Estos dispositivos proporcionan el torque y las fuerzas necesarias para la marcha que el usuario no puede generar por sí mismo.

El propósito principal en este tipo de exoesqueletos es asistir en diversas actividades, como caminar, sentarse, ponerse de pie y subir y bajar escaleras. A continuación, se destacan algunos de los exoesqueletos más importantes para la asistencia en la locomoción humana:

- **Exoesqueleto de Extremidades Inferiores (eLegs):** Desarrollado por Berkeley Bionics, este exoesqueleto cuenta con articulaciones en el torso, cadera, rodillas y tobillos. Tanto las caderas como las rodillas utilizan actuadores para el movimiento, mientras que los tobillos emplean un sistema pasivo con resortes. Las funciones de caminar, sentarse y ponerse de pie son controladas por el usuario a través de controles ubicados en los bastones, los cuales también ayudan a reducir la carga sobre los motores y a mejorar la marcha del usuario. [1]

Figura 5: Exoskeleton Lower Extremeti Gait Sysstem (eLegs)

2.3. EXOESQUELETOS PARA EL AUMENTO DE FUERZA

Los exoesqueletos de aumento de fuerza están diseñados para llevar a cabo tareas que resultan difíciles o incluso imposibles para una persona sin asistencia. Estos exoesqueletos, proporcionan al usuario la capacidad de aplicar una fuerza considerable o resistencia adicional. Su enfoque principal se centra en aplicaciones militares, de rescate o para personal de emergencia, entre otros ejemplos.

A continuación, se presentan algunos de los exoesqueletos más destacados para el aumento de fuerza:

- **Hybrid Assistive Limb (HAL):** Desarrollado por la Universidad de Tsukuba en Japón, este exoesqueleto tiene dos versiones: una de cuerpo completo y otra de miembros inferiores. Se puede utilizar para aumentar la fuerza del usuario o como asistencia para caminar. El modelo completo pesa alrededor de 23 kilogramos, mientras que la versión de miembros inferiores pesa aproximadamente 15 kilogramos. Las articulaciones son controladas por motores eléctricos y puede levantar objetos de hasta 70 kilogramos. [1]
- **Berkeley Lower Extremity Exoskeleton (BLEEX**): Desarrollado por la Universidad de California, Estados Unidos, este exoesqueleto está diseñado para transportar cargas pesadas y es operado por soldados. Cuenta con articulaciones en la cadera, rodillas y tobillos, que son activadas mediante actuadores hidráulicos lineales y sistemas pasivos de resortes. El exoesqueleto ha alcanzado una velocidad promedio de 1.3 m/s mientras cargaba 34 kilogramos. [1]

A continuación, en la siguiente figura (*Figura 6*) se muestran los exoesqueletos "Hybrid Assistive Limb (HAL)" y "Berkeley Lower Extremity Exoskeleton (BLEEX)". [1]

Figura 6: a) Hybrid Assistive Limb (HAL), b) Berkeley Lower Extremity Exoskeleton (BLEEX)

Biblioteca
UNIVERSITAS Miguel Hernández

Capítulo 3

3. MÉTODOS

Como parte del proyecto ExoEpi, se ha desarrollado un clasificador para detectar tareas relacionadas con el movimiento de miembros inferiores. En esta investigación, se han empleado un conjunto de datos recopilados por el grupo de investigación para entrenar y validar el clasificador.

El objetivo principal es evaluar el rendimiento del clasificador en la detección de tareas específicas utilizando los datos preexistentes. A su vez, se encarga de elaborar un modelo capaz de clasificar las distintas actividades que se llevan a cabo durante la marcha. Esta clasificación, permitirá evaluar la capacidad del exoesqueleto para adaptar la asistencia a las necesidades individuales de cada usuario.

Los datos utilizados para entrenar el clasificador fueron recolectados en un estudio previo realizado por el grupo de investigación, donde se evaluó la capacidad del exoesqueleto para brindar asistencia durante la marcha.

Con el objetivo de analizar estas ventajas, se desarrollará un modelo a partir de los datos recogidos en la experimentación, el cual, pueda ayudar a predecir las actividades realizadas durante la marcha, permitiendo así, adaptar la asistencia a cada usuario de manera personalizada.

Además, en este proyecto, se ha llevado a cabo una sesión experimental de laboratorio con sujetos sanos. El objetivo de este experimento es desarrollar un sistema capaz de realizar análisis cinemáticos de los miembros inferiores, y así, distinguir de forma autónoma entre las diferentes AVD de la marcha.

En este Capítulo, se proporcionará información más detallada sobre los participantes que contribuyeron a la realización de este estudio. A su vez, se describirá el procedimiento utilizado durante la recopilación de los datos, así como el proceso completo que se ha seguido para desarrollar y entrenar el clasificador del modelo.

3.1. DESCRIPCIÓN DE LAS ACTIVIDADES

En este estudio participaron un total de 12 usuarios, de los cuales 10 son hombres y 2 son mujeres, sin deterioro motor ni cognitivo. Las alturas de los usuarios están comprendidas entre 1.65 m y 1.87 m (176.2 $\pm$ 7.4 cm). Los pesos variaron entre 56.1 kg y 90.2 kg (76.0 $\pm$ 12.5 kg), y las edades están en el intervalo entre los 23 y los 52 años (29.8 $\pm$ 7.4).

Como se puede ver a continuación en la siguiente figura (*Figura 7*), se muestran los distintos tipos de dispositivos que se han utilizado durante la sesión experimental. Estos dispositivos han sido: andar en plano, subir y bajar rampa y subir escaleras.

Figura 7: Configuración experimental y tareas realizadas [5]

En la imagen de la izquierda (*Figura 7*), se muestran las posiciones de los sensores inerciales (IMU), utilizados para calcular los ángulos de las articulaciones de las extremidades inferiores. Durante la sesión experimental, los usuarios caminaron en una cinta de correr y una máquina de escaleras para simular las diferentes actividades de la vida diaria, como andar en plano, subir y bajar rampas con una pendiente del 12 %, y subir escaleras.

Se utilizaron 4 IMU XSens Dot para registrar el movimiento de las extremidades inferiores a una frecuencia de muestreo de 60 Hz. Los sensores se colocaron en la pelvis, el muslo y la espinilla con bandas elásticas, mientras que el cuarto sensor se colocó en el pie con adhesivos.

Para simular y recopilar los movimientos durante varias AVD de la marcha, se empleó una cinta de correr h/p/cosmos 150/50 para simular andar en plano, subir y bajar rampa, y una máquina de escaleras para simular subir escaleras. [5]

Al inicio de la sesión, se colocaron los sensores IMU en las posiciones especificadas en la *Figura 7* y se registró la altura y peso de los usuarios. Posteriormente, se les pidió a los participantes que realizaran movimientos de calibración para estimar los ángulos de flexión/extensión, como mover la pierna de manera arbitraria durante 30 segundos y caminar en una cinta de correr a una velocidad cómoda durante 1 minuto.

Una vez realizada la calibración, los participantes realizaron cuatro AVD de la marcha sobre la cinta de correr y la máquina de escaleras:

1. **Andar en plano.** Los participantes caminaron en la cinta de correr a una velocidad constante de 4,5 km/h durante 5 min.
2. **Subir una pendiente del 12%.** Los participantes caminaron en la cinta ascendente a una velocidad constante de 2,5 km/h durante 5 min.
3. **Bajar una pendiente del 12%.** Los usuarios caminaron en la cinta de correr descendente a una velocidad constante de 2,5 km/h durante 5 min.
4. **Subir escaleras.** Los usuarios caminaron sobre la máquina de escaleras a una velocidad de 50 escaleras/minuto durante 3 minutos.

Se registraron los datos de aceleración y giroscopio de las IMUs colocadas sobre las extremidades inferiores para medir los ángulos de las articulaciones, las cuales fueron estimadas en el plano sagital.

En los siguientes gráficos, se muestran los ángulos estimados sobre la flexión/extensión de las extremidades inferiores, durante el ciclo de la marcha de los 12 usuarios utilizados para la recopilación de los datos empleados en el entrenamiento del clasificador. Las actividades realizadas durante la experimentación son andar en plano, subir rampa, bajar rampa y subir escaleras. Para cada una de estas actividades, se calcularon los ángulos de las articulaciones de la cadera, la rodilla y el tobillo. [5]

Ángulos de la Cadera en las distintas actividades

En cuanto a la primera gráfica (*Figura 8*), se puede apreciar que al comienzo del paso, el ángulo entre el muslo y la cadera disminuye (flexión). La máxima flexión se muestra como mínimo en la gráfica, esto ocurre aproximadamente a la mitad del paso. Posteriormente, el ángulo entre el muslo y la cadera aumenta (extensión). Esta máxima extensión se muestra como un máximo en la gráfica, esto ocurre cuando se está finalizando el paso. Por último, el ángulo vuelve a estabilizarse, aproximándose a los ángulos iniciales una vez se ha completado el apoyo del pie.

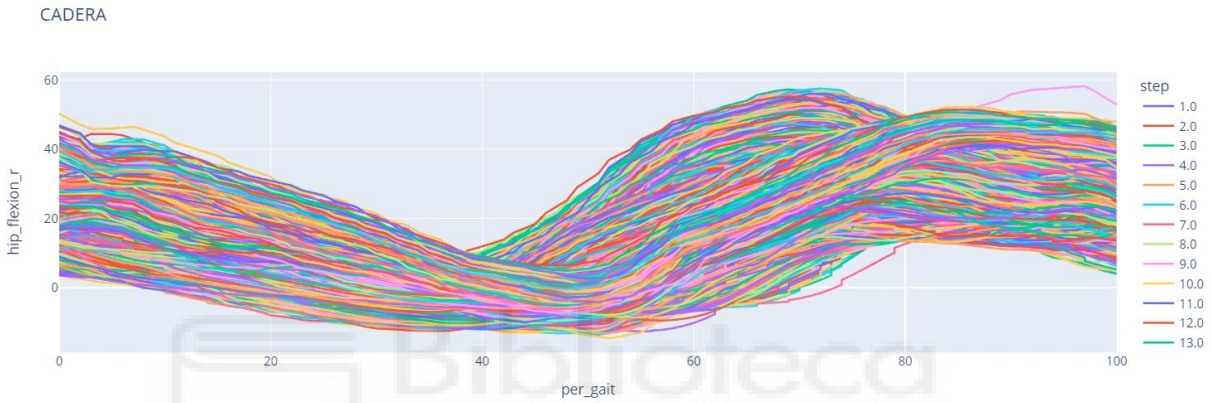

Figura 8: Andar en Plano

En esta gráfica (*Figura 9*), inicialmente la cadera se encuentra en una posición más inclinada, por lo que la flexión y la extensión de esta está más pronunciada. Por ello, la flexión máxima se produce a un mayor ángulo negativo y la extensión máxima a un mayor ángulo positivo.

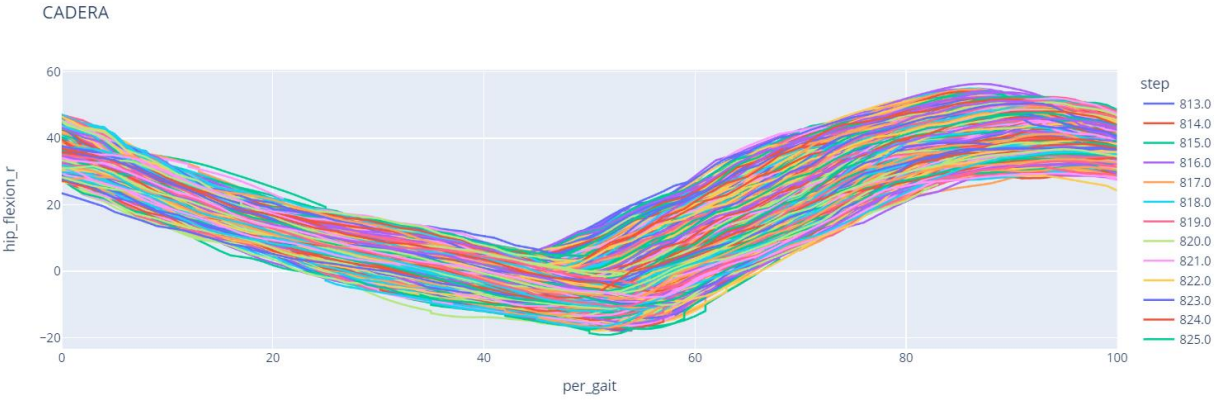

Figura 9: Subir Rampa

En esta figura (*Figura 10*), la máxima flexión se produce a un menor ángulo negativo y la máxima extensión se produce a un menor ángulo positivo que en el caso de subir rampa. En los datos mostrados se aprecia más variabilidad al bajar la cuesta que al subirla.

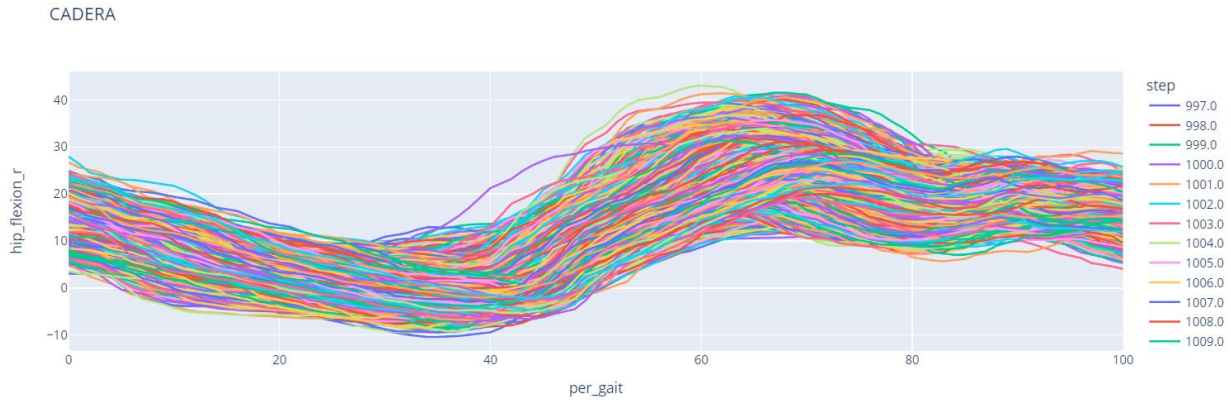

Figura 10: Bajar Rampa

En la *Figura 11*, inicialmente la cadera se encuentra más flexionada debido al posicionamiento de los pies en escalones distintos (a distinto nivel). La flexión máxima se produce en ángulos más cercanos a 0 y la extensión máxima ocurre a un ángulo mayor, debido al estiramiento pronunciado de la pierna al cambiar de escalón.

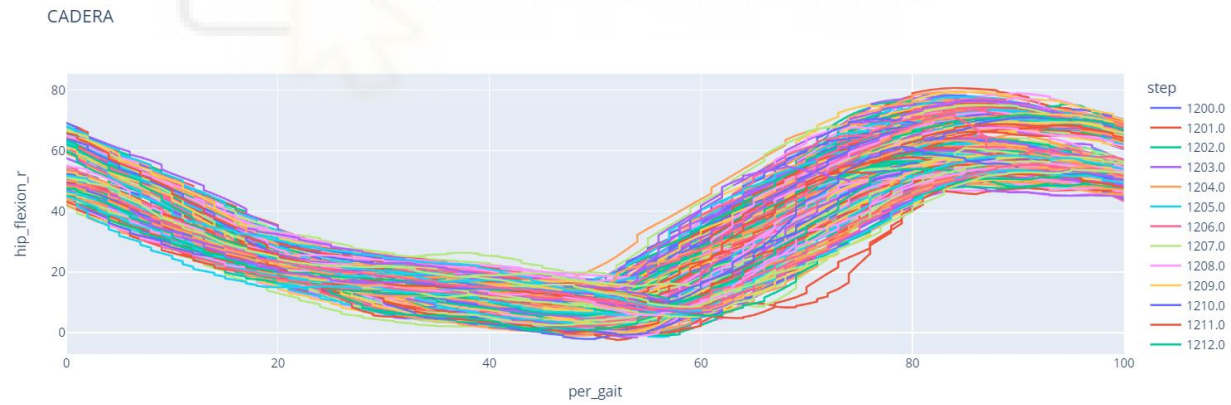

Figura 11: Subir Escaleras

Ángulos de la Rodilla en las distintas actividades

En la siguiente gráfica (*Figura 12*), se puede observar que inicialmente la rodilla se encuentra ligeramente flexionada. El mínimo representa la máxima rotación al flexionar la rodilla y el máximo representa la mínima rotación de la rodilla, es decir, la extensión.

El valor máximo de extensión se produce en valores cercanos a 0, ya que ese punto es la mayor rotación posible de la rodilla. Sin embargo, la flexión de la rodilla puede variar en cualquier ángulo.

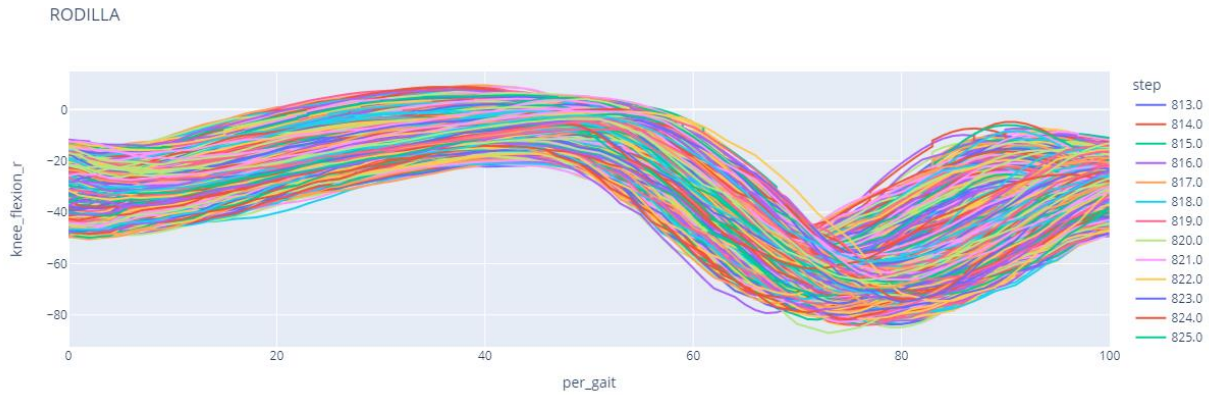

Figura 12: Andar en Plano

En la siguiente gráfica (*Figura 13*), se puede ver que la máxima extensión de la rodilla también se produce en valores cercanos a 0. Por ello, en la máxima flexión el ángulo será menor debido a la inclinación del terreno, ya que se encuentra hacia arriba.

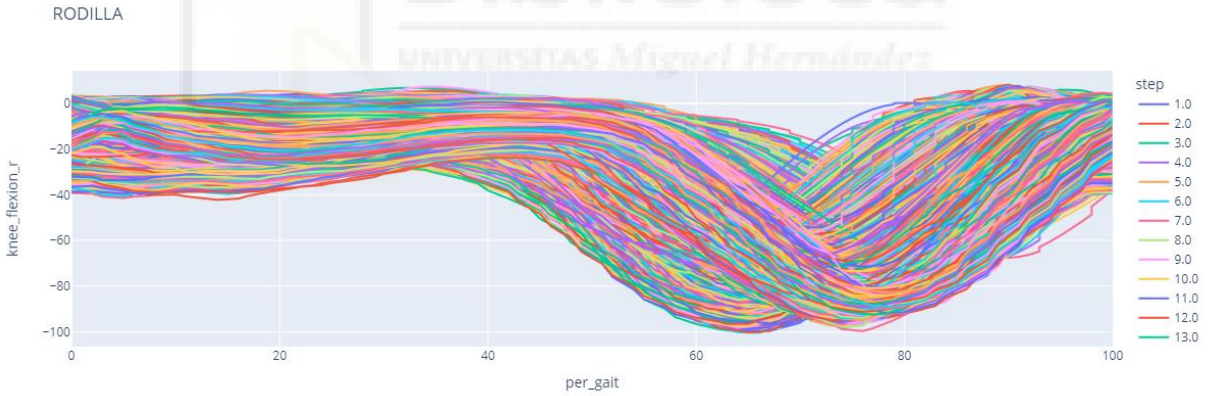

Figura 13: Subir Rampa

En la *Figura 14*, se puede decir que a partir de este gráfico se observa que la máxima extensión de la rodilla no varía de ángulo. A su vez, la máxima flexión ocurre en un ángulo mayor negativo que en la subida de la cuesta debido a la inclinación que esta produce sobre el terreno hacia abajo.

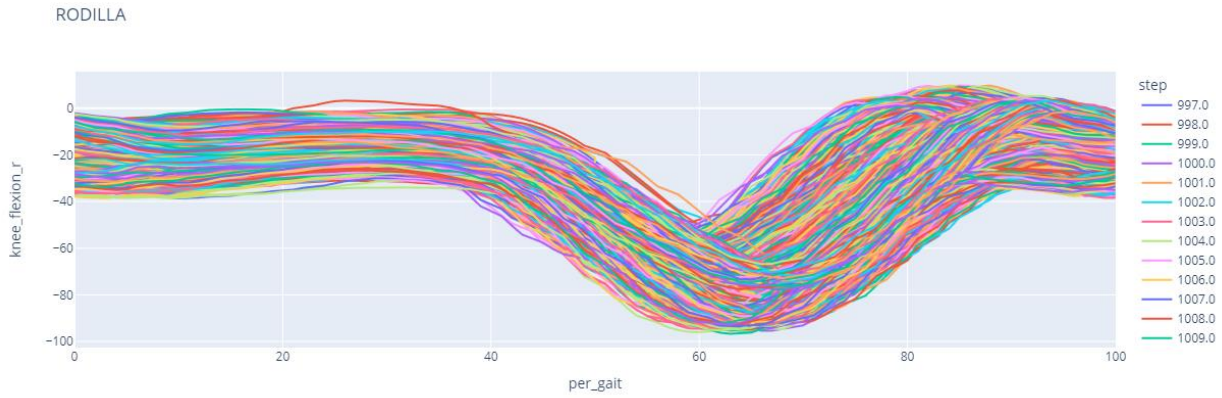

Figura 14: Bajar Rampa

Seguidamente, en esta gráfica (*Figura 15*), se puede apreciar que la máxima extensión de la rodilla no varía. Sin embargo, al ser este terreno con unos cambios bruscos de nivel, la extensión máxima de la rodilla se observa que es más pronunciada.

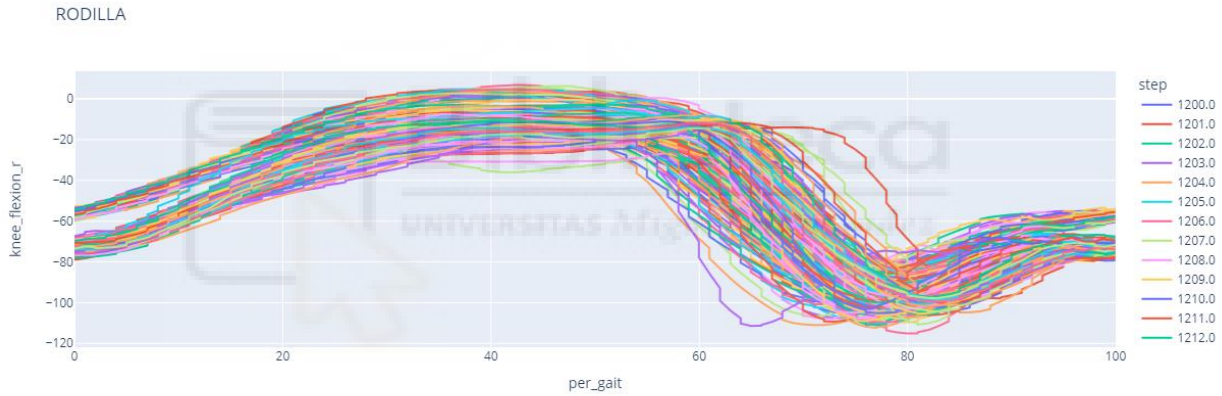

Figura 15: Subir Escaleras

Ángulos del Tobillo en las distintas actividades

A continuación, en la siguiente gráfica (*Figura 16*), inicialmente el tobillo está en su posición normal, y este va inclinándose un poco como preparación para comenzar el siguiente paso. El máximo se produce cuando está en su posición neutral y el mínimo cuando esta inclinado al máximo.

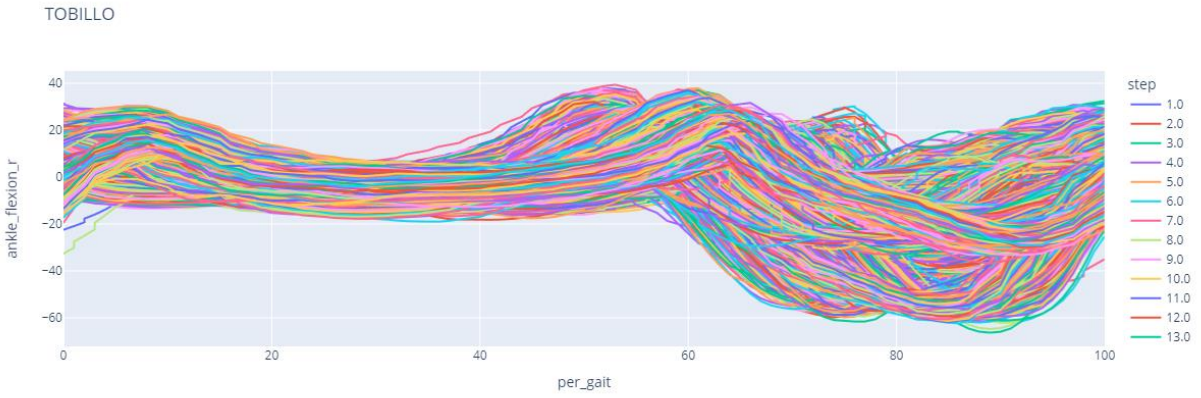

Figura 16: Andar en Plano

En el siguiente gráfico (*Figura 17*), la máxima flexión del tobillo se produce a un ángulo menor. Por ello, la primera concavidad es debida a la pequeña flexión que realiza el tobillo a comienzos del paso.

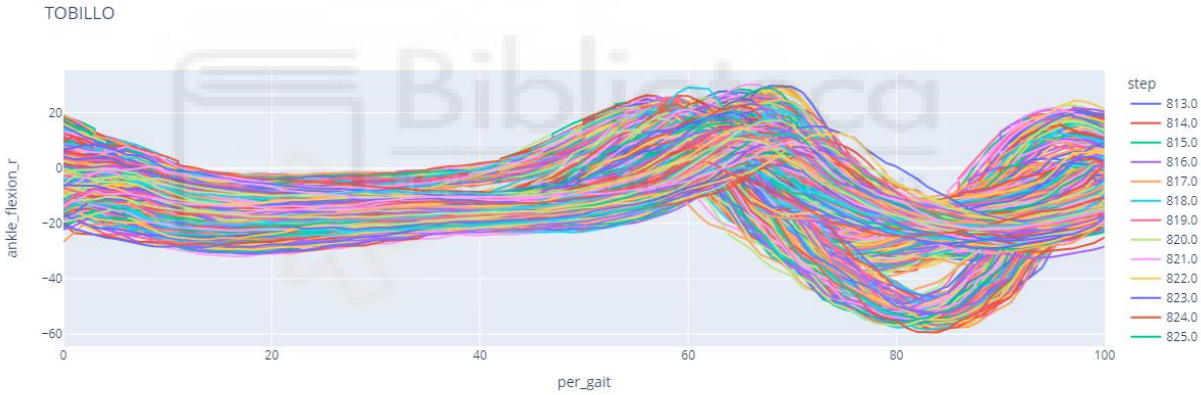

Figura 17: Subir Rampa

Como se observa en la siguiente gráfica (*Figura 18*), la mínima flexión se produce a un valor menor que en plano, por lo que no se flexiona tanto el tobillo. En este caso, el máximo de la función se produce al acabar el paso y no aproximadamente a la mitad como en el resto de las gráficas. Esto se debe a que el tobillo permanece flexionado en mayor o menor medida hasta finalizar el proceso.

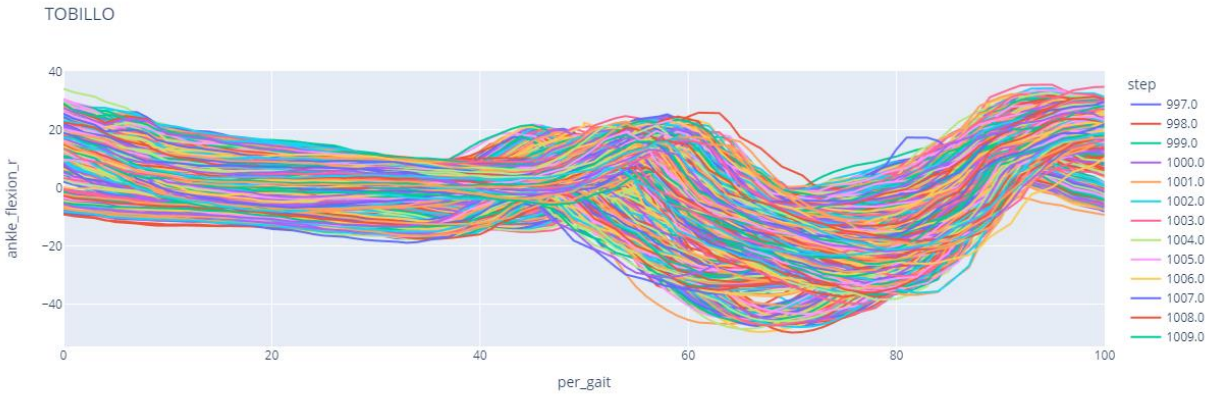

Figura 18: Bajar Rampa

Finalmente, en esta gráfica (*Figura 19*), se observa que al principio el tobillo esta ligeramente flexionado hasta la elevación del pie (produciéndose al máximo), por lo que deja de estar inclinado. El mínimo se produce cuando se está finalizando el paso, es decir, cuando se está apoyando el pie sobre la superficie y el tobillo procede a su máxima inclinación. Con estos datos se puede ver que se flexiona menos el tobillo comparado con el terreno plano, mientras que el máximo permanece estable.

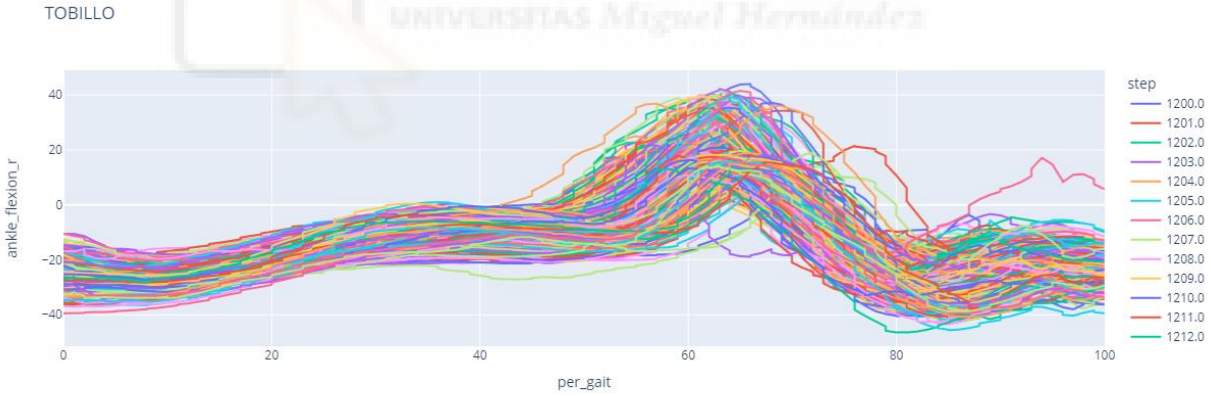

Figura 19: Subir Escaleras

3.2. PROCESAMIENTO DE DATOS

En el procesamiento de datos, es importante mencionar que las entradas del modelo corresponden a los diez descriptores por articulación (cadera, rodilla y tobillo), los cuales se utilizan para describir cada paso de los 12 usuarios.

Cada paso se ha descrito empleando diez características de dominio temporal para entrenar el clasificador. Las trayectorias angulares se han descrito en base a los descriptores con el valor mínimo y máximo y el instante durante el paso, la raíz cuadrática media (RMS), la media, la mediana, la varianza, la desviación típica y la amplitud. Mientras que las salidas del modelo que se están tratando de predecir son las actividades de andar en plano, subir rampa, bajar rampa y subir escaleras.

En primer lugar, se llevó a cabo una división de los datos en conjuntos de entrenamiento, validación y test (*Figura 20*). Este paso es importante en el proceso de construcción de un modelo de Machine Learning, ya que permite entrenar el modelo con datos conocidos, ajustar los parámetros utilizando el conjunto de validación y finalmente evaluar la eficacia del modelo en datos no vistos durante el entrenamiento, representados por el conjunto de test.

En este fragmento de código, primero se seleccionan los datos de los usuarios “u10”, “u11” y “u12” para formar el conjunto de test, mientras que el resto de los datos se utilizan para el entrenamiento y la validación. Seguidamente, se divide el conjunto de entrenamiento restante en un 75 % para entrenamiento y un 25 % para la validación, utilizando la función *train_test_split* de la librería *sklearn.model_selection*.

```
# Preprocesamiento de datos
# Importación de librerías
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, OneHotEncoder

# Crear conjuntos de entrenamiento, validación y test
users = ["u10" , "u11" , "u12"] # Usuarios del conjunto de test
```

```
training_df = descriptor_df[(descriptor_df.user != users[0]) & (descriptor_df.user != users[1]) & (descriptor_df.user != users[2])] # Conjunto de entrenamiento
test_df = descriptor_df[(descriptor_df.user == users[0]) | (descriptor_df.user == users[1]) | (descriptor_df.user == users[2])] # Conjunto de test

train_df, val_df = train_test_split (training_df, test_size=0.25, random_state=42)
# División de los datos de training_df en un 75% train y 25% val
```

Una vez realizada la división, se procede al preprocesamiento de los datos, estos se seleccionan a partir de las columnas de entrada y las etiquetas objetivo para ambos conjuntos (*train_df* y *val_df*). Estos conjuntos se preparan para ser utilizados en el entrenamiento y evaluación del modelo, esto se puede observar en el siguiente código.

```
# Crear datos de entrada y objetivos
input_cols = list(train_df.columns)[3:-1]
target_cols = 'actividades'

train_inputs = train_df[input_cols].copy()
train_targets = train_df[target_cols].copy()

val_inputs = val_df[input_cols].copy()
val_targets = val_df[target_cols].copy()
```

Biblioteca
UMH Miguel Hernández

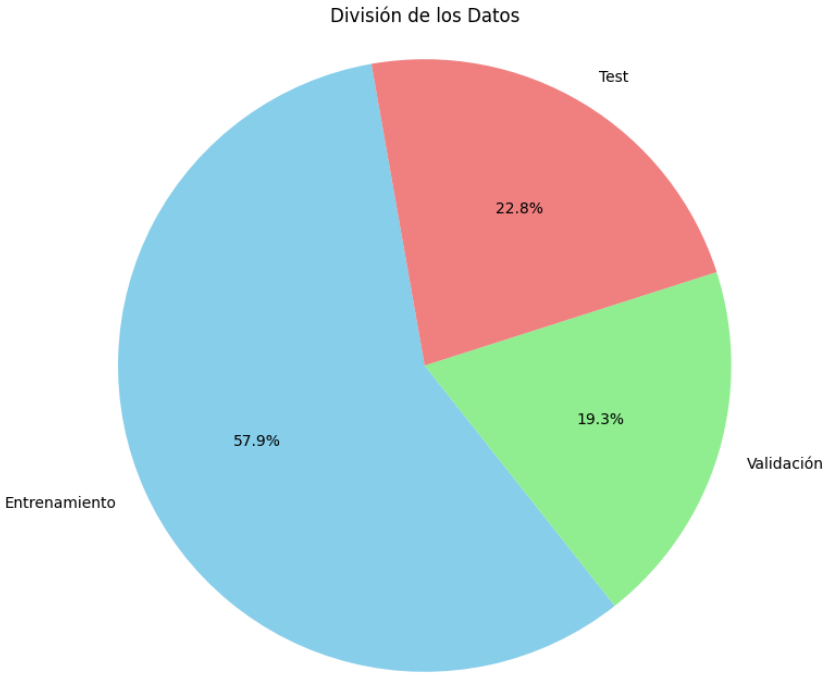

Figura 20: Representación de división de los datos en Entrenamiento, Validación y Test

En esta etapa de preprocesamiento, se aplican técnicas para escalar las características numéricas y asegurar que estén en un rango común entre 0 y 1. Esto se logra utilizando la técnica de *MinMaxScaler* de la librería *sklearn.preprocessing*. Al escalar las características numéricas, se puede evitar que aquellas con valores más grandes dominen sobre aquellas con valores más pequeños durante el entrenamiento del modelo, lo que ayuda a mejorar su desempeño y capacidad de generalización.

Como se puede ver a continuación en el siguiente código, se muestra cómo se ha llevado a cabo el proceso de escalado en los conjuntos de entrenamiento y validación para normalizar las características numéricas:

```
# Escalar características numéricas
scaler = MinMaxScaler()
train_inputs = scaler.fit_transform(train_inputs)
val_inputs = scaler.transform(val_inputs)
```

Además, se ha llevado a cabo la codificación de la columna de actividades con variables categóricas binarias utilizando *OneHotEncoder*. Está técnica permite al modelo manejar

las actividades como variables categóricas durante el entrenamiento, asegurando una representación adecuada de la información en el modelo.

Para llevar a cabo esta codificación, se utilizó *OneHotEncoder* de la biblioteca *scikit-learn*. Esta clase transforma las etiquetas categóricas en una representación binaria, donde cada categoría se convierte en una columna binaria, y solo una de ellas tendrá un valor de 1 (indicando la presencia de esa categoría) mientras que las demás serán 0.

El parámetro *sparse_output=False* se utiliza para asegurar que la salida sea una matriz densa en lugar de una matriz dispersa, lo que facilita su manejo y visualización. Por otro lado, el parámetro *handle_unknown='ignore'* se utiliza para manejar automáticamente las categorías desconocidas durante el proceso de codificación, ignorándolas en lugar de generar errores.

Después de ajustar el codificador a los datos de las etiquetas de las actividades utilizando el método *fit*, se aplicó la codificación a los conjuntos de entrenamiento y validación utilizando el método *transform*. Esto dio lugar a la creación de nuevas matrices que representan las actividades codificadas en formato binario.

Por último, se almacenaron los nombres de las características generadas por la codificación en la lista *encoded_cols* para su posterior referencia y análisis si es necesario. Este proceso garantiza que las actividades estén adecuadamente representadas en el formato requerido para el entrenamiento del modelo.

```
# Codificación One-Hot de la columna de actividades
encoder = OneHotEncoder(sparse_output=False,
handle_unknown='ignore').fit(descriptor_df[target_cols].values.reshape(-1,1))
encoded_cols = list(encoder.get_feature_names_out([target_cols]))
train_targets_encoder = encoder.transform(train_targets.values.reshape(-1,1))
val_targets_encoder = encoder.transform(val_targets.values.reshape(-1,1))
```

Finalmente, los datos fueron guardados en archivos Parquet para su posterior uso en el entrenamiento y evaluación del modelo. Estos fueron almacenados como entradas (features) y objetivos (targets), mientras que los conjuntos de test se preprocesaron de manera similar para su evaluación posterior.

Para hacer su evaluación, primero se convirtieron las matrices NumPy que contenían las características y los objetivos de los conjuntos de entrenamiento y validación en DataFrames de Pandas. Estos DataFrames se crearon utilizando las columnas de características *input_cols* y la columna de objetivos *target_cols* respectivamente.

Una vez que los conjuntos de datos estuvieron en formato DataFrame, se guardaron como archivos Parquet utilizando la función *to_parquet* de Pandas. La elección de este formato de archivo se debe a su capacidad para almacenar datos de manera eficiente y comprimida, lo que resulta en tiempos de lectura y escritura más rápidos.

```
# Convertir matrices NumPy en DataFrames de Pandas
train_inputs_df = pd.DataFrame(train_inputs, columns=input_cols)
val_inputs_df = pd.DataFrame(val_inputs, columns=input_cols)
train_targets_df = pd.DataFrame(train_targets, columns=[target_cols])
val_targets_df = pd.DataFrame(val_targets, columns=[target_cols])

# Guardar como archivos Parquet
train_inputs_df.to_parquet('train_inputs.parquet')
val_inputs_df.to_parquet('val_inputs.parquet')
train_targets_df.to_parquet('train_targets.parquet')
val_targets_df.to_parquet('val_targets.parquet')
```

A continuación, al haber procesado los conjuntos de entrenamiento y validación, se realizó el preprocesamiento del conjunto de test. En este proceso, se tomaron las características del conjunto de test y se copiaron en una nueva variable llamada *test_inputs*, mientras que una copia de los objetivos se mantuvo en *test_targets*.

Las características numéricas fueron escaladas utilizando el mismo escalador *scaler* que había sido ajustado previamente en los conjuntos de entrenamiento y validación. Esto garantizó que las características del conjunto de test estuvieran en el mismo rango que las utilizadas durante el entrenamiento y la validación del modelo.

Además, la columna de objetivos fue codificada en variables binarias utilizando el mismo codificador *encoder* que se había utilizado anteriormente con los conjuntos de

entrenamiento y validación. Esta codificación aseguró una representación uniforme de las clases de actividad durante todo el proceso.

Para simplificar el análisis, se utilizó únicamente la columna correspondiente a la clase real en *test_targets_encoder_column*. Esto se logró mediante la función *np.argmax*, que permitió seleccionar la clase más probable para cada instancia del conjunto de test.

Preprocesamiento del conjunto de test

```
# Preprocesamiento de datos de test
test_inputs = test_df[input_cols].copy()
test_targets = test_df[target_cols].copy()

# Escalar características numéricas
test_inputs = scaler.transform(test_inputs)

# Codificación One-Hot de la columna de actividades
test_targets_encoder = encoder.transform(test_targets.values.reshape(-1,1))

# Utilizar solo la columna correspondiente a la clase real
test_targets_encoder_column = np.argmax(test_targets_encoder, axis=1)
```

3.3. DESARROLLO DE CLASIFICADOR

Una vez completado el procesamiento de los datos, se procedió a la fase de desarrollo del clasificador. En esta etapa, se construyó y entrenó un modelo de clasificación utilizando un enfoque de Regresión Logística. La elección de este modelo se basó en su simplicidad, interpretabilidad y su capacidad para manejar problemas de clasificación multiclase.

La Regresión Logística es un algoritmo de aprendizaje supervisado utilizado para la clasificación. Su principal uso, es para problemas de clasificación binaria, donde el

objetivo es predecir la probabilidad de que una observación pertenezca a una de las dos clases posibles.

Utilizando la función logística, este modelo establece la relación entre una variable dependiente categórica (la variable objetivo que se quiere predecir) y una o más variables independientes (predictoras) utilizando la función logística. Esta función transforma los valores de entrada en un rango entre 0 y 1, representándolos como probabilidades de pertenencia a una de las categorías.

Durante el entrenamiento, el modelo de regresión logística estima los coeficientes de las variables independiente, lo que ayuda a maximizar la probabilidad de que los datos observados se ajusten a los datos reales. Para ello, se ajustan los parámetros del modelo para minimizar una función de pérdida que mida la discrepancia entre las probabilidades predichas y las etiquetas reales.

La Regresión Logística para la clasificación multiclase es una extensión del algoritmo de regresión logística, que se utiliza para problemas en los que hay más de dos clases posibles. Uno de los enfoques más utilizado es One-vs-Rest (OvR, uno contra el resto), que consiste en entrenar varios modelos de regresión logística binaria, uno para cada clase.

En cada modelo, una clase específica se trata como la clase positiva, mientras que el resto de clases se agrupan como la clase negativa. Después de entrenar estos modelos, se realiza una predicción para una nueva observación utilizando todos los modelos y se elige la clase con la probabilidad más alta como la predicción final.

El modelo de Regresión Logística se implementó utilizando la biblioteca *scikit-learn* en Python. Para ello, se configuraron los parámetros del modelo, como el tipo de clasificación multinomial y el solucionador de optimización adecuado para el problema en cuestión.

Posteriormente, se procedió a entrenar el modelo usando los datos de entrada del conjunto de entrenamiento y las etiquetas correspondientes. Durante este proceso, el modelo ajusta sus parámetros internos para minimizar una función de pérdida, lo que ayuda a mejorar su capacidad para predecir las etiquetas de clase correctas.

```
# Entrenamiento y evaluación de modelos
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import joblib

# Utilizar solo la columna correspondiente a la clase real
train_targets_encoder_column = np.argmax(train_targets_encoder, axis=1)
val_targets_encoder_column = np.argmax(val_targets_encoder, axis=1)

# Crear y entrenar el modelo de regresión logística
model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=1000)
model.fit(train_inputs, train_targets_encoder_column)
```

Una vez entrenado el modelo, se evaluó su rendimiento utilizando los diferentes conjuntos de datos, incluyendo el conjunto de entrenamiento, validación y test. Para comprender mejor cómo el modelo clasifica las instancias en cada clase, se utilizó la matriz de confusión, una herramienta que proporciona detalles sobre los verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. Esto puede ser especialmente útil para entender dónde está cometiendo errores el modelo.

La métrica que se ha usado para evaluar el rendimiento fue la precisión (*accuracy*), la cual indica la proporción de predicciones correctas respecto al total de predicciones realizadas.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Donde:

TP (True Positives): Número de muestras positivas que fueron correctamente clasificadas como positivas.

TN (True Negatives): Número de muestras negativas que fueron correctamente clasificadas como negativas.

FP (False Positives): Número de muestras negativas que fueron incorrectamente clasificadas como positivas.

FN (False Negatives): Número de muestras positivas que fueron incorrectamente clasificadas como negativas.

Después de configurar los parámetros del modelo, se procedió a entrenarlo y evaluar su rendimiento utilizando tanto el conjunto de entrenamiento como el conjunto de validación. Esto permitió tener una idea inicial de cómo se desempeña el modelo en datos conocidos y cómo generaliza a nuevas instancias.

Posteriormente, para tener una comprensión más detallada de cómo el modelo clasifica las instancias en cada clase, se empleó la matriz de confusión. Esta herramienta visual ayudó a identificar con mayor precisión dónde el modelo acierta y dónde comete errores en sus predicciones.

```
# Podemos visualizar el desglose de las entradas clasificadas correcta e incorrectamente utilizando la matriz de confusión
from sklearn.metrics import confusion_matrix
import seaborn as sns

# Función para predecir, calcular la precisión y mostrar la matriz de confusión
def predict_and_plot(inputs, targets, class_names, name=''):
    preds = model.predict(inputs)
    accuracy = accuracy_score(targets, preds)
    print("{} Accuracy: {:.2f}%".format(name, accuracy * 100))
    cf = confusion_matrix(targets, preds, normalize='true')
    plt.figure()
    sns.heatmap(cf, annot=True , xticklabels=class_names, yticklabels=class_names)
    plt.xlabel('Prediction')
    plt.ylabel('Target')
    plt.title('{} Confusion Matrix'.format(name));
    return preds

class_names = descriptor_df['actividades'].unique().tolist()
```

```
# Evaluación de los conjuntos de entrenamiento y validación
train_preds = predict_and_plot(train_inputs, train_targets_encoder_column, class_names, 'Entrenamiento')
val_preds = predict_and_plot(val_inputs, val_targets_encoder_column, class_names, 'Validacion')
```

Seguidamente de haber entrenado y validado el modelo, se procedió a evaluar su rendimiento final utilizando el conjunto de test, el cual consiste en datos no vistos previamente por el modelo. Esta evaluación, permite comprender cómo generaliza el modelo a nuevas instancias y proporciona una medida más precisa de su capacidad predictiva en situaciones del mundo real.

Por otro lado, para evaluar el rendimiento en el conjunto de test, también se midió el tiempo medio por paso durante la predicción. Esto proporcionó nueva información sobre la eficiencia computacional del modelo, es decir, cuánto tiempo tarda en realizar una predicción para cada instancia de entrada.

```
# Evaluación de los conjuntos de test
# Medir el tiempo de predicción en el conjunto de prueba
start_time = time.time()

test_preds = predict_and_plot(test_inputs, test_targets_encoder_column, class_names, 'Test')

total_time = time.time() - start_time

# Calcular el tiempo medio por paso
num_pasos = len(test_preds)
tiempo_medio_por_paso = total_time / num_pasos

print(f"Tiempo medio por paso: {tiempo_medio_por_paso} segundos.")
```

Finalmente, se guardó el modelo entrenado en un archivo para su posterior uso y se compararon sus resultados con los otros modelos probados. Este proceso de desarrollo

del clasificador incluyó la creación y entrenamiento del modelo de Regresión Logística, seguido por su evaluación y comparación con los otros modelos de clasificación.

```
# Guardar el modelo entrenado y cargarlo nuevamente
Actividad = {'model': model, 'scaler': scaler, 'input_cols': input_cols, 'target_cols': target_cols}
joblib.dump(Actividad, 'Actividad.joblib')
tarea = joblib.load('Actividad.joblib')
```

Además del modelo de Regresión Logística, se evaluaron varios modelos de clasificación, incluyendo el Árbol de Decisión, Random Forest y Support Vector Machine (SVM). Cada modelo se entrenó utilizando los mismos datos de entrada y se evaluó su rendimiento tanto en el conjunto de entrenamiento como en el conjunto de validación.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import pandas as pd

# Entrenar modelos
models = {
    'Regresión Logística': LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=1000),
    'Árbol de Decisión': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'SVM': SVC()
}

# Inicializar un diccionario para almacenar los resultados
results = {'Modelo': [], 'Precisión (Entrenamiento)': [], 'Precisión (Validación)': []}
```

```
for model_name, model in models.items():
    model.fit(train_inputs, train_targets_encoder_column)
    # Predicciones y evaluación en el conjunto de entrenamiento
    train_preds = model.predict(train_inputs)
    train_accuracy = accuracy_score(train_targets_encoder_column, train_preds)*100

    # Predicciones y evaluación en el conjunto de validación
    val_preds = model.predict(val_inputs)
    val_accuracy = accuracy_score(val_targets_encoder_column, val_preds)*100

    # Almacenar resultados en el diccionario
    results['Modelo'].append(model_name)
    results['Precisión (Entrenamiento)'].append(train_accuracy)
    results['Precisión (Validación)'].append(val_accuracy)

# Crear un DataFrame a partir del diccionario
results_df = pd.DataFrame(results)

# Imprimir la tabla de resultados
print(results_df)
```

Cada modelo de machine learning tiene sus propias características y formas de procesar los datos. Estas formas pueden ser más adecuadas para ciertos tipos de problemas o conjuntos de datos que para otros. Por lo tanto, el rendimiento de cada modelo puede variar dependiendo de las características específicas de los datos con los que se trabaje.

La Regresión Logística es ampliamente utilizada para los problemas de clasificación binaria y multiclase debido a su simplicidad, eficiencia computacional y capacidad para manejar la clasificación multiclase. Utiliza la función logística para modelar la probabilidad de pertenencia a cada clase y es fácil de interpretar, lo que permite comprender cómo cada característica contribuye a la predicción de la clase.

Los Árboles de Decisión dividen el espacio de características en regiones rectangulares, y asignan una etiqueta a cada región. Estos son fáciles de interpretar y visualizar, pero tienden a sobreajustarse a los datos de entrenamiento sino se controla su complejidad.

El Random Forest es una técnica de conjunto que combina múltiples árboles de decisión para mejorar la generalización y reducir el sobreajuste. Cada árbol se entrena en una muestra aleatoria del conjunto de datos y realiza predicciones. Estas predicciones se promedian para obtener la predicción final.

El Support Vector Machine (SVM) es un modelo de aprendizaje supervisado que busca encontrar el hiperplano que mejor separa las clases en el espacio de características. Por lo que puede manejar datos lineales y no lineales utilizando diferentes funciones de kernel.

3.4. EVALUACIÓN DEL RENDIMIENTO DE LOS MODELOS

En cuanto a la evaluación del rendimiento de los modelos de clasificación en el conjunto de test, se utilizaron diversas métricas que ofrecen una comprensión exhaustiva de su capacidad para realizar predicciones precisas sobre los datos no vistos. Las métricas principales que se usaron en el informe de clasificación son las siguientes:

La *precision* indica la proporción de muestras clasificadas como positivas que realmente son positivas. Esta se calcula como el cociente entre el número de verdaderos positivos y la suma de los verdaderos positivos y falsos positivos.

$$\text{Precision} = 100 \cdot \frac{TP}{TP + FP}$$

El *recall*, también conocido como sensibilidad, indica la proporción de muestras positivas que fueron correctamente identificadas por el modelo.

$$\text{Sensibilidad} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

El *F1-score* es la media armónica de la *precision* y el *recall*. Esta proporciona una medida de precisión general del modelo, teniendo en cuenta tanto la precision como el recall.

$$F1 = 100 \cdot \frac{2 \cdot \text{Sensibilidad} \cdot \text{Precision}}{\text{Sensibilidad} + \text{Precision}}$$

El *Support* indica el número de muestras en el conjunto de datos que pertenecen a cada clase. Este proporciona información sobre la distribución de clases en el conjunto de datos y puede ayudar a interpretar las métricas de *precision*, *recall* y *F1-score*.

En el siguiente código, se muestra cómo se realiza la evaluación del modelo en el conjunto de test y cómo se imprime el informe de clasificación utilizando la función *classification_report* de la biblioteca *scikit-learn*:

```
from sklearn import metrics

# Evaluar el modelo en el conjunto de test
test_predict = model.predict(test_inputs)

# Imprimir el informe de clasificación
# La función classification_report toma como argumentos las etiquetas reales (test_targets)
# Las etiquetas predichas (test_predict), y target_names, que es una lista de nombres de clases
print(metrics.classification_report(test_targets_encoder_column , test_predict,
target_names=class_names))
```

Finalmente, estas métricas proporcionan una comprensión completa del rendimiento del modelo en términos de su capacidad para clasificar correctamente las instancias en las diferentes clases, lo que ayuda a evaluar su eficacia y su aplicabilidad en la tarea de clasificación específica.

Capítulo 4

4. RESULTADOS

En función de los resultados, se detallan las métricas de precisión alcanzadas por cada modelo entrenado. Estos resultados muestran el rendimiento de cada modelo en términos de precisión (accuracy), tanto en el conjunto de entrenamiento como en el de validación.

	Modelo	Precisión (Entrenamiento)	Precisión (Validación)
0	Regresión Logística	98.434325	98.350663
1	Árbol de Decisión	100.000000	99.282897
2	Random Forest	100.000000	99.892435
3	SVM	99.952193	99.892435

A partir de estos datos, se puede observar que todos los modelos lograron altas precisiones tanto en el conjunto de entrenamiento como en el de validación. En particular, el Árbol de Decisión y el Random Forest alcanzaron una precisión del 100% en el conjunto de entrenamiento, mientras que la Regresión Logística y las SVM obtuvieron resultados muy cercanos al 100%.

Sin embargo, al analizar el rendimiento en el conjunto de validación, se observa que la Regresión Logística y los modelos basados en árboles (Árbol de Decisión y Random Forest) continúan mostrando una precisión muy alta, superior al 99%. Las SVM también muestran un rendimiento sólido, con una precisión del 99,89%.

A partir de estos resultados, se puede concluir que todos los modelos evaluados son capaces de capturar eficazmente los patrones en los datos y realizar predicciones precisas. Sin embargo, teniendo en cuenta la simplicidad y la interpretabilidad de la Regresión Logística, así como su rendimiento competitivo en comparación con los otros modelos, se decide seleccionarla como el clasificador final para este problema.

Los resultados del clasificador de Regresión Logística entrenado revelan un alto nivel de precisión, con un 98.43% en el conjunto de entrenamiento y un 98.35% en el conjunto de validación. Además, se proporcionan las matrices de confusión correspondientes, que

ofrecen una visión detallada del desempeño del clasificador al predecir las clases en ambos conjuntos de datos.

En las matrices de confusión, las filas representan los datos de la tarea realizada y cada columna representa los datos de la tarea predicha. En el entrenamiento (*Figura 21*) se muestra que las actividades de bajar rampa y escaleras están correctamente clasificadas, aunque la de subir escaleras la confunde con un 0.41% con andar en plano. Sin embargo, la actividad de andar en plano se clasifica incorrectamente con un 0.52% de las veces como subir rampa. Además, la actividad de subir rampa la confunde con un 7.5% como andar en plano.

Entrenamiento Accuracy: 98.43%

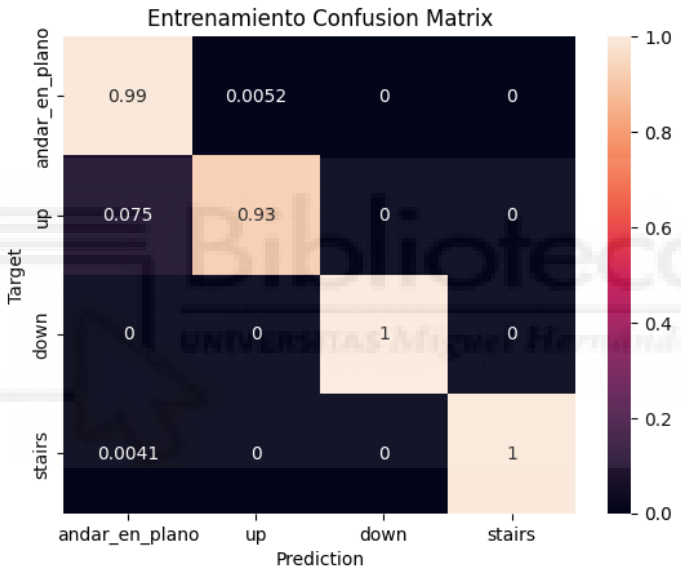

Figura 21: Matriz de Confusión del Conjunto de Entrenamiento

Para el conjunto de validación (*Figura 22*), se clasifican correctamente las actividades de bajar rampa y subir escaleras. Mientras que la actividad de andar en plano, la confunde con un 0.6% de las veces como subir rampa y con un 0.054% como subir escaleras. Al mismo tiempo, la actividad de subir rampa la clasifica incorrectamente con un 7.6% de las veces como andar en plano.

Validación Accuracy: 98.35%

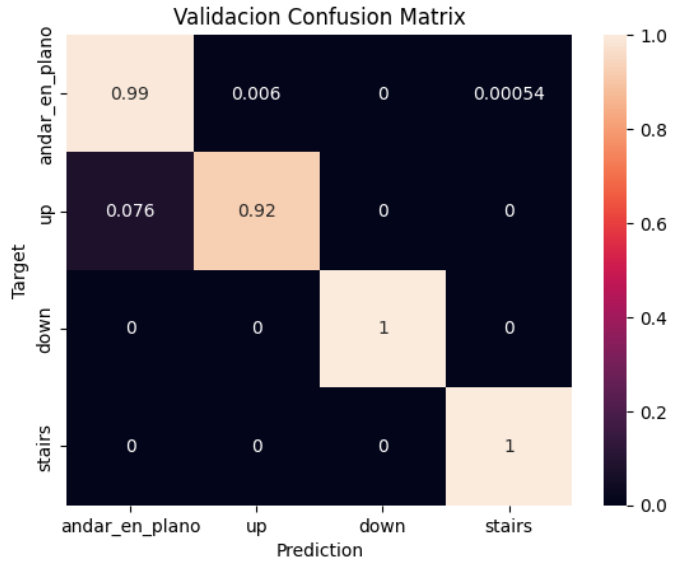

Figura 22: Matriz de Confusión del Conjunto de Validación

En el conjunto de test (*Figura 23*), se obtiene un accuracy de 93.73% con un tiempo medio de predicción por paso de 2.56e-05 segundos. La matriz de confusión muestra que la actividad de bajar rampa se clasifica correctamente. Mientras que la actividad de subir escaleras se clasifica correctamente con un 96% de las veces, confundiéndose con un 3.6% como andar en plano. Andar en plano la clasifica correctamente con un 99% de las veces, cometiendo errores de predicción del 0.37% como subir rampa y con un 0.24% como bajar rampa. Por último, el subir rampa se clasifica correctamente con un 54% de las veces, pero con un fallo del 46% de las veces como andar en plano.

Test Accuracy: 93.73%

Tiempo medio por paso: 2.560195437785607e-05 segundos.

Figura 23: Matriz de Confusión del Conjunto de Test

En estos resultados, se pueden observar las muestras de las métricas de evaluación de un modelo de clasificación para cuatro clases diferentes: andar en plano, subir rampa, bajar rampa y subir escaleras.

Por otro lado, también muestran que el modelo tiene un rendimiento bastante alto en cuanto a la clasificación de andar en plano, con una precision, recall y F1-score superiores al 90%.

Sin embargo, el rendimiento es menor en la clase de subir rampa, donde el recall es significativamente más bajo, lo que indica que el modelo tiene más dificultades para identificar correctamente todas las instancias de esas clases.

	precision	recall	f1-score	support
andar_en_plano	0.93	0.99	0.96	2456
up	0.96	0.54	0.69	391
down	0.92	1.00	0.96	71
stairs	1.00	0.96	0.98	385
accuracy			0.94	3303
macro avg	0.95	0.88	0.90	3303
weighted avg	0.94	0.94	0.93	3303

Después de entrenar el modelo de Regresión Logística, se procedió a calcular los coeficientes de importancia de las características (*Figura 24*). Estos coeficientes indican la importancia que tiene cada una de las característica en la capacidad del modelo para hacer predicciones precisas.

En primer lugar, se ajustaron los parámetros del modelo utilizando los datos de entrenamiento. Posteriormente, se obtuvieron los coeficientes de importancia mediante el atributo *coef* del modelo entrenado. Estos coeficientes indican cómo cada característica afecta a la predicción de las etiquetas de clase.

Seguidamente, para facilitar la interpretación y comparación de la importancia de las características, se normalizaron los coeficientes para que estén en la misma escala. Esto se logró dividiendo el valor absoluto de cada coeficiente por la suma total de los valores absolutos.

```
# Después de entrenar el modelo de regresión logística
model.fit(train_inputs, train_targets_encoder_column)

# Obtener los coeficientes (importancia) de las características
feature_importance = model.coef_[0]

# Tomar el valor absoluto de los coeficientes
abs_feature_importance = np.abs(feature_importance)

# Normalizar los coeficientes
normalized_feature_importance = abs_feature_importance /
np.sum(abs_feature_importance)
```

A continuación, se organizó la importancia de las características en un DataFrame de Panadas, donde cada fila representa una característica y su respectiva importancia normalizada. Este DataFrame se ordenó en sentido descendente según la importancia de las características.

```
# Crear un DataFrame para organizar las características y su importancia normalizada
importance_df = pd.DataFrame({'Feature': input_cols, 'Importance':
normalized_feature_importance})

# Ordenar el DataFrame por importancia en orden descendente
importance_df = importance_df.sort_values(by='Importance', ascending=False)
```

Al observar los resultados, se puede ver que las características “mediana_tobillo” y “mediana_rodilla” son las más importantes para el modelo, seguidas por “min_cadera” y “RMS_tobillo”. Esto significa que estas características tienen un mayor impacto en la capacidad del modelo para realizar predicciones precisas.

Por otro lado, las características como “media_tobillo”, “RMS_cadera”, “instante_max_rodilla”, “instante_max_tobillo” e “instante_max_cadera” tienen una importancia relativamente baja, lo que sugiere que su contribución a las predicciones del modelo es menor.

Finalmente, se visualizó la importancia de las características utilizando un gráfico de barras, lo que proporciona una representación intuitiva de qué características son más relevantes para el modelo. Esto permite identificar fácilmente las características más importantes en el proceso de la toma de decisiones del modelo de Regresión Logística.

```
# Imprimir la importancia normalizada de cada característica
for index, row in importance_df.iterrows():
    print(f"{row['Feature']}: {row['Importance']}")

# Crear un gráfico de barras para visualizar la importancia de las características
plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=importance_df, palette='viridis')
plt.title('Importancia de las Características')
plt.show()
```

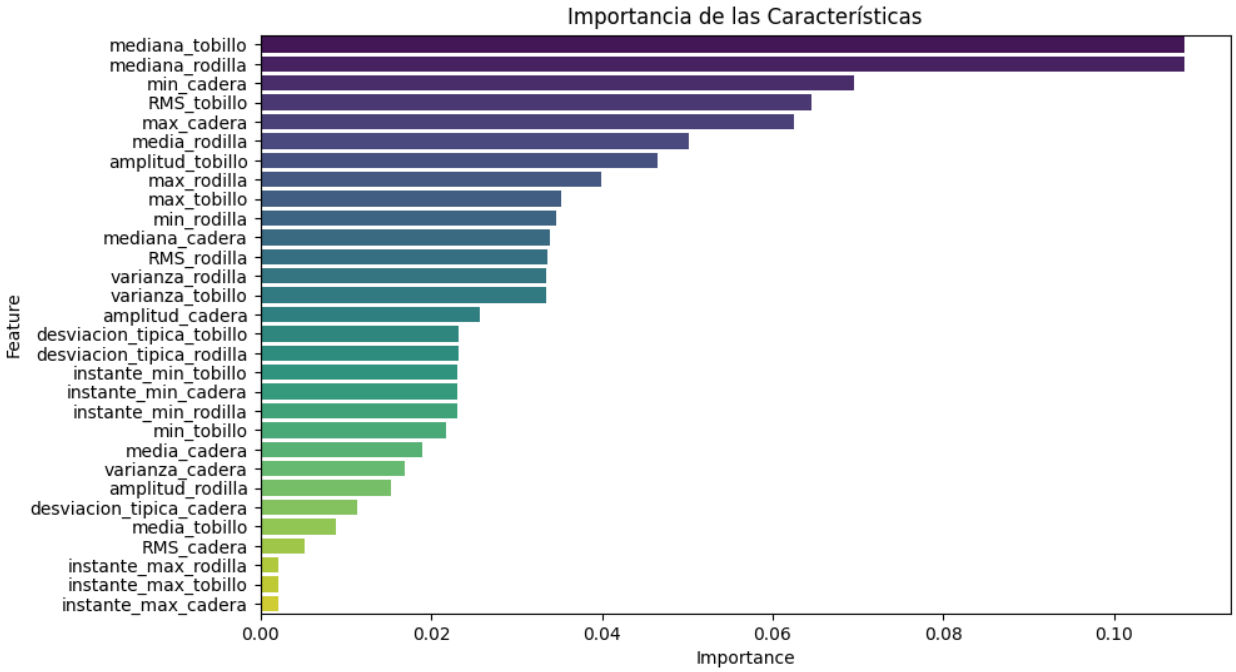

Figura 24: Importancia de las Características de Entrada

Biblioteca
UNIVERSITAS Miguel Hernández

Capítulo 5

5. CONCLUSIÓN Y TRABAJOS FUTUROS

5.1. CONCLUSIÓN

En este trabajo, se ha entrenado un modelo de Regresión Logística para clasificar las distintas tareas de AVD de la marcha: andar en plano, subir rampa, bajar rampa y subir escaleras.

Al analizar los resultado obtenidos de la importancia de las características de entrada del modelo, se ha observado que algunas variables son más significativas que otras. Esto sugiere que se podría intentar reducir la cantidad de características que entran al modelo sin perder mucha información.

Al eliminar aquellas características menos relevantes, se puede simplificar el modelo, haciéndolo más eficiente y rápido de entrenar. Además, una menor cantidad de características puede facilitar la interpretación del modelo, lo que permitiría comprender mejor que aspectos del conjunto de datos están influyendo más en las predicciones.

Sin embargo, es posible que algunas características, aunque tengan una importancia relativamente baja individualmente, podrían contribuir de manera significativa cuando se combinan con otras. Por lo tanto, cualquier reducción de características debe llevarse a cabo con un análisis detallado y validación para asegurar que no se pierda información relevante que afecte a la precisión del modelo.

Al disminuir el número de entradas al modelo, se podría también reducir el número de IMUs empleadas en el ExoEpi. Si se opta por no medir el ángulo de la cadera o la rodilla, se podría perder información relevante que podría ser importante para clasificar con precisión las tareas. Eliminar la medición de ciertos ángulos articulares podría afectar a la capacidad del modelo para entender la complejidad de los movimientos y, por tanto, reducir la precisión de la clasificación.

Por ejemplo, el ángulo de la cadera podría ser relevante para distinguir entre tareas que implican movimientos amplios y dinámicos. La eliminación de esta medida podría hacer

que el modelo tenga dificultades para diferenciar entre actividades como andar en plano y subir escaleras, en el que los patrones de movimiento de la cadera son distintos.

Del mismo modo, el ángulo de la rodilla es importante para diferenciar entre posturas y acciones específicas durante la marcha. Su exclusión podría afectar a la capacidad del modelo para distinguir entre actividades que requieran diferentes niveles de flexión de la rodilla, como subir rampas y bajar rampas.

La posibilidad de clasificar las tareas solo midiendo el ángulo del tobillo, es poco probable que sea suficiente para capturar toda la variedad de movimientos realizados durante las tareas. Aunque el ángulo del tobillo puede proporcionar cierta información sobre la posición y el movimiento del pie, es probable que no sea lo suficientemente completo para distinguir entre las distintas tareas con precisión, especialmente aquellas que involucran cambios significativos en la postura y la biomecánica de las extremidades inferiores.

Por tanto, la medición de los ángulos de la cadera, la rodilla y el tobillo es importante para mantener un alto nivel de precisión en la clasificación de las tareas relacionadas con la marcha. La exclusión de alguna de estas medidas podría afectar negativamente a la capacidad del modelo para realizar una clasificación precisa y detallada. Por consiguiente, es importante considerar cuidadosamente que información es esencial para el modelo y cómo afectaría cualquier simplificación al rendimiento general del modelo.

5.2. TRABAJOS FUTUROS DE DESARROLLO E INVESTIGACIÓN

En cuanto al Trabajo de Fin de Grado desarrollado, se podrían plantear posibles trabajos futuros para detectar tareas relacionadas con el movimiento de miembros inferiores. Este trabajo se centra en desarrollar y evaluar modelos de clasificación para identificar diversas actividades durante la marcha.

Para continuar este trabajo, se podrían explorar distintas direcciones de investigación. Una posible línea de investigación futura podría centrarse en la mejora de la precisión del

clasificador mediante el uso de técnicas de aprendizaje profundo, como podrían ser las redes neuronales convolucionales (CNN) o las redes neuronales recurrentes (RNN). Asimismo, podrían capturar de manera más efectiva las características complejas y temporales de los datos cinemáticos.

Además, podría ser beneficioso realizar pruebas adicionales en un entorno más diverso y con una muestra más amplía de participantes, incluidos aquellos con diversos niveles de movilidad o con condiciones médicas que afecten a la marcha. Esto permitiría una evaluación más completa del rendimiento del modelo en una variedad de situaciones y contextos clínicos.

Por otro lado, se podría considerar ampliar el número de situaciones detectadas, como quedarse quieto, bajar escaleras, correr, levantarse o sentarse de una silla. Posteriormente, se examinaría el rendimiento del clasificador en un entorno real. Esto daría una idea más completa de cómo funciona el modelo en diferentes situaciones, lo que ayudaría a entender mejor cómo se puede usar en la práctica.

Biblioteca
UNIVERSITAS Miguel Hernández

6. BIBLIOGRAFÍA

[1] **Cabrera, B. N.** (2017). Diseño y construcción de una pierna exoesquelética para la asistencia de la marcha. (U. d. Chile, Ed.) 139. Obtenido de https://repositorio.uchile.cl/bitstream/handle/2250/144525/Dise%C3%B1o-y-construcci%C3%B3n-de-una-pierna-exoesquel%C3%A9tica-para-la-asistencia-de-la-marcha.pdf?sequence=1

[2] **Casas, D. F., Blanco, A. C., Múnera, M., Montoya, M., Sierra, A., Rodriguez, L., & Cifuentes, C. A.** (2017). Diseño bioinspirado de exoesqueleto de miembro inferior para rehabilitación. *Encuentro Internacional De Educación En Ingeniería*, 10. doi:https://doi.org/10.26507/ponencia.522

[3] **D, S., Z, H., J, W., P, S., & Z, L.** (2023). Review of adaptive control for stroke lower limb exoskeleton rehabilitation robot based on motion intention recognition. (P. U. Dongming Gan, Ed.) *17*, 21. doi:doi: 10.3389/fnbot.2023.1186175

[4] **Herraiz-Sala, M., Fernández-Irles, C. M.-P., Blanco-Ivorra, A., Aran-Ais, F., & García-Aracil, N.** (2023). ExoEpi: exoesqueleto de tobillo para asistencia de personal de emergencias. *XLIV Jornadas de Automática 2023*, 5. doi:https://doi.org/

[5] **Martínez-Pascual, D., Catalán, J. M., García-Pérez, J. V., Sanchís, M., Arán-Ais, F., & García-Aracil, N.** (2023). Activity Classification with Inertial Sensors to Perform Gait Analysis. (L. N. Systems, Ed.) *Distributed Computing and Artificial Intelligence, 20th International Conference, 740*, 74-82. doi:https://doi.org/10.1007/978-3-031-38333-5_8