UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

ESCUELA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:

"RENDIMIENTO DE ALGORITMOS DE CLASIFICACIÓN EN EL RECONOCIMIENTO DE ACTIVIDADES DE PERSONAS"

AUTOR: RICHARD XAVIER POTOSI POTOSI

DIRECTOR: CARLOS XAVIER ROSERO CHANDI

IBARRA-ECUADOR
ABRIL 2019

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

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

1. IDENTIFICACIÓN DE LA OBRA

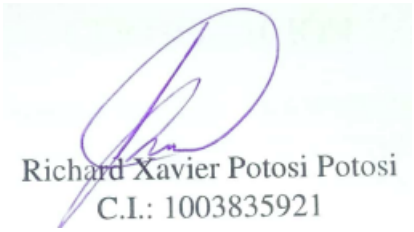
En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DEL AUTOR			
CÉDULA DE IDENTIDAD	1003835921		
APELLIDOS Y NOMBRES	POTOSI POTOSI RICHARD XAVIER		
DIRECCIÓN	NATABUELA		
EMAIL	rxpotosip@utn.edu.ec - richardpotosi6@gmail.com		
TELÉFONO FIJO	06 – 2535 – 181	**TELÉFONO MÓVIL**	0969968620
DATOS DE LA OBRA			
TÍTULO	“RENDIMIENTO DE ALGORITMOS DE CLASIFICACIÓN EN EL RECONOCIMIENTO DE ACTIVIDADES DE PERSONAS”		
AUTOR	RICHARD XAVIER POTOSI POTOSI		
FECHA	17-04-2019		
SÓLO PARA TRABAJOS DE GRADO			
PROGRAMA	PREGRADO		
TÍTULO POR EL QUE OPTA	INGENIERO EN MECATRÓNICA		
DIRECTOR	CARLOS XAVIER ROSERO C.		

2. CONSTANCIA

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

Ibarra, a los 17 días del mes de Abril de 2019



UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN

En calidad de director del trabajo de grado “RENDIMIENTO DE ALGORITMOS DE CLASIFICACIÓN EN EL RECONOCIMIENTO DE ACTIVIDADES DE PERSONAS”, presentado por el egresado RICHARD XAVIER POTOSI POTOSI, para optar por el título de Ingeniero en Mecatrónica, certifico que el mencionado proyecto fue realizado bajo mi dirección.

Ibarra, Abril de 2019

UNIVERSIDAD TÉCNICA DEL NORTE

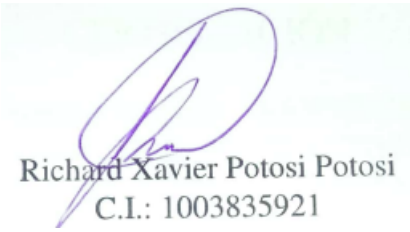
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DECLARACIÓN

Yo, Richard Xavier Potosi Potosi con cédula de identidad Nro. 1003835921, declaro bajo juramento que el trabajo aquí escrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Técnica del Norte - Ibarra, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

Ibarra, Abril de 2019



Agradecimiento

A mi madre por confiar y creer en mí, gracias por estar siempre pendiente y ser parte fundamental en mi vida.

A la Universidad Técnica del Norte, por ser la institución que me acogió y me brindó la oportunidad de formarme como un nuevo profesional. A los docentes de la carrera de ingeniería en mecatrónica, por compartir sus conocimientos, tiempo y experiencias que contribuyeron en mi formación profesional.

A todos mis compañeros por compartir momentos inolvidables y su apoyo incondicional a lo largo de esta etapa de mi vida.

Dedicatoria

El presente trabajo dedico principalmente a mi madre, por sus consejos, paciencia y por la motivación constante que me han permitido llegar a cumplir con una meta más en mi vida. Por su sacrificio, amor y trabajo que ha sido una fuente de inspiración para cumplir con responsabilidad con todos los retos o desafíos que la carrera demanda.

A todos mis familiares por su apoyo incondicional, en especial a mis tíos quienes siempre estuvieron presentes en los momentos más difíciles con sus palabras de aliento para guiarme y mostrarme el camino correcto. Por dedicarme su tiempo y ser ejemplo en mi formación personal.

Richard P.

Resumen

El reconocimiento del accionar de personas utilizando técnicas de aprendizaje de máquina y la gran cantidad de información proporcionada por sensores y procesadas por ordenadores, son sintetizados en modelos que permite estimar comportamientos futuros y prevenir posibles acontecimientos o acciones que afecten al ser humano. Un modelo correctamente entrenado y validado, facilmente puede sustituir al ser humano en la toma de decisiones.

Los modelos de reconocimiento surgen de la aplicación de varias metodologías de aprendizaje de máquina para el tratamiento de los datos. Parten con el análisis del problema a solucionar, la organización de la información en variables, la extracción de características que expresan la acción y el entorno, la segmentación de las características para el entrenamiento y evaluación, el entrenamiento de cada algoritmo y finalmente la evaluación.

Este trabajo se centra en la solución a un problema de reconocimiento de cuatro actividades caminar, sentarse en una cama, sentarse en una silla y acostarse. Se utiliza la base de datos de un repositorio digital, los mismos que se someten a metodologías para acondicionar los datos de tal forma que sean útiles para el entrenamiento. Los algoritmos considerados en el entrenamiento son el K-vecinos más cercanos, máquina vectorial de soporte y una red neuronal artificial.

La evaluación se basa en métricas que permiten cuantificar el rendimiento de los modelos. La matriz de confusión y sus derivaciones como: la tasa de error, especificidad y recall son los principales indicadores. Los resultados obtenidos, independientemente del modelo, son aceptables debido que sus rendimientos están sobre el 80 % de eficiencia en el reconocimiento.

Abstract

The recognition of the actions of people using machine learning techniques and large amount of information provided by sensors and processed by computers are synthesized in models that allow to estimate future behaviors and preventing possible events or actions that affect the humans. A properly trained and validated model can easily replace the humans in decision making.

The recognition models arise of the application of several machine learning methodologies for the treatment of data. They start with the analysis of the problem to solve, the organization of the information in variables, the extraction of characteristics that express the action and the environment, the segmentation of the characteristics for the training and evaluation, the training of each algorithm and finally the evaluation.

This work focuses on the solution to a recognition problem of four activities: walking, sitting on a bed, sitting on a chair and lying down. The database of a digital repository is used, the same ones that undergo methodologies to prepare the data in such a way that they are useful for training. The algorithms considered in the training are the K-nearest neighbor, support vector machine and an artificial neural network.

The evaluation is based on metrics that allow to quantify the performance of the models. The confusion matrix and its derivations such as: the error rate, specificity and recall are the main indicators. The results obtained, regardless of the model, are acceptable because their performance are above 80 % recognition efficiency.

Índice general

Índice de figuras

Índice de tablas

Listado de códigos

Capítulo 1

Introducción

1.1. Motivación

El estudio del accionar del ser humano entre sí o con entes artificiales despierta el interés de encontrar metodologías basadas en aprendizaje de máquina que tengan la capacidad de valorar el comportamiento y su posible reacción [1].

Con el avance de la tecnología es más factible dar soluciones prácticas a numerosos aspectos del mundo real. Sin duda el aprendizaje de máquina en sus diferentes variantes dentro de la tecnología informática ofrece una variedad de métodos y estrategias para dar solución a diferentes problemáticas de la vida cotidiana [2]. Los métodos se caracterizan por algoritmos que cumplen funciones propias como clasificación y regresión, principalmente en la toma de decisiones en base a información proporcionada.

Modelos basados en aprendizaje de máquina con entrenamiento óptimo fácilmente pueden reemplazar en la toma de decisiones a un ser humano. Son utilizados generalmente para brindar soluciones a problemas de predicción y clasificación, por lo tanto el presente trabajo se centra en el análisis del rendimiento de modelos en la clasificación de actividades de personas con el uso de un software matemático.

1.2. Objetivos

1.2.1. Objetivo Principal

- Desarrollar una metodología para la medición de desempeño de algoritmos de clasificación en el contexto del reconocimiento de actividades de personas.

1.2.2. Objetivos Específicos

- Analizar el contexto del problema en base a los datos de repositorios para la selección de las técnicas de aprendizaje de máquina a comparar.
- Establecer una metodología para la comparación del desempeño de las técnicas escogidas.
- Resolver el problema a través de las técnicas de aprendizaje de máquina.
- Validar los modelos de clasificación automática con nuevos conjuntos de datos.

1.3. Antecedentes

El aprendizaje de máquina o automático es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a los computadores aprender. Para modelar un problema se puede realizar mediante tres formas diferentes en función del conocimiento existente: como un problema de aprendizaje de máquina supervisado, no supervisado o semi supervisado [3].

El aprendizaje supervisado en la literatura es considerado para el desarrollo de clasificadores multiclase que identifiquen estados tales como: el modo de transporte en [4], el reconocimiento de actividades de personas en [5] entre otros. Los algoritmos más comunes que se emplean en problemas de clasificación son: K vecino más cercano (KNN, k-nearest neighbor), máquina de vectores de soporte (SVM, support vector machine) entre otros.

Los resultados en diferentes experimentos muestran a la precisión en el reconocimiento de las clases como un indicador de rendimiento de un clasificador. La precisión alta en la clasificación es sinónimo de conocer la pertenencia de cada instancia en su clase correcta, lo cual permite inferir en la toma de decisiones correctas posteriormente.

1.4. Problema

El reconocimiento de actividad tiene como objetivo interpretar las acciones de las personas a través del uso de diversas tecnologías de detección. Normalmente hace uso de sensores alrededor del sujeto para registrar sus movimientos, mientras que los sistemas expertos emplean los datos monitoreados para detectar las actividades realizadas [6].

El análisis del comportamiento humano ha demostrado ser de valor clave para comprender mejor las necesidades y demandas de las personas. Así, la asistencia médica y el bienestar son posiblemente los campos que aprovechan más activamente el conocimiento obtenido del análisis del comportamiento humano [7].

Los primeros sistemas de reconocimiento de actividad usaban principalmente la visión como la modalidad sensorial y esa línea de investigación aún continúa hoy. Sin embargo, los sistemas basados en visión sólo se pueden usar en un espacio confinado [8].

Actualmente existen varias aplicaciones exitosas de aprendizaje de máquina para el reconocimiento de actividades de personas. Uno de las factores a considerar dentro de otras aplicaciones es la mejora del rendimiento de las técnicas. En base al conocimiento del autor, no se han encontrado en la literatura metodologías de comparación de desempeño referentes a algoritmos de reconocimiento de actividades del ser humano.

1.5. Justificación

Los algoritmos utilizados por el aprendizaje de máquina tienen errores y el punto clave está en resolver o mejorar las consecuencias de estos errores [9].

La trascendencia del trabajo propuesto ayuda en una investigación como fuente de información para determinar una metodología de implementación de algoritmos para la clasificación adecuada de actividades de personas.

Los resultados obtenidos podrían ser utilizados para realizar un software que tenga la capacidad de aprender de rutinas diarias y realizar estudios del comportamiento de personas. Podría mejorar la calidad de vida y promover estilos de vida más saludables, brindar sugerencias de comportamiento al usuario y detectar comportamientos anómalos.

1.6. Alcance

El presente estudio desarrollará la síntesis de modelos de aprendizaje de máquina para la clasificación automática de actividades de personas con datos obtenidos desde un repositorio digital. Además se analizará el análisis del rendimiento de varios clasificadores. Para el efecto, se utilizará un software matemático.

Capítulo 2

Revisión Literaria

Este capítulo detalla la teoría base utilizada para la ejecución de este proyecto.

2.1. Reconocimiento de actividades de personas

La capacidad de reconocer se considera un comportamiento inteligente, por lo que hoy en día se trabaja en la creación de sistemas informáticos capaces de reconocer las acciones, metas, planes e intenciones de seres humanos o sistemas artificiales a partir de la observación de su comportamiento y del entorno [5].

Algunas teorías sobre el funcionamiento del cerebro humano vienen a confirmar que gran parte de su capacidad se centra en predecir el futuro, incluyendo el comportamiento de otras personas [10]. Los seres humanos buscan maneras de generar modelos de conducta de otros seres humanos y en relación de estos modelos, actuar en consecuencia.

En la actualidad existen sistemas con la capacidad de reconocer las actividades mediante el uso de técnicas como son el modelado de usuarios, la comprensión de lenguaje natural, los agentes inteligentes, la visión artificial, las interfaces inteligentes de usuarios y el aprendizaje de máquina [5].

El proceso de reconocimiento de actividad se considera complejo, sin embargo según [11] se puede generalizar en las tareas presentadas en la Fig. 2.1.

2.2. Enfoques de reconocimiento

El interés por esta línea de investigación es alto, existen varios trabajos relacionados en esta área en función de diferentes enfoques o criterios. A continuación se detallan algunos enfoques que se encuentran en la literatura.

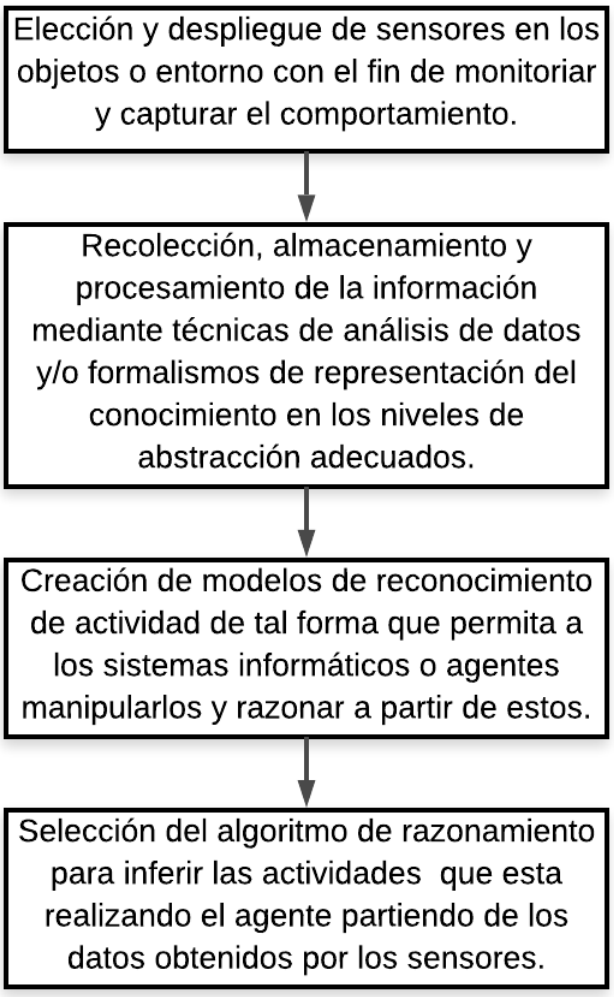


Figura 2.1: Etapas del proceso de reconocimieno

2.2.1. Sujeto de reconocimiento

Considera al actor o actores que son objeto de reconocimiento, es posible diferenciar entre reconocer el comportamiento de un robot, una persona o un agente software [5].

Existen estudios enfocados en reconocer el comportamiento de personas tales como: el comportamiento de un estudiante durante su interacción con una plataforma de aprendizaje en [12] y las actividades que realiza una persona mayor o con discapacidad en su vivienda en [13].

2.2.2. Sujeto reconocedor

Generalmente el trabajo de reconocimiento se lo realiza desde un sistema informático en un ordenador, pero tambien lo pueden llevar a cabo agentes de software autónomos como es el caso del fútbol simulado de la RoboCup [5]. Un ejemplo es el reconocimiento de actividades humanas llevado a cabo por robots de uso doméstico [14].

2.2.3. Modelo basado al origen de los datos

Trata de interpretar la forma en como se crean los modelos a partir de la información obtenida de los datos. El reconocimiento de actividades puede clasificarse como orientado a datos u orientado a conocimiento [11]. El enfoque orientado a datos, utiliza técnicas de minería de datos y aprendizaje de máquina para entrenar los modelos de actividad partiendo de grandes bases de datos preexistentes. En el enfoque orientado a un conocimiento, busca hacer uso de conocimiento del dominio y de heurísticas para realizar el modelado y el reconocimiento de patrones [5].

2.2.3.1. Reconocimiento de actividad basado en visión

Utiliza la tecnología de percepción visual para monitorear el comportamiento de la persona y los cambios que se producen en el entorno. Emplea técnicas de visión por computador para analizar los datos obtenidos por las cámaras con el fin de obtener patrones de comportamiento [5].

Generalmente el proceso de reconocimiento de actividades basado en visión cumple con la siguiente secuencia:

- Detección de objetos
- Seguimiento de comportamiento
- Reconocimiento de actividad
- Evaluación de actividad

2.2.3.2. Reconocimiento de actividad basado en sensores

Emplea redes de sensores y la identificación por radiofrecuencia (RFID) para obtener información del comportamiento de la persona. Para clarificar el uso de los dispositivos en tareas de reconocimiento se describe 2 enfoques [5].

Basado en sensores portátiles: Hace referencia a sensores portátiles que se colocan directa o indirectamente en el cuerpo de la persona, el monitoreo de la actividad se la realiza con la recolección de datos de la posición de la persona.

Basado en densidad de sensores: Los sensores se encuentran incorporados a los objetos, la tarea de reconocimiento esta centrada en la interacción entre el objeto y la persona.

2.2.3.3. Reconocimiento de actividad basado en interacción

Los datos se generan a partir de la interacción entre el usuario y el dispositivo (ordenador), generalmente este tipo está orientado a modelar el comportamiento de los usuarios de aplicaciones informáticas [5].

2.3. Aprendizaje de máquina

Es un subcampo de la inteligencia artificial que busca que mediante el uso de diferentes herramientas informáticas y estadísticas hacer que las computadoras aprendan y puedan comportarse de forma mas inteligente en la toma de decisiones. Este aprendizaje está basado en el análisis de diferentes conjuntos de datos que caracterizan el problema a resolver.

El aprendizaje de máquina según [15] puede definirse de la siguiente manera: (Un programa de computadora aprende de la experiencia E respecto a alguna clase de tarea T y una medida del desempeño D, si su desempeño en la tarea T, medido a través de D, aumenta con la experiencia E.) Es necesario identificar estos tres elementos para plantear correctamente un problema de aprendizaje: el tipo de tarea a realizar por el aprendiz (T), la medida del desempeño que se debe mejorar (D) y la fuente de experiencia (E).

Entonces, en un problema de aprendizaje máquina centrado en la identificación de actividades realizadas por el ser humano se define los elementos de la siguiente manera:

- 1. E, experiencia de realizar varias veces la misma actividad.
- 2. T, la tarea que cumple el ser humano (sentarse, caminar, etc.).
- 3. D, la probabilidad que el programa identifique la actividad realizada.

El esquema conceptual del aprendizaje de máquina se define en la Fig. 2.2, donde se visualiza la etapa del aprendizaje en un computador, a medida que crece la experiencia también existe incremento notorio en su medida de desempeño.

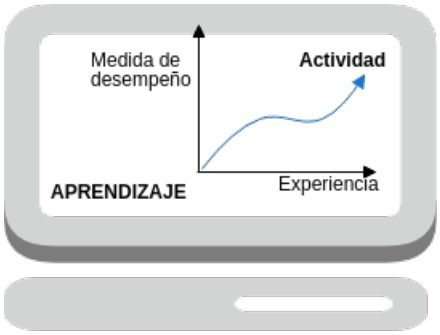


Figura 2.2: Conceptualización de aprendizaje máquina

El aprendizaje es la capacitación de un modelo con información que caracterizan el conocimiento extraído de los datos y el conocimientos de expertos en el problema. Generalmente la información para el aprendizaje se presenta en instancias que son la agrupación de varias características que representan una acción en un instante dado [16].

En la Fig. 2.3 se ilustra el proceso de aprendizaje de máquina, la etapa de aprendizaje inicia con el ingreso de los datos de entrenamiento (instancias) en el algoritmo para determinar el modelo, una vez determinado el modelo con su respectivo entrenamiento o aprendizaje se aplica un nuevo conjunto de datos para inferir un salida.

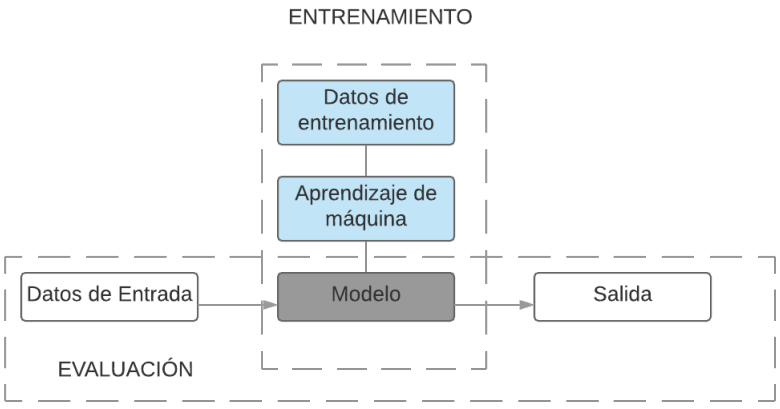


Figura 2.3: Proceso de aprendizaje

2.3.1. Tipos de aprendizaje de máquina

En la revisión de la literatura en [16] y [2] los tipos de aprendizaje dependen de la información que presenta los datos y en función del problema a tratar.

Según la información disponible en el conjunto de datos:

- Aprendizaje supervisado: Cada instancia presenta su valor característico de variables de entrada con su respectivo resultado esperado.
- Aprendizaje no supervisado: Sus instancias para entrenamiento no contienen información de algún resultado esperado. La predicción del resultado se basa en establecer relaciones entre sus datos.

Según el tipo de problema a resolver:

- Regresión: Es la obtención de un valor numérico continuo a partir de la ponderación de los datos de entrada por una serie de parámetros.
- Clasificación: Es la asignación a una categoria a cada instancia del conjunto de evaluación. Existen clasificadores binarios (verdadero o falso) o múltiples (3 o más opciones de categorias).
- Agrupamiento o clustering: Es la agrupación de los datos en diferentes grupos en base a su "similaridad". La métrica empleada para ello debe ser definida previamente. Existen reglas de asociación: la búsqueda de conjuntos de objetos que suelen aparecer juntos en los datos, a partir de ellos, se determina la probabilidad de que un objeto esté presente o no en una categoria determinada.

2.4. Aprendizaje supervisado

El aprendizaje supervisado busca que los ordenadores tengan la capacidad de aprender en una función, para que posteriormente permita predecir con la información asociada a cada clase. Este proceso parte de un conjunto de datos conformado por varias instancias, cada una de ellas formada por un par (x,y), donde $x = (x_1,x_2...,x_N) \in X$ que es el vector de N atributos o instancias y $y \in Y$ que es el conjunto de etiquetas de decisión de las instancias. Los atributos y las etiquetas de decisión son tipicamente elementos de los números reales (atributos numéricos) o bien conjuntos de nombres de un dominio específico (atributos nominales).

En la Tabla 2.1 muestra la forma en que se representa la información en el aprendizaje supervisado. Cada instancia x_i esta descrita por un vector de atributos y una etiqueta de decisión y_i [16].

Tabla 2.1: Estructura de la base de datos

Ejemplos	Atributo 1	Atributo 2	...	Atributo N	Decisión
x_1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,N}$	y_1
x_2	$x_{2,1}$	$x_{2,2}$	...	$x_{2,N}$	y_2
...	...	...	...	...	...
x_M	$x_{M,1}$	$x_{M,2}$	...	$x_{M,N}$	y_M

2.4.1. Preprocesamiento de la información

Las señales de los sensores acorde con su configuración tienden a recopilar grandes volúmenes de datos con gran cantidad de información, donde es posible extraer características dominantes para cada clase. Existe una extensa literatura dedicada a metodologías útiles para la extracción de características tales como: el método de máxima dependencia mínima redundancia mRMR [4] y el uso de series de tiempo [17].

2.4.1.1. Series de tiempo

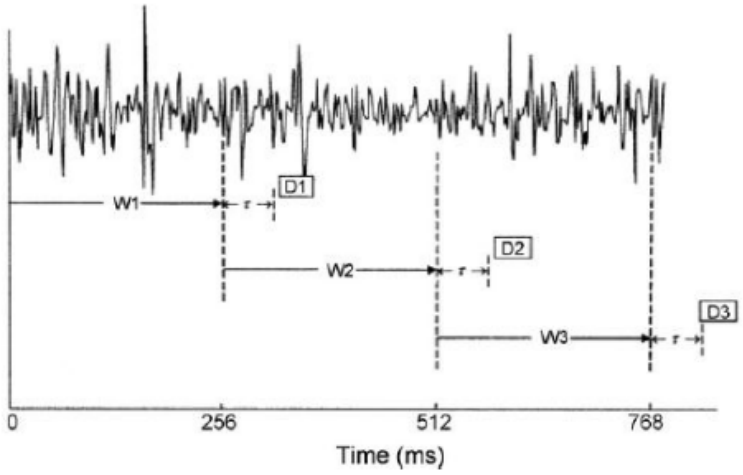
El análisis de series de tiempo procura obtener patrones de comportamiento de un conjunto de datos recolectados en un período de tiempo (ventanas de tiempo). Existen técnicas de segmentación de los datos que generan ventanas adyacentes o ventanas superpuestas como lo muestra la Fig. 2.4. W y R representa el tamaño de la ventana de tiempo y τ es el tiempo requerido para el cálculo de las características [18].

El tamaño de las ventanas Tw se determina de acuerdo con la siguiente relación $Tw \geq 1/f$ propuesta en [19], donde f es la frecuencia de muestreo. Según con la aplicación de interés se usan diferentes Tw. La reducción del tamaño de la ventana se refleja en un reconocimiento más rápido a expensas de utilizar menos datos para el cálculo de características [7]. Intuitivamente, cuanto mayor es el tamaño de la ventana de tiempo, aumenta la precisión, debido a que con tamaños de ventana más grandes hay más información disponible. Mientras que ventanas de tiempo pequeñas, se tiene menos información y su tiempo de procesamiento es más rápido por lo que son utilizadas para aplicaciones donde se desea proveer información en tiempo real con limitaciones en su precisión.

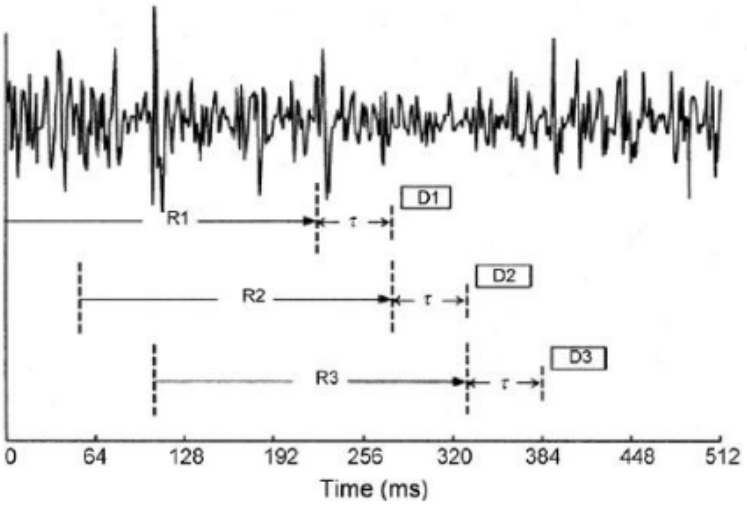
2.4.1.2. Métodos de división

Hold Out

La validación Hold Out es un método para la división del conjunto de datos en porciones para entrenamiento y prueba evitando la superposición entre ellos, proporciona una estimación más precisa del rendimiento generalizado del algoritmo. Evita el ajuste excesivo debido a la



(a) V. adyacente



(b) V. superpuesta

Figura 2.4: Ventanas de tiempo. a) ventana adyacente. b) ventana superpuesta. [18]

división aleatoria de los datos y que el conjunto de prueba no intervenga por ningún motivo en el proceso de entrenamiento [20].

En la Fig. 2.5 muestra la división del conjunto datos y su acción en el proceso de aprendizaje. En [21], considera que en muchas ocasiones se necesita de un tercer conjunto (validación) para ajustar determinados aspectos del modelo para que finalmente aprenda y que una vez se ajuste el modelo, haya una evaluación final. Completamente independiente del proceso se cree conveniente que se dividan el conjunto de datos en tres grupos: entrenamiento, validación y prueba, los porcentajes habituales en la literatura son: (50:20:30) y (40:20:40) respectivamente.

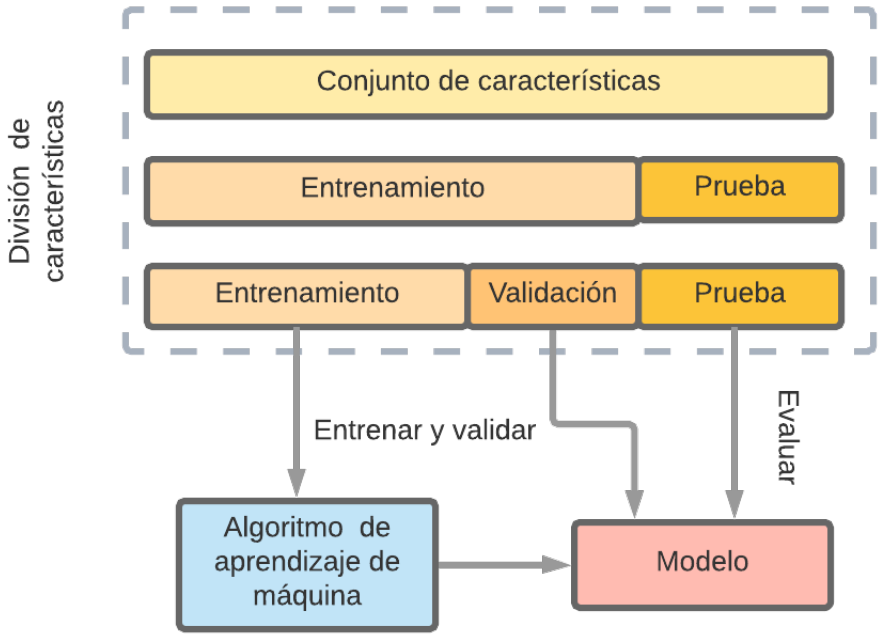


Figura 2.5: Segmentación de los datos por hold out

Validación cruzada

En la validación cruzada de k-fold, los datos se dividen en k segmentos de igual tamaño (N/k) donde N es la cantidad de datos y k es el número de grupos a formar. El valor común de k en la literatura de problemas de aprendizaje de máquina varía alrededor de 10, que a su vez depende del total de datos disponible [20].

El uso de los k segmentos en el modelo se detalla en la Fig. 2.6, uno de los k segmentos se usa para pruebas, y los restantes se usan para entrenamiento. Este enfoque se repite seleccio-

nando cada uno de los k segmentos diferentes como un conjunto de prueba. La precisión total del modelo con validación cruzada es el promedio de la precisión evaluada con los diferentes conjuntos [20].

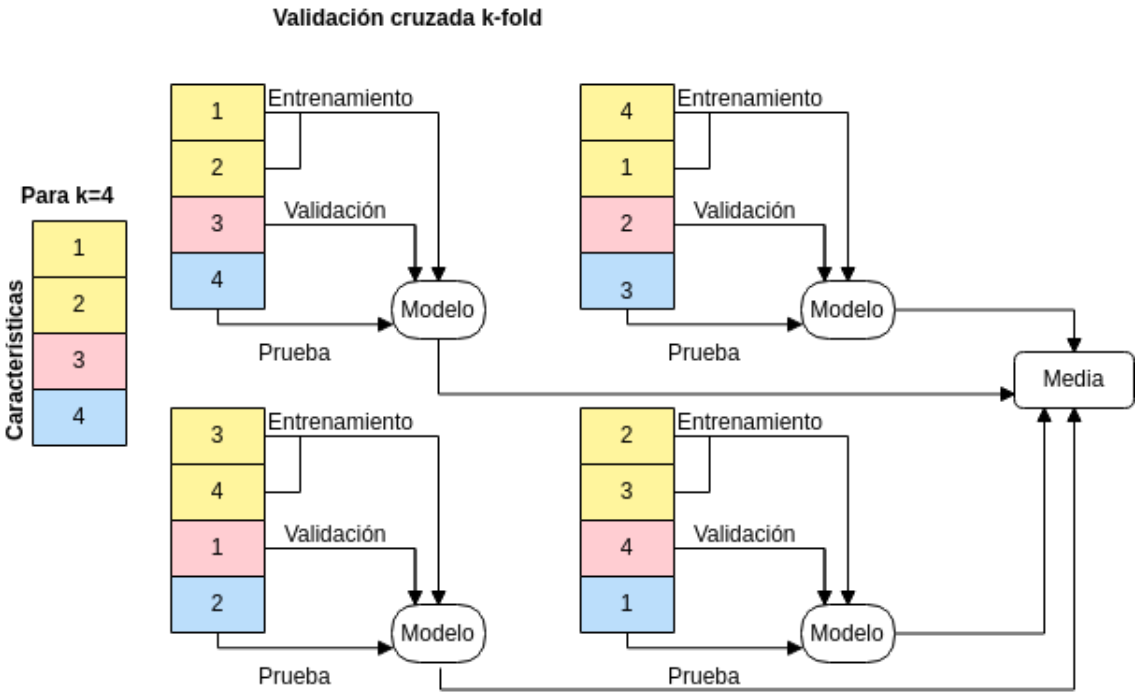


Figura 2.6: Validación cruzada

2.4.2. Clasificación

El proceso de clasificación utiliza un conjunto de instancias $(x_1, x_2, ...x_m)$ pertenecientes al espacio de entradas X, donde cada una ha sido etiquetada como perteneciente a una de las clases posibles. Así, para cada instancia x_i, se asigna la etiqueta y_i [2].

Con un conjunto de pares $S = (x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$, que aporta información para generalizar un modelo con los criterios para la clasificación, el modelo debe ser capaz de asignar la etiqueta de pertenencia cuando se somete a nuevos grupos de instancias tal como lo muestra la Fig. 2.7 [2].

Para entender el comportamiento del clasificador se define dos terminos que reflejan su forma de entrenamiento basado en la información proporcionada.

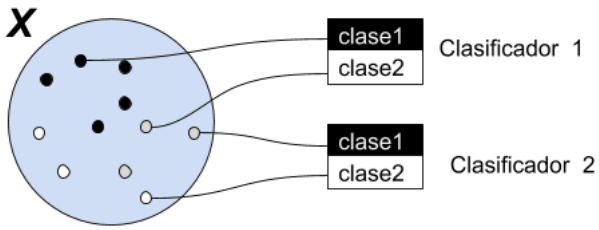


Figura 2.7: Clasificación de muestras usando diferentes clasificadores

Sobreajuste(alta varianza):mapea casi perfectamente la tendencia de los datos de entrenamiento, pero puede fallar en la generalización de nuevos registros. Para reducir el sobreajueste se opta por reducir el número de características o eliminar las no relevantes y seleccionar otro tipo de algoritmo que maneje mejor el problema [22].

Subajuste: mapea pobremente la tendencia de los datos de entrenamiento, con lo cual también tiene problemas con la generalización de nuevos registros. Para disminuir el subajuste se aumenta el número de características relevantes o se incluye un número mayor de instancias [22].

2.4.3. Algoritmos para clasificación

2.4.3.1. K- vecinos más cercanos (KNN, k-nearest neighbor)

K-vecinos más cercanos (KNN) es un método simple, se ha aplicado a varios problemas de clasificación enfocados en el reconocimiento de actividades de personas u otros elementos como estados de bicicletas en [23].

La ventaja de K-NN es la facilidad para obtener el modelo. Parte de un conjunto de datos de entrenamiento $(x_1^{train},...x_i^{train})$ con sus respectivas etiquetas $(y_1^{train},...,y_i^{train})$ que sirven como información para generar el modelo. Para cada observación del conjunto de prueba o validación $(x_1^{test},...x_i^{test})$, calcula la distancia entre el conjunto de datos de entrenamiento para la observación de prueba y los almacena en el conjunto de salida NK. El resultado final es el voto mayoritario de las clases para los K puntos más cercanos e identifica la clase de la observación de prueba detallado en Fig. 2.8. Calcular el promedio en la Ecuación 2.1 es equivalente a tomar el voto mayoritario en el caso de clasificación [4].

$$y_j^{test} = \frac{1}{k} \sum_{x_j^{train} \varepsilon Nk} y_j^{train} \tag{2.1}$$

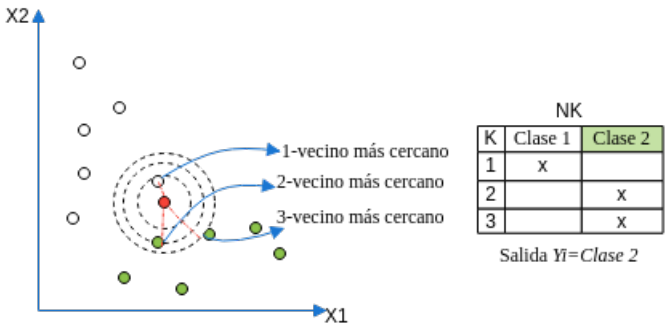


Figura 2.8: K-vecino más cercano

2.4.3.2. Máquina vectorial de soporte (SVM, support vector machine)

SVM o clasificador de margen grande, determina el mejor límite de decisión posible debido que proporciona el mayor espacio entre las clases. Esta característica contribuye a una mayor confianza en la resolución de problemas de clasificación [4].

Para construir el modelo SVM, se necesita encontrar el hiperplano óptimo de separación entre las clases como se muestra en la Fig. 2.9. Hiperplano óptimo es aquel con el margen máximo entre los vectores soporte, cuanto mayor sea el margen, mejor es el clasificador [2].

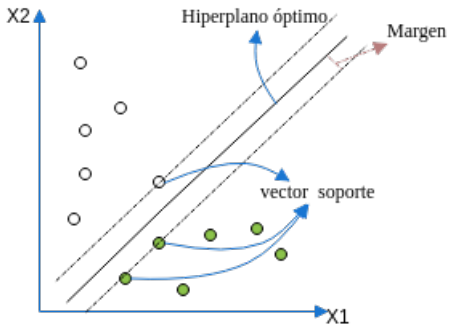


Figura 2.9: Hiperplano óptimo

El hiperplano óptimo se obtiene de un conjunto de datos de entrenamiento y requiere determinar el mayor margen mediante funciones de decisión. Cuando el conjunto de entrenamiento no puede separarse mediante una función lineal, se acude a transformaciones no lineales que llevan del espacio de entrada a otro espacio de alta dimensionalidad, en el que los datos son separables como se gráfica en la Fig. 2.10 [2].

Para solución de problemas multiclase con algoritmos SVM es necesario emplear estrategias tales como:

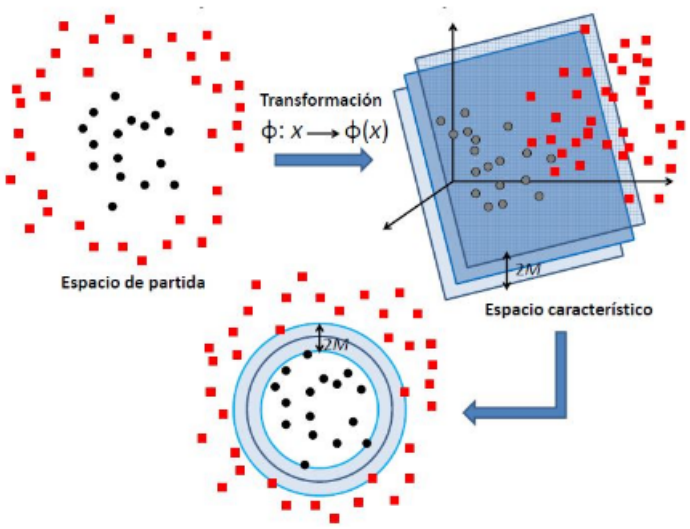


Figura 2.10: Transformaciones no lineales

[24]

La estrategia de uno-vs-uno consiste en generar un total de $K(K-1)/2$ SVMs, donde K es el número de clases, compara todos los posibles pares de clases. Para generar una predicción se emplean cada uno de los $K(K-1)/2$ clasificadores, registrando el número de veces que la observación es asignada a cada una de las clases. Finalmente, se considera que la observación pertenece a la clase a la que ha sido asignada con más frecuencia [25].

Estrategia uno vs todo, consiste en ajustar K SVMs distintos, cada uno comparando una de las K clases frente a las restantes $K-1$ clases. Como resultado, se obtiene un hiperplano de clasificación para cada clase. Para obtener una predicción, se emplean cada uno de los K clasificadores y se asigna la observación a la clase para la que la predicción resulte positiva [25].

2.4.3.3. Redes neuronales artificiales(ANN, artificial neural networks)

En el aprendizaje de máquina, las redes neuronales artificiales (ANN) se utilizan para estimar o aproximar funciones lineales y no lineales desconocidas que dependen de un gran número de entradas [23].

Las redes neuronales artificiales pueden calcular valores o devolver etiquetas usando entradas. Una ANN consta de varias unidades de procesamiento, llamadas neuronas, que están dispuestas en capas. En ANN las neuronas están conectadas mediante conexiones dirigidas que permiten el flujo de información en la dirección desde la capa de entrada a la capa de salida.

Una neurona k en la capa m recibe una entrada de cada neurona j en la capa $m-1$ ver Fig. 2.11. La neurona agrega la suma ponderada de sus entradas a un término sesgo o peso; luego aplica todo a una función de transferencia y pasa el resultado a su salida hacia la última capa. En general, la ANN requiere la definición del número de capas, el número de neuronas en cada capa y la función de transferencia de la neurona. Dado el conjunto de datos de entrenamiento, la ANN puede utilizar el algoritmo de aprendizaje, como la propagación hacia atrás, para conocer los pesos de cada neurona individual que minimicen el error [23].

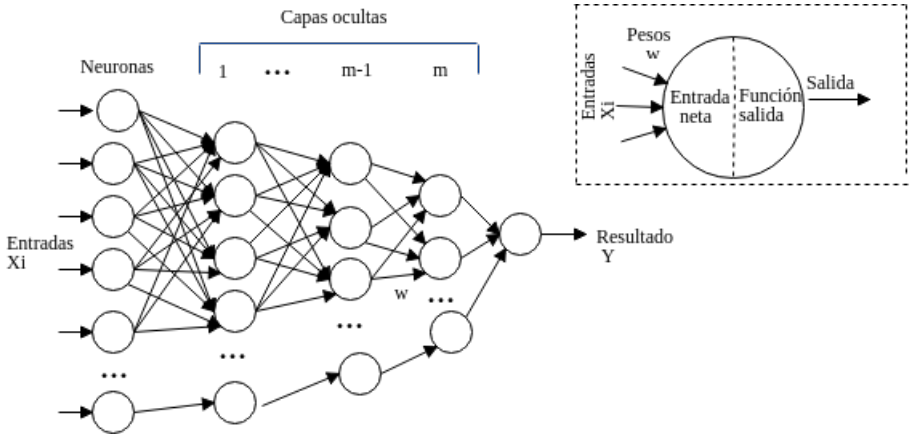


Figura 2.11: Estructura de una red neuronal

Capítulo 3

Metodología

En este capítulo se realiza una revisión condensada de metodologías para resolver problemas de reconocimiento con aprendizaje de máquina.

3.1. Introducción

Aprendizaje de máquina contempla un conjunto de técnicas y algoritmos para generar información de un conjunto de datos. Esta información permite estimar cierto conocimiento desconocido de un sistema, basado en un conjunto de observaciones de entradas y salidas [1]. Todo proceso de aprendizaje sigue una metodología, así para este trabajo se considera la expuesta en [22]. En la Fig. 3.1 detalla los principales consideraciones a llevar a cabo dentro de un proceso de aprendizaje.

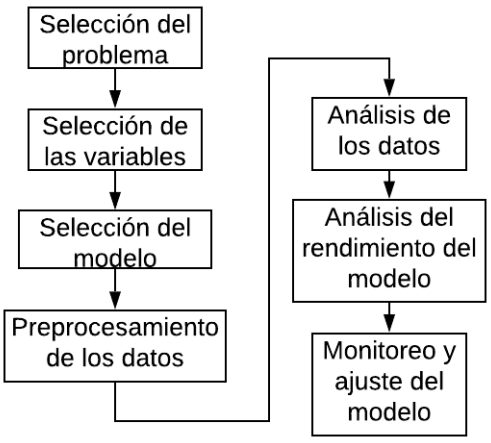


Figura 3.1: Metodología de aprendizaje de máquina

3.2. Selección del problema

La selección del problema parte de un análisis previo, fundamentalmente con la determinación del tipo de información disponible, la veracidad de la fuente de los datos y su disponibilidad sujeta a posibles variantes en el entorno, que permitan a posterior una correcta selección de las variables a utilizar. La mayoría de las fuentes de los datos para problemas de aprendizaje de máquina son sensores que deben ser sujeto a una respectiva etapa de acondicionamiento de señales. Este estudio previo genera un mejor visión en etapas posteriores para la selección de los métodos a implementar [22].

3.3. Selección de variables

De acuerdo con [22], para iniciar con el desarrollo de un modelo de aprendizaje es conveniente utilizar un número reducido de variables que deriven entre 10 a 20 características para evitar tiempos altos en el procesamieto de la información. En la selección de las variables a utilizar se debe tener en cuenta que sean de fácil consecución y extracción. Las variables más comunes en problemas de reconocimiento son las provenientes de acelerometros, giroscopios y GPS. Con el modelo debidamente entrenado es posible aumentar el número de características siempre y cuando las nuevas variables aporten con información extra que genere un modelo más robusto.

3.4. Selección del modelo

Con la información de la problemática correctamente analizada, se determina a que tipo de modelo de aprendizaje de máquina se ajusta el problema. Generalmente con aprendizaje de máquina se pretende resolver problemas de regresión, clasificación, clustering, entre otros. Este trabajo se centra en un problema de clasificación con aprendizaje de tipo supervisado, por lo tanto, su información esta organizada de la forma expuesta en la Tabla 2.1.

3.5. Preprocesamiento y extracción de las características

El desarrollo de un modelo de aprendizaje de máquina parte de crear un conjunto de características para que los algoritmos puedan trabajar. En ocasiones, los datos se encuentran sin procesar y ordenar (señales de sensores), por lo que se debe ordenar y extraer características relevantes que definen a cada variable medida [20].

En [26], realiza un análisis comparativo entre la segmentación por ventanas superpuestas y ventanas adyecentes demostrando que no existe mayor ventaja entre una y otra, simplemente que, con ventanas superpuestas incrementa el tiempo de procesamiento pero existen menos posibilidades de pérdida de información, por lo que se opta por las ventanas superpuestas por su menor tiempo de procesamiento.

3.5.1. Extracción de características

La señal contiene información redundante en cada ventana de tiempo, por lo que trabajar directamente con la misma no es la opción óptima. La extracción de características sintetiza la señal en información más relevante que representan a las variables medidas [17].

El conjunto de características normalmente utilizado en problemas de aprendizaje de máquina como el reconocimiento de estados de ciclistas en [4] y [23] son medidas estadísticas, a continuación en la Tabla 3.1 se resumen las más relevantes.

Tabla 3.1: Mediciones usadas para crear características

Modelo	Medición
1	Media
2	Varianza
3	Desviación Estandar
4	Máximo
5	Mínimo
6	Rango Intercuartil

El conjunto de características seleccionado y su número responden a un equilibrio entre la necesidad de minimización de la información, y al error en la representación de la señal real [17].

Para cada conjunto de características que se extrae de cada ventana de tiempo, se agrega a un único vector de características que se utiliza como conjunto de entrada para un clasificador. Cada conjunto de características debe estar correctamente etiquetado.

3.5.2. Normalización

La normalización tiene como finalidad modificar el rango de valores de las características o atributos y garantizar que se encuentren dentro de una misma escala que conduce a elevar el desarrollo del modelo y aumentar la velocidad de entrenamiento [4].

Para la mejora de la eficiencia de los modelos generalmente se opta por normalizar en la escala entre 0 y 1 mediante un reescalado con la Ecuación 3.1, utiliza el valor máximo y mínimo de la variable a normalizar [27].

$$xi(normalizado) = \frac{x_i - min(x)}{max(x) - min(x)} \tag{3.1}$$

En algunos trabajos sus autores optan por normalizar en la escala de -1 a 1 como es el caso de Jahangiri mediante la ecuación 3.2, emplea el valor medio y la desviación estandar de la variable a normalizar [4].

$$xi(normalizado) = \frac{x_i - mean(x)}{std(x) - mean(x)} \tag{3.2}$$

Con las características dentro de un mismo rango permite al modelo independientemente del tipo, realizar cálculos más fácilmente, y que las características con rangos de valores más altos no dominen sobre los bajos [27].

3.6. Desarrollo del modelo

Este proceso es conocido también como etapa de aprendizaje o entrenamiento, consiste en establecer un modelo utilizando el conjunto de características. Esta etapa normalmente necesita de una medida que permita comparar cuán bien se ajusta cada modelo a los datos o bien de una medida que permita cuantificar la fiabilidad de la predicción del modelo ante nuevos datos [2].

3.6.1. Metodologías de planteamiento y evaluación del modelo

En la solución de problemas con aprendizaje de máquina una de las fases principales es la división de las características en conjuntos de entrenamiento, validación y prueba. La metodología empleada es importante, especialmente cuando la cantidad de los datos etiquetados son pequeños debido que para el conjunto de características de prueba se debe asegurar que no sea un representante preciso de los datos de entrenamiento. Para los casos en que los datos etiquetados son pequeños, se requieren cuidadosas variaciones metodológicas tales como: hold out y la validación cruzada para evitar evaluaciones erróneas [20].

3.6.2. Entrenamiento

Para el entrenamiento del modelo de clasificación se consideran tres algoritmos como: máquina soporte vectorial, k-vecinos mas cercanos, la implementación de una red neuronal

artificial y se considera un conjunto de características X de N instancias u observaciones correctamente etiquetadas.

Luego de someter a los métodos de división a las características y etiquetas. En esta sección, exclusivamente se utiliza el conjunto destinado para el entrenamiento como variables de entrada para cada algoritmo.

3.6.2.1. Clasificador KNN

Es un clasificador multiclase. En la obtención del modelo se considera como parámetros fundamentales la distancia métrica y el valor K (número de vecinos más cercanos) donde $K \in \mathbb{N}$.

Generalmente en este tipo de clasificador el rendimiento está ligado al número de K vecinos más cercanos propuesto. El valor K depende fundamentalmente de las características del tipo de problema. El valor óptimo de K es el que muestra el mejor rendimiento en el clasificador luego de realizar una serie de entrenamientos y validaciones modificando su valor, K suele estar entre 5 y 20 [4].

La distancia euclidiana es la métrica típica en problemas de clasificación, sirve para el cálculo de la distancia entre las k características de entrenamiento x^{train} y la característica de evaluación x^{test} [4].

3.6.2.2. Clasificador SVM

Es un modelo capaz de identificar a qué clase pertenece un nuevo elemento que aparezca en el sistema. Máquina de soporte vectorial representa cada elemento como un punto en el espacio, separando las clases a espacios lo más amplios posibles, mediante hiperplanos de separación llamados vectores de soporte [3].

SVM aprovecha algunas funciones conocidas como kernels que devuelven el producto interno del vector en el espacio Z deseado para separar cada clase. Existen diferentes tipos de kernels, tal como: el lineal, polinómico y gaussiano [4].

Para cierto tipo de problemas, SVM puede producir mejores resultados con kernels más avanzados, como es el kernel gaussiano. Para datos con gran cantidad de características, se considera conveniente al kernel gaussiano como el más apropiado [28]. Si se realiza una selección completa del modelo, no hay necesidad de probar con el kernel lineal porque sus resultados obtenidos están contenidos en los resultados con el kernel gaussiano [29].

La formulación del kernel gaussiano se muestra en la Ecuación 3.3. El rendimiento del clasificador depende de su coeficiente de gauss σ, el mismo que se debe variar hasta encontrar su valor óptimo.

$$K(x,x) = exp\frac{|x-x'|^2}{2\sigma} \tag{3.3}$$

3.6.2.3. Clasificador con una red neuronal artificial

Los modelos con redes neuronales, parten con la definición de una topología adecuada según el problema a resolver. Si bien no existen reglas estrictas a seguir, sin embargo existen ciertas recomendaciones que permitan que el diseño de la red sea la mejor. Una red neuronal artificial tradicional tiene su capa de entrada,su capa de salida y al menos una capa oculta [30].

No existe razón teórica que determinen la cantidad necesaria de capas ocultas. Gran cantidad de problemas de la vida han sido resueltos satisfactoriamente con redes de hasta dos capas ocultas. El alto número de capas puede ocasionar un sobreajuste [30].

El número de neuronas en la capa oculta es una aproximación de la Ecuación 3.4 que refleja la regla de la pirámide geometría que resume muchas redes prácticas donde su estructura sigue una forma piramidal como en la Fig. 3.2 [30].

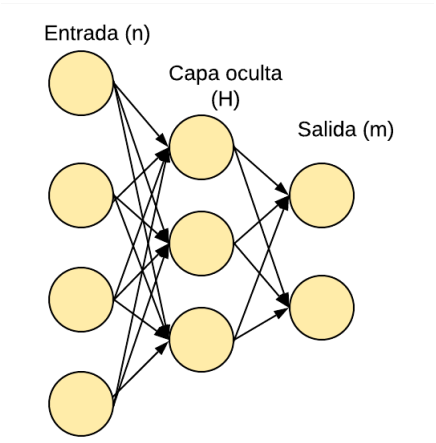


Figura 3.2: Estructura de la red neuronal artificial

$$Hidden = \sqrt{m * n} \tag{3.4}$$

La red neuronal artificial almacena información en forma de pesos. Por lo tanto, para el entrenamiento con las características de entrada, existe una variación en los pesos para que representen la nueva información. El enfoque sistemático para modificar los pesos de acuerdo con la información dada se denomina regla de aprendizaje que se resumen en los siguientes pasos [31].

- Inicialización de los pesos a valores adecuados.
- Ingreso de los datos de entrenamiento configurados como entrada y salida correcta a la red neuronal, con su resultado de salida calcula el error a partir de la salida correcta.
- Ajuste de los pesos para reducir el error.
- Repetición de los pasos 2-3 para todos los datos de entrenamiento.

3.7. Métricas de evaluación

Un modelo de clasificación coloca nuevas instancias en las categorías con las que se ha entrendo. La medida más común de la calidad del clasificador es la precisión. Para medir el rendimiento del clasificador, se utiliza una herramienta realmente útil llamada matriz de confusión que es la base de varias medidas de evaluación de rendimiento [3].

Matriz de confusión La matriz de confusión contiene información de los recuentos de las predicciones realizadas por el clasificador ubicadas en los diferentes niveles de pertenencia que se ilustra en la Tabla 3.2. En algunos casos la matriz contiene los porcentajes de algunas métricas básicas [21].

Tabla 3.2: Estructura de la matriz de confusión

		Predicción	
		Positivos	Negativos
Valor Real	Positivos	VP	FN
	Negativos	FP	VN

Indicadores

- Verdaderos Positivos (VP): Todos los elementos clasificados correctamente.

- Verdaderos Negativos(VN): Todos los elementos discriminados correctamente.
- Falsos Positivos (FP): Detectados Incorrectamente.
- Falsos Negativos (FN): Elementos que no se han detectado.

A partir de los datos obtenidos en la matriz de confusión, es posible calcular las siguientes métricas. Valores altos en VP y VN indican un rendimiento aceptable del modelo.

Precisión total: Porcentaje total de aciertos que ha tenido el modelo en su evaluación con un nuevo conjunto de datos.

$$PT = \frac{VP+VN}{Total} * 100\,\% \tag{3.5}$$

Tasa de error: Contabiliza la cantidad de objetos de clasificación incorrecta dividido para el total de elementos introducidos en el clasificador.

$$E = 100\,\% - PT \tag{3.6}$$

Precisión por clase: Porcentaje que indica que las predicciones de pertenencia a la categoría "Verdadero" realmente lo son.

$$PC = \frac{VP}{VP+FP} * 100\,\% \tag{3.7}$$

Recall: Porcentaje del total de instancias de pertenencia a la categoría "Verdadero" que han sido detectadas.

$$R = \frac{VP}{VP+FN} * 100\,\% \tag{3.8}$$

Especificidad: Porcentaje de las instancias que se han predicho en la categoría "Falso" que realmente lo son. Esta métrica resulta especialmente crítica en ámbitos como el militar o el médico.

$$Ep = \frac{VN}{VN+FP} * 100\,\% \tag{3.9}$$

Capítulo 4

Resolución del problema

4.1. Selección de la base de datos

En la revisión de experimentos centrados en el reconocimiento de actividades de personas, se encuentra variedad de conjuntos de datos que sintetizan la información de la interacción entre un entorno determinado y la persona . Para este problema se selecciona la base de datos que cumpla con los siguientes parámetros:

- Conjunto de datos obtenidos en ambientes clínicos.
- Realización de actividades comunes (caminar, sentarse, entre otras).
- Cada instancia registrada en la base de datos debe tener su etiqueta de pertenencia a una actividad.
- La información es certificada con más trabajos de investigación basados en está.
- Acceso libre a la información.

Los parámetros se definen en base a la problemática planteada en este trabajo en la sección 1.4. Se considera varias bases de datos del repositorio digital U.C. Irvine tal como: ”el reconocimiento de la actividad humana utilizando teléfonos inteligentes”, ”sistema de reconocimiento de actividad basado en la fusión de datos multisensor”, pero la base de datos “reconocimiento de actividad con personas mayores sanas utilizando un sensor portátil sin batería” en [32], es generada en un ambiente clínico y mantiene su información organizada en instancias etiquedas, siendo la base que cumple con los parámetros planteados.

4.2. Base de datos

La base de datos contiene información de 14 personas mayores en buen estado de salud de entre 66 y 86 años, que han realizado actividades con indicaciones generales utilizando un sensor portátil sin batería, en la parte superior de su ropa a nivel del esternón en un ambiente de sala clínica. Los datos están distribuidos en 9161 instancias con 7 atributos cada una, como lo

indica la Tabla 4.1 en archivos csv [32].

.

Las actividades realizadas fueron:

- Caminar a la silla.
- Sentarse en la silla.
- Levantarse de la silla y la cama.
- Caminar a la cama.
- Acostarse en la cama.
- Caminar hacia la puerta.

Por lo tanto, las posibles etiquetas de clase asignadas para cada observación de sensor son:

- Sentado en la cama.
- Sentado en la silla.
- Recostado en la cama
- Caminar, donde la caminata incluye estar de pie y caminar por la habitación.

Tabla 4.1: Información para el aprendizaje supervisado

Tiempo	Aceleración vertical	Aceleración frontal	Aceleración lateral	Fase	Frecuencia	Etiqueta
T_1	$ac.vert._1$	$ac.front._1$	$ac.lat._1$	Fa_1	f_1	E_1
T_2	$ac.vert._2$	$ac.front._2$	$ac.lat._2$	Fa_2	f_2	E_2
...	...	...	...	...	...	...
T_n	$ac.vert._n$	$ac.front._n$	$ac.lat._n$	Fa_n	f_n	E_n

4.2.1. Análisis de la base de datos

La información considerada útil para el desarrollo del modelo es la aceleración en sus 3 ejes con su etiqueta respectiva. Gráficamente se observa en la Fig. 4.1 una sección del conjunto de datos, donde se distingue la señal de las aceleraciones y su etiqueta correspondiente a la actividad. La etiqueta de la señal esta representada en valores enteros de 1 a 4 de la siguiente manera: 1.- sentarse en la cama, 2.-sentarse en la silla , 3.-acostarse y 4.-caminar.

En Fig. 4.1 permite visualizar el patrón generado por las señales para cada actividad. El patrón para cada actividad es diferente, por lo que genera un indicio para la determinación de los datos como adecuados para la obtención del modelo y que cada clase pueda ser clasificable con facilidad.

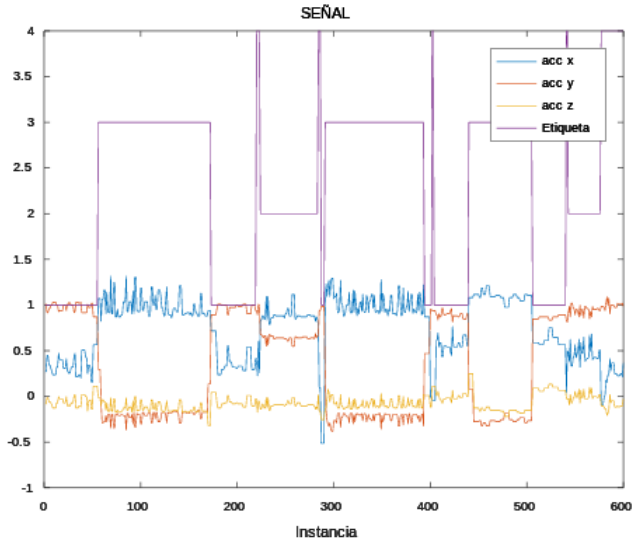


Figura 4.1: Sección de los datos con su etiqueta correspondiente

Figura 4.2: Preprocesamiento de la información

El primer punto en la solución del problema es el preprocesamiento de la información, la base de datos se organiza en grupos de pertenencia a cada actividad, para extraer las características, se aplica la normalización y división de datos para entrenamiento y prueba. El proceso se describe mediante el diagrama de flujo en Fig. 4.2. En la sección A.1.1 hasta A.1.4 se incluye ladescripción del programa que detalla todo el proceso de preprocesamiento.

4.2.2. Normalización de la señal

La normalización a el rango entre 0 a 1, emplea la Ecuación 3.2 utilizada por Jahangiri en [4] a problemas de reconocimiento de actividades. Gráficamente se observa en la Fig. 4.3 una señal que mantiene iguales características iniciales, pero con la magnitud de cada instancia ajustadas a valores entre 0 y 1.

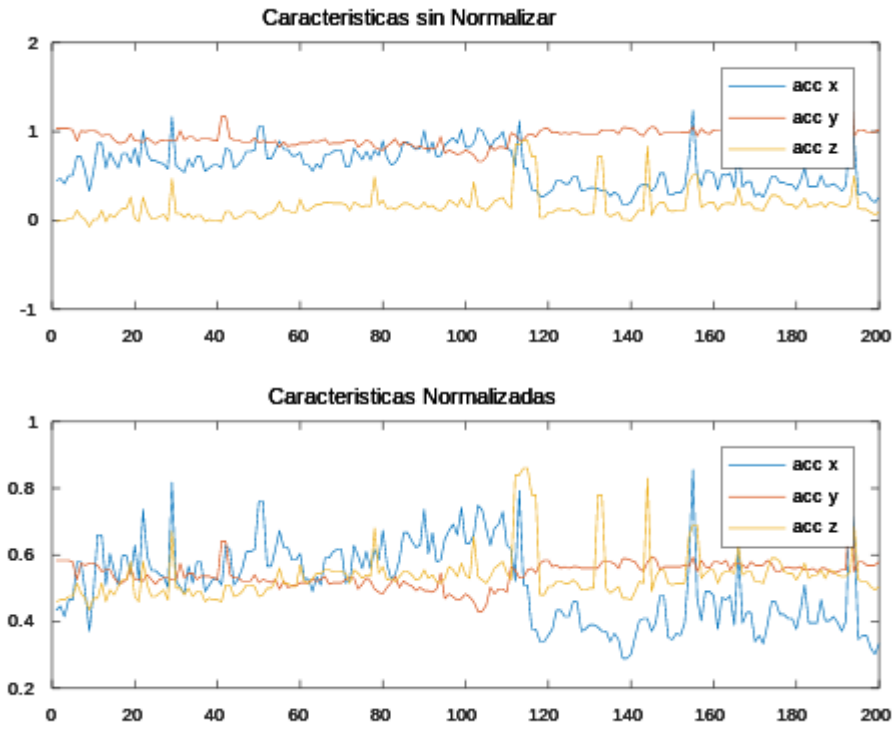


Figura 4.3: Señal normalizada

4.3. Extracción de las características

El proceso de extracción de las características se realiza con la generación de ventanas de tiempo y la obtención de métricas que reflejen la información de la señal. Para este problema se

utiliza la técnica de serie de tiempo superpuesta, conjuntamente con la aplicación de medidas estadísticas expuestas en la Tabla 3.1.

$$Tw >= 1/f$$
$$Tw >= 1/10$$
$$Tw >= 0{,}1s$$

De la relación se tiene el tamaño mínimo de ventada $Tw = 0{,}1s$, basado en el criterio propuesto en la metodología se considera adecuado establecer ventanas de tiempo de $1segundo$, generando grupos de 10 instancias cada uno, que derivan el conjunto de características calculadas con las medidas estadísticas.

4.4. Obtención de los modelos

4.4.1. División de los datos

La función *crossvalind* de tipo hold out disponible en Matlab ayuda en la partición de las instancias. El método elije aleatoriamente las instancias u observaciones que serán reservadas para el conjunto de prueba. *Crossvalind* devuelve vectores de índice lógico [1 (entreamiento), 0 (prueba)], que al relacionar con el vector de características totales genera la división. *Prueba* es de tamaño $(P*N)$ y $entrenamiento$ de tamaño $((1-P)*N)$. N es el número total de instancias y P es la proporción para la división que debe ser un escalar entre 0 y 1. P se omite a 0.5 cuando corresponde a dividir con el 50 % tanto para entrenamiento como para prueba.

Para el desarrollo de este trabajo se utiliza una división equivalente al 70 % para el conjunto de entrenamiento y el 30 % restante para el conjunto de prueba.

4.4.2. Modelo con KNN

En la generación del modelo KNN se utiliza la función $fitcknn$ disponible en el software para la creación del modelo de clasificación KNN con los datos para el entrenamiento. En la Tabla 4.2 se detalla los parámetros iniciales para el modelo.

Como se menciona en el capítulo anterior, es importante ser capaces de configurar los parámetros iniciales del modelo como son: la distancia métrica y el valor de k vecinos más cercanos. El software permite configuración de los parámetros $'Distance'$ y $'NumNeighbors'$ como variables de la función. La Fig.4.4 muestra el diagrama de flujo del proceso de entrenamiento y prueba del modelo KNN, la descripción de la programación se encuentra en la sección A.1.5.

Tabla 4.2: Parámetros del modelo KNN

Parámetro	Valor
ClassNames	'[1 2 3 4]'
ScoreTransform	'none'
NumObservations	5260
Distance	'euclidean'
NumNeighbors	10

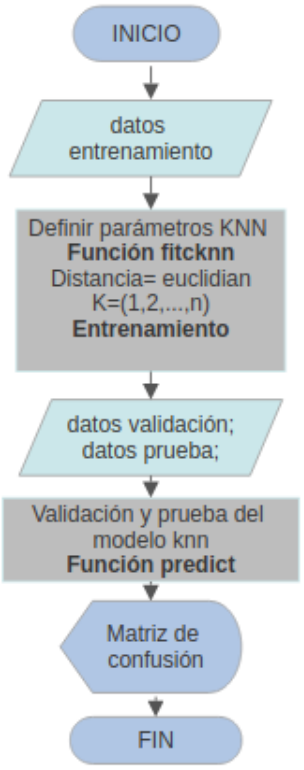


Figura 4.4: Entrenamiento, validación y prueba de modelo KNN

4.4.3. Modelo con SVM

En la solución del problema planteado con un clasificador SVM se utiliza la función $templeteSVM$ que devuelve una plantilla de aprendices (Learners) adecuada para entrenar modelos multiclase, donde es posible configurar varios parámetros como los expuesto en la Tabla 4.3. Los principales parámetros a considerar son: $'KernelFunction'$ y $'KernelScale'$ ya que de acuerdo a su variación se mide el desempeño del clasificador.

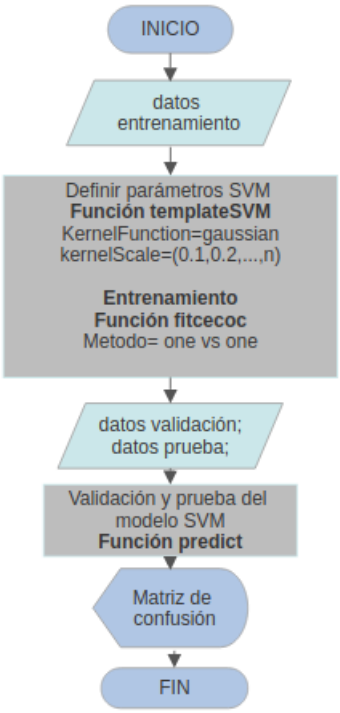


Figura 4.5: Entrenamiento, validación y prueba de modelo SVM

Tabla 4.3: Parámetros de la función templeteSVM

Parámetro	Valor
KernelFunction	'gaussian'
KernelScale	0.1, 0.2,...,0.9
Type	'classification'
Method	'SVM'

Para el entrenamiento de la plantilla anterior es apropiado generar un modelo de aprendizaje, para ello se utiliza la función $fitcecoc$ que obtiene el modelo completamente entrenado,

utilizando las variables predictivas del conjunto de entrenamiento. Utiliza máquinas de vectores de soporte binarios (SVM) utilizando un esquema de codificación uno a uno.

Tabla 4.4: Parámetros de la plantilla de aprendizaje

Parámetro	Valor
Coding	'onevsone'
Type	'classification'
Learners	'templateSVM'
ClassNames	'[1; 2; 3; 4]'

El desarrollo del modelo se describe en Fig. 4.5, comprende la etapa de definición de los parámetros, el entrenamiento, prueba del modelo y mostrar su matriz de confusión. La descripción de la programación se detalla en la sección A.1.5.

4.4.4. Modelo con ANN

En la obtención del modelo para la clasificación con redes neuronales, el software tiene gran variedad de formas de implementación como: el uso del toolbox *nprtool* que no es más que una interfaz gráfica de usuario que realiza el reconocimiento de patrones y otra con el uso de línea de comandos que permite la configuración de ciertos parámetros para personalizar la estructura y capacitación de la red. Para este problema en particular se emplea el uso de comandos.

Inicialmente se debe organizar el conjunto de características y su vector de etiquetas, a diferencia de los modelos anteriores se necesita formar un vector de *nxm* para categorizar la pertenencia de cada instancia a su respectiva clase en valores de 0 y 1 como lo muestra Tabla 4.5, *n* es el número de clases y *m* el total de instancias.

Tabla 4.5: Vector de etiquetas *nxm* de la red neuronal

		Instancias									
	#	1	2	3	4	5	6	7	8	9	...
	Clase	1	1	1	2	2	3	3	2	3	...
Vector	Clase 1	1	1	1	0	0	0	0	0	0	...
	Clase 2	0	0	0	1	1	0	0	1	0	...
	Clase 3	0	0	0	0	0	1	1	0	1	...

Se crea la estructura de la red neuronal por medio de la función *patternnet*, donde se configura la cantidad de capas ocultas y el número de neuronas que conforman la red neuronal como en la Fig. 4.6. Al tratarse de un problema simple, utiliza una sola capa oculta con 4 neuronas. La cantidad de neuronas en la capa oculta es resultado de la aplicación de la regla de la pirámide geométrica con la Ecuación 3.4.

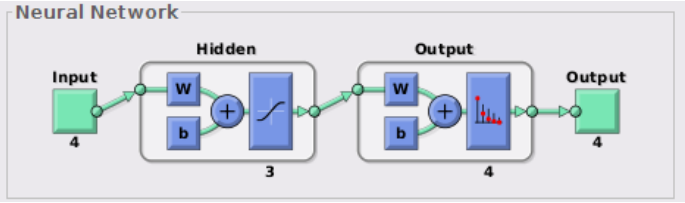


Figura 4.6: Estructura de la red neuronal artificial

En el entrenamiento de la red para el reconocimiento de actividades se utiliza la función *trainscg* debido a su bajo requerimiento de memoria y su rapidez a comparación de algoritmos de descenso estándar. Finalmente, con la red ya entrenada se somete a la prueba del modelo con un nuevo conjunto de datos.

El flujograma en Fig. 4.7, detalla el proceso necesario para entrenar y evaluar el modelo. La programación se describe en la sección A.1.5.

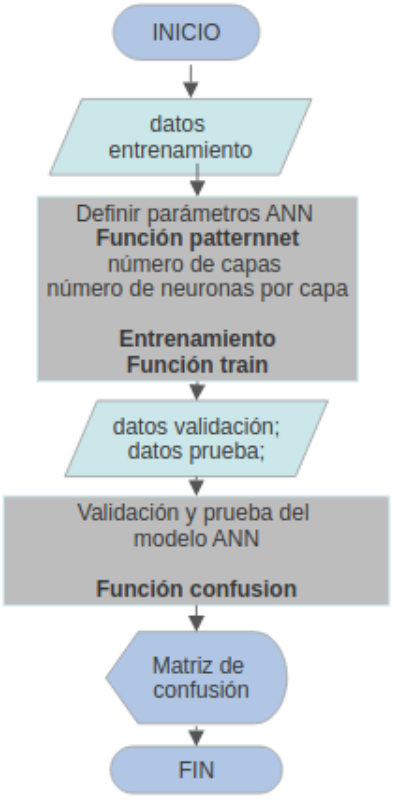


Figura 4.7: Entrenamiento, validación y prueba de modelo ANN

Capítulo 5

Análisis de los resultados

En este capítulo se presentan resultados finales de la evaluación de tres algoritmos propuesto en el capítulo 4 para clasificación de actividades de personas. Cada algoritmo fue sujeto a variantes en la configuración de sus parámetros a fin de determinar el menor error de clasificación.

Los resultados expuestos para cada algoritmo son producto de la validación, mientras que para el análisis comparativo se basa en resultados de la evaluación final.

5.1. Resultados de KNN

El modelo KNN se implementó utilizando la división de las características por medio de hold-out, con el 70 % para entrenamiento y 30 % para evaluación. La distancia métrica es del tipo euclidiana, por lo tanto, la tasa de error varia en función del valor de K vecinos más cercanos. Los resultados se visualizan en la Fig. 5.3, donde se aprecia el incremento de la tasa de error a medida del incremento del valor K de 1 a 100.

La tasa de error aceptable en un clasificador según la literatura debe estar alrededor del 10 %. En la Fig. 5.3 se concluye que el clasificador con KNN es admisible hasta cierto valor de K. Si se considera el criterio expuesto en la metodología para valores de K alrededor de 10, el rendimiento del modelo es aceptable.

Como se ha mencionado en capítulos anteriores otro método de evaluación para un clasificador es la división de las características para entrenamiento por validación cruzada y similar a el método de hold out se realiza el cálculo de la tasa de error con la variación de K. En la Fig. 5.3 muestra la variación del error, el mismo que se mantiene en el rango de aceptabilidad para un rango más amplio de K a comparación del modelo con hold out.

En el análisis más profundo del rendimiento del clasificador, primero se ajusta el valor $K = 10$ que corresponde a la tasa de error $E = 15\,\%$. Para una mejor visualización de los resultados se obtiene la Fig. 5.2 que corresponde a la matriz de confusión, en su diagonal principal se

observa la cantidad de instancias clasificadas correctamente.

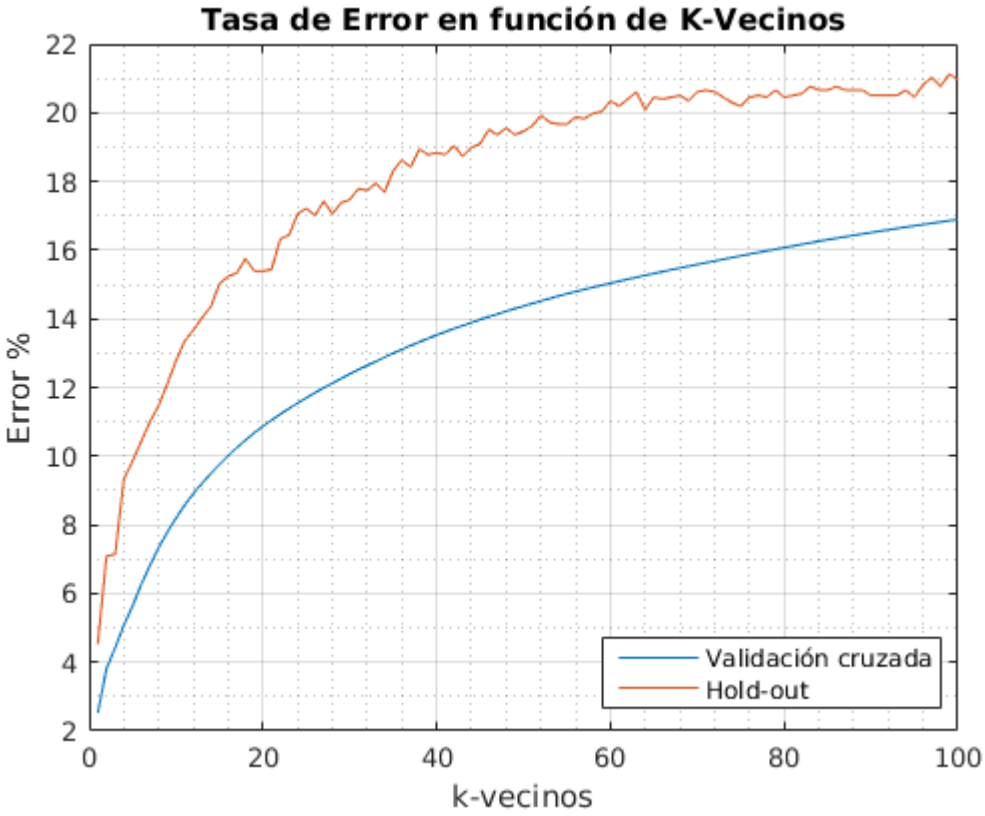


Figura 5.1: Tasa de error - KNN

En la Tabla 5.1 se puede observar los resultados obtenidos de la validación del modelo de clasificación KNN. Para cada clase se obtuvo su precisión, recall y especificidad. La clase 3 (acostado) tiene el 99.79 % de precisión en la clasificación siendo la clase mas diferenciable entre todas.

Existe cierto grado de incertidumbre de la precisión en la clasificación de las clases 1 y 2 (sentarse en la cama y sentarse en la silla) que se asume por la similitud de las actividades.

Los porcentajes generales de recall y especificidad son valores sobre el 79 % que indican que el modelo ha encontrado sus clases con un nivel aceptable.

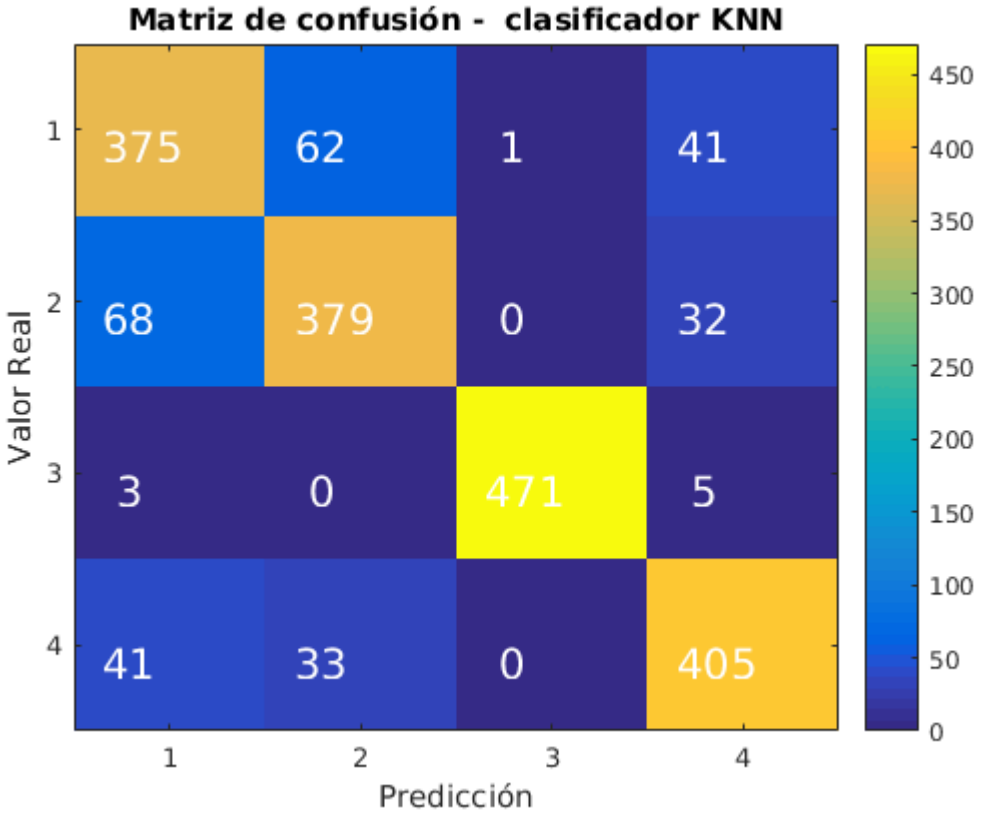


Figura 5.2: Matriz de confusión - KNN

Tabla 5.1: Resultados del modelo KNN

	Clase 1	Clase 2	Clase 3	Clase 4
Precisión	77 %	79,96 %	99,79 %	83,85 %
Recall	78,29 %	79,12 %	98,33 %	84,55 %
Especificidad	91,81 %	92,94 %	99,91 %	94,01 %

5.2. Resultados de SVM

El uso de un clasificador binario SVM como multiclase, implica utilizar métodos de codificación como el uno vs uno o uno vs todo, además de utilizar un kernel gaussiano considerado como el más aceptado en la solución de problemas de clasificación según la revisión literaria. En el desarrollo y validación del modelo se utliza el método hold out para la división de las características similar con el modelo KNN.

SVM es un modelo dependiente de su kernel gaussiano por lo cual su desempeño también varía si su coeficiente lo hace. El cálculo de la tasa de error va en función del coeficiente de gauss y su variación se muestra en la Fig. 5.3. Con la tasa de error y el rango de aceptabilidad del modelo se determina un coeficiente adecuado para el modelo de clasificación.

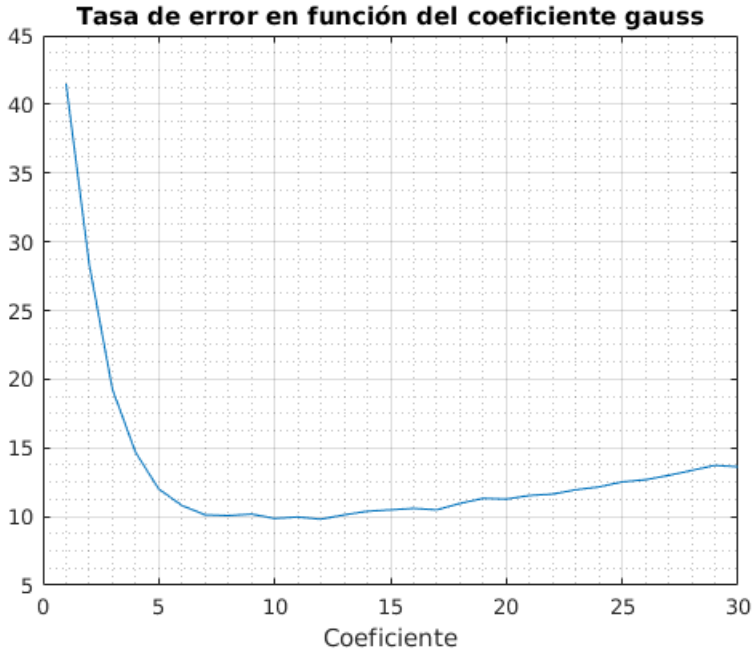


Figura 5.3: Tasa de error - SVM

Gráficamente se determina el coeficiente en $\sigma = 1{,}3$ porque brinda un menor grado de error $E = 11{,}8\,\%$. La validación del coeficiente y el modelo conlleva la obtención de su matriz de confusión y sus respectivos indicadores expuestos en la Tabla 5.4.

El bajo porcentaje de error se refleja en la matriz de confusión en la Fig. 5.4, que muestra en su diagonal principal cerca de la totalidad de las instancias en sus respectivas clases, comprobandose con los altos valores de precisión. Además, las métricas precisión, recall y especificidad, independientemente de las clases tienen porcentajes promedios sobre el 90 % que

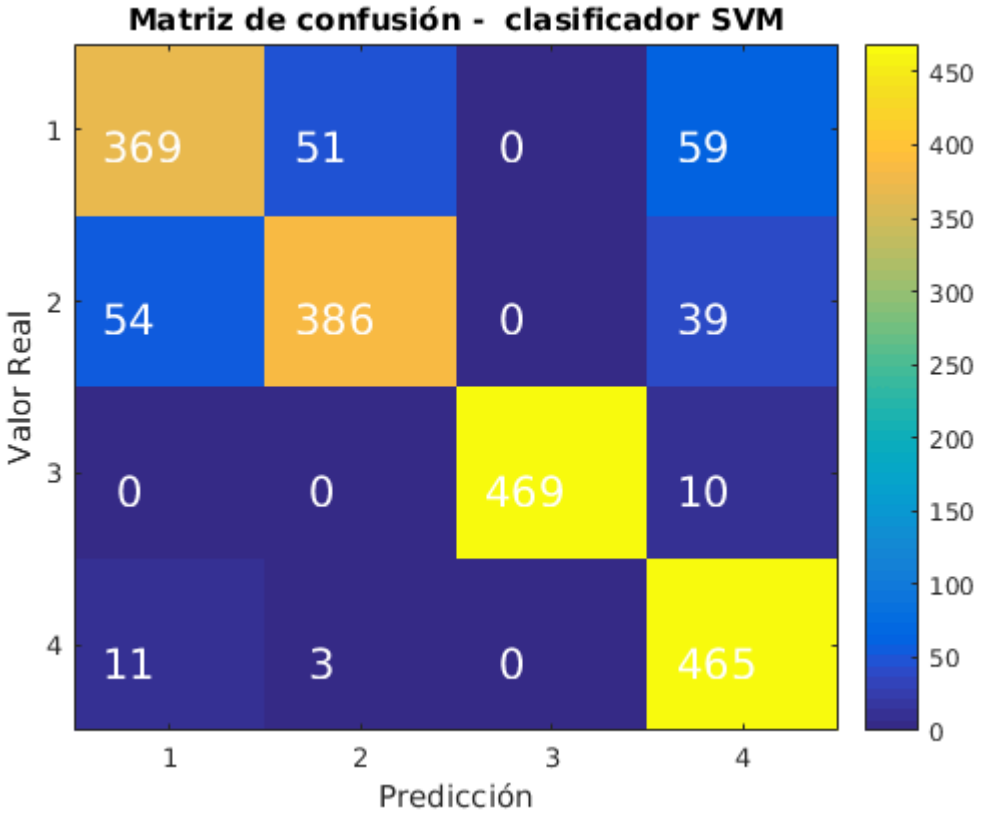


Figura 5.4: Matriz de confusión - SVM

Tabla 5.2: Resultados modelo SVM

	Clase 1	Clase 2	Clase 3	Clase 4
Precisión	85,02 %	87,73 %	100 %	81,15 %
Recall	77,04 %	80,58 %	97,91 %	97,08 %
Especificidad	95,31 %	96,02 %	100 %	91,89 %

lo convierten en un modelo aceptable. Los porcentajes altos en la precisión de cada clase sobre el 80 %, evidencia que al someter a nuevos datos al clasificador encontre con facilidad su clase de pertenencia.

5.3. Resultados de la ANN

Con la topología de la red determinada, su número de capas ocultas y número de neuronas establecidos. Se inicia la etapa de entrenamiento con el grupo de características que deriva un modelo de clasificación. A diferencia de los métodos anteriores, el toolbox para redes neuronales reserva los datos de manera aleatoria para entrenamiento, validación y prueba, no es necesario de otras metodologías de división.

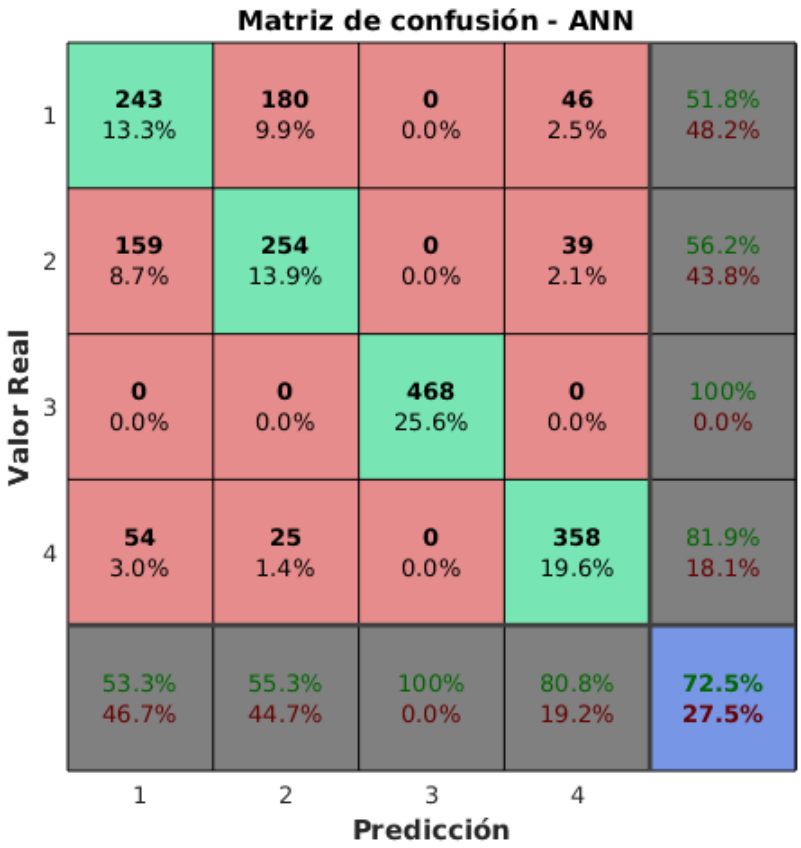


Figura 5.5: Matriz de confusión - ANN

Este modelo tal como los anteriores es sometido a datos para su validación, que refleja porcentajes medios en sus métricas de evaluación. La tasa de error total del modelo corresponde al

E=27.5 % que si bien no es el óptimo, este modelo puede ser aceptable para ciertas aplicaciones que no dependan de un alto porcentaje de precisión en la clasificación.

En la Fig. 5.5 refleja la matriz de confusión correspondiente a la clasificación de los datos para validación del modelo. Los resultados para esta matriz varían cada vez que se ejecute nuevamente el entrenamiento, esto debido al reentrenamiento de la red y la división aleatoria de los datos que realiza el toolbox pero siempre respeta los porciones para cada conjunto de datos(entrenamiento, validación y prueba). Sin embargo, el análisis de la matriz se mantiene, en su diagonal principal se encuentran concentradas las instancias clasificadas correctamente.

En la validación del modelo se obtiene las métricas correspondientes a la clasificación de cada clase expresadas en la Tabla 5.3. La precisión del modelo para este tipo de problema no es la adecuada especialmente para la clase 1 que corresponde al 53.28 % que si se observa la distribución en la matriz se aprecia que existe un total de 159 instancias de la clase 2 que se han predicho en la clase 1. Mientras que la precisión en la clase 2 es de 55.3,3 % que de forma similar 180 instancias de la clase 1 se han predicho por la clase 2. Existe mayor confusión entre estas dos clases por tratarse de actividades similares. En la clase 3 existe 100 % de efectividad del modelo y en la clase 4 tiene un rango aceptable.

Si bien no es el mejor modelo, los porcentajes precisión, recall y especificidad tienden a ser aceptables, por lo tanto es adecuado considerar el modelo para la etapa de evaluación y su respectivo análisis comparativo con los demás modelos.

Tabla 5.3: Resultados del modelo ANN

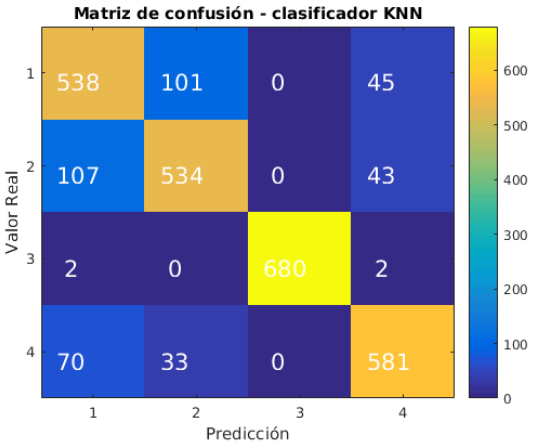
	Clase 1	Clase 2	Clase 3	Clase 4
Precisión	53,28 %	55,33 %	100 %	80,81 %
Recall	51,81 %	56,19 %	100 %	81,92 %
Especificidad	83,52 %	83,90 %	100 %	91,90 %

5.4. Análisis comparativo

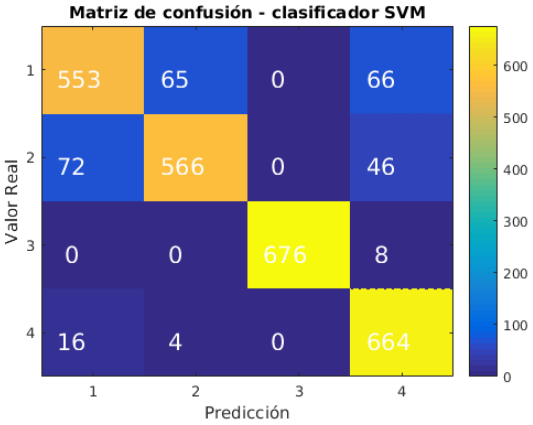
Determinar el modelo de mejor rendimiento trae consigo realizar su evaluación con el conjunto de datos reservado anteriormente.

El análisis final se realiza en base a el 30 % del conjunto total de características reservado para la evaluación. Cada modelo se somete a la evaluación obteniendo las siguientes matrices de confusión de la Fig. 5.6.

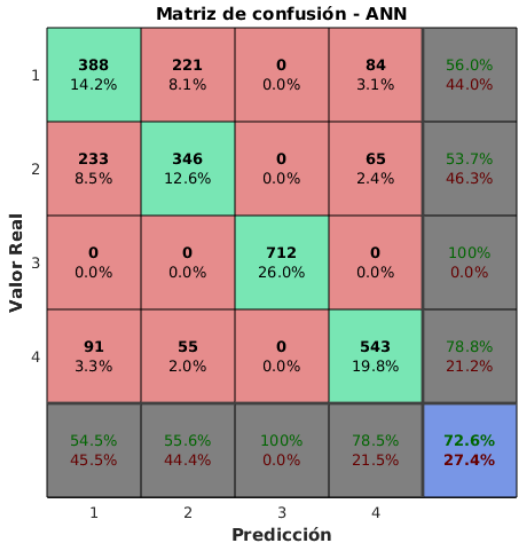
Entre el modelo con KNN y SVM se obtiene gráficamente, que el modelo óptimo entre ellos es el SVM por la escala de colores en su matriz, donde el amarillo en la diagonal principal



(a) M. KNN



(b) M. SVM



(c) M. ANN

Figura 5.6: Matrices de confusión de los modelos.

es un indicador de alto rendimiento (la degradación del color es proporcional al rendimiento). Para la comparación con el modelo con ANN y el resto, es adecuado el uso de las métricas de evaluación descritas en la sección 3.7 que devuelven los resultados detallados en la Fig. 5.7.

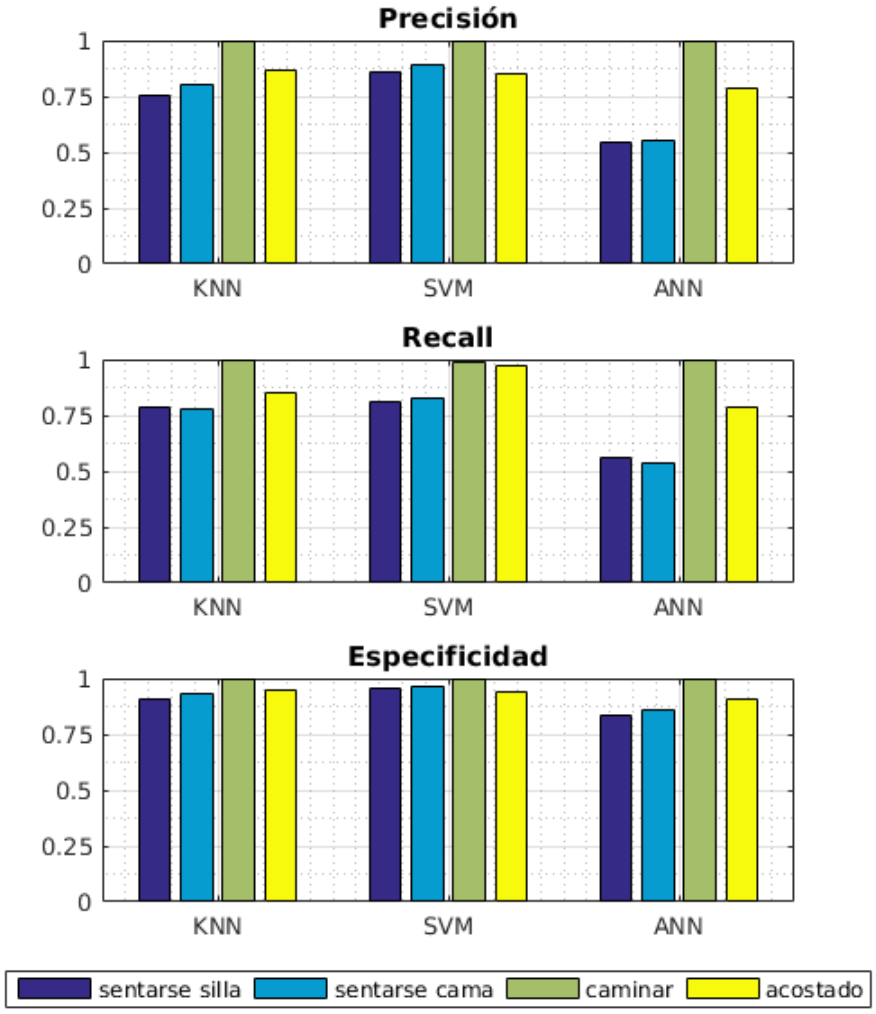


Figura 5.7: Comparación de modelos

Para cada modelo se analiza los porcentajes de las métricas correspondientes a cada clase, claramente se observa que la clase 3 (caminar) es fácilmente identificable entre las actividades consideradas en el problema con cualquier modelo empleado, refleja una tendencia al 100 % en cada una de las métricas.

El desempeño para las clase 1 y 2 (sentarse en la silla y sentarse en la cama respectivamente) independientemente del modelo tienden a ser aceptables aun cuando son actividades

semejantes. Para detallar cualitativamente el rendimiento del modelo acorde a los porcentajes de las métricas se genera la siguiente Tabla 5.4, donde excelente representa los porcentajes sobre 90 %, bueno a porcentajes sobre el 80 % y moderado a porcentajes sobre el 70 %.

Tabla 5.4: Análisis comparativo

		Rendimiento			
		Clase 1	Clase 2	Clase 3	Clase 4
KNN	Precisión	Moderado	Moderado	Excelente	Moderado
	Recall	Moderado	Moderado	Excelente	Bueno
	Especificidad	Bueno	Bueno	Excelente	Excelente
SVM	Precisión	Excelente	Excelente	Excelente	Excelente
	Recall	Excelente	Excelente	Excelente	Excelente
	Especificidad	Excelente	Excelente	Excelente	Excelente
ANN	Precisión	Bueno	Bueno	Excelente	Bueno
	Recall	Moderado	Moderado	Excelente	Bueno
	Especificidad	Bueno	Bueno	Excelente	Excelente

Del análisis comparativo se obtiene, el modelo SVM con el de mejor rendimiento, sin embargo se considera al costo computacional como otro parámetro de comparación que hace referencia al tiempo de ejecución para el entrenamiento detallado en la Tabla 5.5. Para su entrenamiento se utiliza un ordenador de las siguientes características Tabla 5.6.

Tabla 5.5: Análisis costo computacional

Modelo	KNN	SVM	ANN
Costo computacional	0.2	0.7	4.8

Tabla 5.6: Análisis costo computacional

	Procesador	Frecuencia	RAM	Disco duro
Ordenador	Core i7	2.4 GHz	12	1 TB

El clasificador SVM tiene mejor rendimiento pero requiere de mayor tiempo de entrenamiento, por lo tanto una correcta elección del modelo depende de la aplicación que se de al modelo de clasificación. Para aplicaciones donde se priorice mayor precisión, el modelo SVM es el mejor y cuando se priorice el menor tiempo de respuesta sin importar su precisión, el modelo KNN es el mejor.

Capítulo 6

Conclusiones y recomendaciones

Este capítulo presenta las conclusiones del presente trabajo y manifiesta algunas posibles sugerencias para su aplicación en trabajos futuros.

6.1. Conclusiones

1. El reconocimiento de actividades de personas por medio de aprendizaje de máquina es un problema de clasificación, que puede ser tratado con diferentes técnicas según la naturaleza de sus datos. Este trabajo se centró en técnicas que trabajan con señales de sensores tales como la segmentación, normalización y extracción de las características de los datos.

2. La literatura referente al entrenamiento de modelos de clasificación, utilizan en su gran mayoría los algoritmos como: máquina vectorial de soportes, redes neuronales artificiales y K-vecinos más cercanos.

3. Con la obtención, validación y evaluación del reconocimiento se obtuvo, que el clasificador con mejor rendimiento para este tipo de problemas es el modelo SVM, llegando a cumplir con el porcentaje de aceptabilidad de 90 % en su precisión.

4. El rendimiento del clasificador depende de la naturaleza de los datos, sin embargo su incremento se obtiene mediante la modificación de parámetros propios de cada algoritmo en el entrenamiento.

5. La aceptabilidad del modelo va acorde con la aplicación que se desee dar, generalmente se desarrolla aplicaciones donde se priorice los verdaderos positivos o caso contrario donde se priorice los falsos positivos.

6.2. Recomendaciones

1. En problemas de aprendizaje de máquina es indispensable realizar el estudio de la información disponible. Se sugiere que la etapa de obtención de los datos sea minuciosa y ordenada, evitando obtener datos erróneos o confusión entre distintas señales.

2. En la extracción de las características se debe centrarse en medidas estadísticas que realmente representen cada categoría. Evitar el uso de gran cantidad de características en el caso de no ser necesarias, ya que afectan en el tiempo de entrenamiento.

3. En la determinación del mejor modelo es adecuado realizar una serie de entrenamientos para cada algoritmo, modificando sus parámetros que afectan en el rendimiento. El mejor modelo será el que brinde menor tasa de error en su validación y prueba.

4. Para un trabajo enfocado en el análisis de rendimiento de algoritmos de clasificación, la mejor opción es usar base de datos ya estructuradas que por lo general son producto de resultados comprobados.

5. En la evaluación de los modelos siempre es necesario realizarlo con nuevos datos que sean totalmente independientes de los datos usados en el entrenamiento para evitar el sobreentrenamiento que reflejarán en resultados erróneos en la evaluación.

6.3. Trabajo futuro

En éste trabajo se ha realizado un análisis de rendimiento de algoritmos de aprendizaje de máquina en el reconocimiento de actividades de personas que servirá como fuente de información para la solución de problemas similares con aprendizaje de máquina.

Con la determinación del modelo de mejor rendimiento en el reconocimiento de actividades, brinda la apertura para que se formulen experimentos basados en asistencia médica, permitiendo ser capaces de realizar un diagnóstico basado en sus actividades diarias, e incluso abre la posibilidad de desarrollo de aplicaciones inteligentes que sean asistentes para el control de posibles actividades que pueda causar algún problema en cada persona.

Apéndice A

Listado de códigos

Este apéndice incluye los programas implementados en la ejecución de la identificación del sistema.

A.1. Programas implementados

A.1.1. Lectura y organización de la base de datos

códigos A.1: Lectura y organización de la base de datos

```
%%%%%%%%%%=====LECTURA DE LA BASE DE DATOS Y ORGANIZACIÓN  =====
clear
clc
close all
load('datosd2')
load('datosd1')

data_base=[data;data_base2];
xdata=data_base(1:end,2:4); % Señales aceleración x-y-z
ylabel=data_base(1:end,9); % Etiquetas

%Gráfica de las señales
figure(1)
plot(data_base(1:600,2:4));
title('SEÑAL');
xlabel ('Instancia');
ylabel('Magnitud')
legend ('acc x','acc y', 'acc z','etiqueta');
hold on
plot((data_base(1:600,9)));

set_data=[xdata,ylabel]; % Conjunto de señales etiquetadas
g=[1 2 3 4]; % Etiquetas para grupos a clasifiar


% Variables auxiliares para separar ordenar el conjunto de datos según la actividada realizada
```

```
m=length(ylabel);
incx=0;
incy2=1;

for inc = 1:4
       for da = 1:m
        if ylabel(da) == incy2
        incx=incx+1;
        set_orden(incx,1:4) = set_data(da,1:4); % Conjunto de datos ordenados
        end
       end
    incy2=incy2+1;
end


 % Separación en diferentes conjuntos de datos según la actividad
 insidencia=set_orden(1:end,4);
  [N0 Xn0]=hist(insidencia,g);
  limit1=N0(1);
  limit2=N0(1)+N0(2);
  limit3= limit2+N0(3);
  limit4=limit3+N0(4);

 % Actividades en grupos diferentes
 sit_on_bed=set_orden(1:N0(1),1:4);
 sit_on_chair=set_orden(N0(1)+1:limit2,1:4);
 lying=set_orden(limit2+1:limit3,1:4);
 ambulating=set_orden(limit3+1:limit4,1:4);
```

A.1.2. Extracción de las características

códigos A.2: Extracción de características

```
 %% Extracción de características
[ caract_class1] = caracter( sit_on_bed,1);
[ caract_class2] = caracter( sit_on_chair,2);
[ caract_class3] = caracter( lying,3);
[ caract_class4] = caracter( ambulating,4);

% Vector de características
vector_caract=[caract_class1;caract_class2;caract_class3;caract_class4];
```

códigos A.3: Función para generar las series de tiempo y calcular las medidas estadísticas

```
function [ caract_class ] = caracter(datos, clase)
% Extraer características
% Ingreso de datos y clase de pertenencia

 i=0;
for R = 10:1:2291
```

```
i=i+1;
d=R-9;
%== datos ==%%
s1M=datos([d:R],[1:3]);% Generar ventana de tiempo
s1s=mean(s1M); % Media
s1s3=min(s1M); %Mínimo
s1s2=max(s1M); %Máximo
s1s4=var(s1M); % Variación
s1s5=std(s1M); % Desviación Estandar
s1s6=iqr(s1M); % Cuartil

% Guardar en un vector
s1desv(i,[1:3])=s1s5;
s1med(i,[1:3])=s1s;
s1vari(i,[1:3])=s1s4;
s1cuart(i,[1:3])=s1s6;
s1mini(i,[1:3])=s1s3;
s1maxi(i,[1:3])=s1s2;
end
s1etcam(1:length(s1med),[1])=clase; % Etiquetado
caract_class=[s1med,s1maxi,s1mini,s1desv,s1vari,s1cuart,s1etcam]; % Vector final de caracterí
    sticas

end
```

A.1.3. Normalización

códigos A.4: Normalización

```
%%--------- Normalizar la señal ---------- %%
% Ecuación X_norm=(Xi-min_X)/(max_X-min_X)
for i=1:18
    for C = 1:length(vector_caract)
        min_vec=min(vector_caract(1:end,i));
        max_vec=max(vector_caract(1:end,i));
        rest(C,i)=vector_caract(C,i)-min_vec;
        X_norm(C,i)=rest(C,i)/(max_vec-min_vec);
    end
end

% Gráfica de la señal normalizada
figure(6)
subplot(2,1,1),
plot(vector_caract(1:916,[1:3]))
title('Características sin Normalizar')
legend('acc x','acc y', 'acc z')
subplot(2,1,2),
plot(X_norm(1:916,[1:3]))
title('Características Normalizadas')
legend('acc x','acc y', 'acc z')
```

A.1.4. División para entrenamiento y evaluación de los modelos

códigos A.5: Método hold out

```
%%%%%%%%%%División en grupos de entrenamiento y validación %%%%%%%%%%%%%%

p=0.3; % Porción para la división de entrenamiento y validación
[Train,Test]=crossvalind('HoldOut', vector_caract(1:end,19), p);

pv=0.3;
X=X_norm(1:end,1:18);
y=vector_caract(1:end,19);

SetTrainingSample=X(Train,:);
SetTrainingLabel=y(Train,1);
[TrainT,TrainV]=crossvalind('HoldOut',SetTrainingLabel,p);

% Datos para entrenamiento de los modelos
TrainingSample=SetTrainingSample(TrainT,:);
TrainingLabel=SetTrainingLabel(TrainT,1);

% Datos para validación
ValidationSample=SetTrainingSample(TrainV,:);
ValidationLabel=SetTrainingLabel(TrainV,1);

% Datos para evaluar los modelos, corresponde al 30% de las caracerísticas
TestingSample=X(Test,:);
TestingLabel=y(Test,1);
```

A.1.5. Modelos de clasificación y validación

códigos A.6: Modelos de clasificación

```
%%======= ALGORITMO  k-NN ======

% === Método validación cruzada  ==%

ind_knn = crossvalind('Kfold',y,4); % Validación cruzada
cp_cv = classperf(y);
for pro_cv=1:100
    for cv = 1:4

        % Selección de los grupos para entrenamiento y validación
        test_cv = (ind_knn == cv);
        train_cv = ~test_cv;

        % Generar modelo knn
        t=fitcknn(X(train_cv,:),y(train_cv,:),...
            'Distance','euclidean','NumNeighbors',pro_cv,'Standardize',1);

        % Validación
        label_cv = predict(t,X(test_cv,:));
        classperf (cp_cv,label_cv, test_cv);
```

```
        confcv=confusionmat(y(test_cv,:),label_cv);
    end
    error_cv(pro_cv)=cp_cv.ErrorRate % Tasa de error

    % Guardar modelo para K=10
    if pro_cv==10
        modelKNNcv=t;
    end
end

% === Método Hold-Out == %

cp = classperf(y);
for pro=1:100

    % Generar modelo knn
    t=fitcknn(TrainingSample,TrainingLabel,...
        'Distance','euclidean','NumNeighbors',pro,'Standardize',1);

    % Validación
    label1 = predict(t,ValidationSample);
    conf1=confusionmat(label1,ValidationLabel);

    %Cálculo de la tasa de error
    for xx = 1:4
        for yy = 1:4
            if(xx==yy)
                A1(xx) = conf1(xx,yy);
            end
        end
    end
    precisionknn(pro)=sum(A1)/length(ValidationLabel)*100;

    % Guardar modelo con k=10
    if pro==10
        modelKNNc=t;
    end
end

%Gráfica de la variación del error
error_ho=(100-precisionknn)
error_hoo=error_ho-1;
figure(20)
plot(error_cv*100)
title('Tasa de Error en función de K-Vecinos')
xlabel('k-vecinos')
ylabel('Error %')
hold on
plot(error_hoo)
legend('Validación cruzada','Hold-out')
grid on
grid minor


%----Plot matriz de confusión de validación para k=10 ---------%
validationknn=predict(modelKNNc,ValidationSample);
datadesc = ' clasificador KNN';
tst_data_binary.X = ValidationSample;
tst_data_binary.y = ValidationLabel;
data4class=tst_data_binary;
[error_rate,idx_error,match_table] = class_results(validationknn,data4class,datadesc);
fh = 12;
fh_last = visualise_results(idx_error,match_table,data4class,datadesc,fh,0);
```

```
%%=============== ALGORITMO SVM ================= %%
ks=0.1:0.1:2;
for sks=1:20;

    % Plantilla para el modelo
    fsvm= templateSVM('KernelFunction','gaussian',...
        'KernelScale',ks(sks),...
        'BoxConstraint', 1, ...
        'Standardize', true);

   % Generar modelo knn
    ecoc = fitcecoc(TrainingSample,TrainingLabel,...
        'Coding','onevsone',...
        'Learners',fsvm,...
        'ClassNames', [1; 2; 3; 4]);

    % Validación
    label3 = predict(ecoc,ValidationSample);
    conf2=confusionmat(ValidationLabel,label3)

        for xx = 1:4
            for yy = 1:4
                if(xx==yy)
                    A(xx) = conf2(xx,yy);
                end
            end
        end

    presicionsvm(sks)=sum(A)/length(ValidationLabel)*100

    % Guardar modelo para coeficiente de gauss igual a 1
    if sks== 2
      modelSVM=ecoc;
    end

end

% Gráfica  de la tasa de error
errorsvm=100-presicionsvm;
figure(25)
plot(errorsvm)
title('Tasa de error en función del coeficiente gauss')
xlabel('Coeficiente')
ylabel('Error %')
grid on
grid minor

%----Plot matriz de confusión de validación para gauss = 1 ---------%
validationsvm=predict(ecoc,ValidationSample);
datadesc = ' clasificador SVM';
tst_data_binary.X = ValidationSample;
tst_data_binary.y = ValidationLabel;
data4class=tst_data_binary;
[error_rate, idx_error, match_table] = class_results(validationsvm,data4class,datadesc);
fh = 13;
fh_last = visualise_results(idx_error,match_table,data4class,datadesc,fh,0);

%%=============== ALGORITMO RED NEURONAL ====================%%%%
% Obtención de la matriz de etiquetas con 0 y 1
mi= length(y);
for nn = 1:4
    a=0;
    for mm = 1:mi
```

```
        a=a+1;
        if y(a) == g(nn)
            mateti(nn,mm) =1;
        else
            mateti(nn,mm) =0;
        end
    end
end

matetiq=mateti'; % Matriz de etiquetas

% Conjunto de características
dat_ini=X_norm(1:end,1:18)';

% Definición topología de la red
red=patternnet([18 15 9 ],'trainscg');
% División las características
red.divideParam.trainRatio = 50/100; % Entrenamiento
red.divideParam.valRatio = 20/100; % Validación
red.divideParam.testRatio = 30/100;% Evaluación

red.trainParam.epochs=300; % número máximo de épocas
red.trainParam.showWindow=1; % mostrar ventana de entrenamiento
red.trainParam.goal=0.00005; % meta de performance
red.trainParam.mu=0.001;

% Entrenamiento
[red,entrenamiento]=train(red,dat_ini,mateti);
nntraintool
plotperform(entrenamiento)

% Validación del modelo
valX=dat_ini(:,entrenamiento.valInd);
valT=mateti(:,entrenamiento.valInd);
valY=red(valX);
valYIndices=vec2ind(valY);
valTIndices=vec2ind(valT);
[c cm ind per]=confusion(valT,valY);

% Gráfica de la matriz de confusión
figure(28)
plotconfusion(valT,valY);
```

A.1.6. Evaluación

códigos A.7: Evaluación

```
%% %%%%%%%%%%EVALUACIÓN ALGORITMOS %%%%%%%%%%%%%%%%%%%%%%
% =====  Evaluar KNN =====
tic
% Evaluación
evaluationknn=predict(modelKNNc,TestingSample);

% Gráfica de la matriz
datadesc = 'clasificador KNN';
tst_data_binary.X = TestingSample;
```

```
tst_data_binary.y = TestingLabel;
data4class=tst_data_binary;
[error_rate,idx_error,match_table] = class_results(evaluationknn,data4class,datadesc);
fh = 30;
fh_last = visualise_results(idx_error,match_table,data4class,datadesc,fh,0);

%Cálculo de la precisión, recall y especificidad
[ Pre_knn  Rec_knn Esp_knn] = resultados( match_table )
T1e=toc

% =====  Evaluar SVM =====
tic

% Evaluación
evaluationsvm=predict(ecoc,TestingSample);

%Gráfica de la matriz
datadesc = 'clasificador SVM';
tst_data_binary.X = TestingSample;
tst_data_binary.y = TestingLabel;
data4class=tst_data_binary;
[error_rate,idx_error,match_table] = class_results(evaluationsvm,data4class,datadesc);
fh = 35;
fh_last = visualise_results(idx_error,match_table,data4class,datadesc,fh,0);

%Cálculo de la precisión, recall
[ Pre_svm  Rec_svm Esp_svm] = resultados( match_table )
T2e=toc

% ======= Evaluar ANN ======
tic
% Evaluación
testX=dat_ini(:,entrenamiento.testInd);
testT=mateti(:,entrenamiento.testInd);
testY=red(testX);
testYIndices=vec2ind(testY);
testTIndices=vec2ind(testT);

%Gráfica de la matriz
[c cm]=confusion(testT,testY);
figure(40)
plotconfusion(testT,testY);

%Cálculo de la precisión, recall
[ Pre_ann  Rec_ann Esp_ann] = resultados(cm )
T3e=toc

%%
%Gráfica comparativa de los modelos
figure(45)
PREC= [Pre_knn;Pre_svm;Pre_ann];
RECALL=[Rec_knn;Rec_svm;Rec_ann];
ESP=[Esp_knn;Esp_svm;Esp_ann];
subplot(3,1,1), bar(PREC,.75,'grouped')
title('Precisión')
legend('sentarse silla','sentarse cama','caminar','acostado');
clases={'KNN','SVM','ANN'};
set(gca,'XTickLabel',clases );
grid on
grid minor

subplot(3,1,2), bar(RECALL,.75,'grouped')
title('Recall')
legend('sentarse silla','sentarse cama','caminar','acostado');
```

```
clases={'KNN','SVM','ANN'};
set(gca,'XTickLabel',clases );
grid on
grid minor

subplot(3,1,3), bar(ESP,.75,'grouped');
title('Especificidad')
legend('sentarse silla','sentarse cama','caminar','acostado');
clases={'KNN','SVM','ANN'};
set(gca,'XTickLabel',clases );
grid on
grid minor
```

Bibliografía

[1] A. García, *Inteligencia Artificial: Fundamentos, práctica y aplicaciones*. México: Alfaomega, 2013.

[2] G. P. Martinsanz and J. M. García, *Aprendizaje Automático*. Madrid, España: Ra - Ma, 2010.

[3] M. A. Mouriño, "Clasificicación multilingüe de documentos utilizando machine learning y la Wikipedia", Tesis doctoral, Escola Internacional de Doutoramento, Univ. de Vigo, España, 2017.

[4] A. Jahangiri & H. A. Rakha, "Applying Machine Learning Techniques to Transportation Mode Recognition Using Mobile Phone Sensor Data", IEEE Transactions on intelligent Transportation., vol. 16, pp. 2406-2417, Oct. 2015.

[5] A. Ledezma, "Reconocimiento de actividades: anticipando las necesidades del usuario", Departamento de informática, Universidad Carlos III de Madrid.2014.

[6] I. Aguilera, D. García, P. Morante & A. Pidal, "Reconocimiento de actividad mediante pulsera con sensores", Tesis de Grado, Facultad de Informática,Univ. Complutense de Madrid, España, 2017.

[7] O. Banos, J. M. Galvez, , M. Damas, H. Pomares, & I. Rojas, "Window size impact in human activity recognition", Sensors, 14(4), 6474-6499, 2014.

[8] A. Bayat, M. Pomplun & D. A. Tran, "A study on human activity recognition using accelerometer data from smartphones", Procedia Computer Science, 34, 450-457, 2014.

[9] AEFOL.(2016), Machine Learning: Un paso hacia la inteligencia artificia [En línea]. Disponible en: https://www.expoelearning.com/machine-learning-inteligencia-artificial/

[10] N. J. Mulcahy, & J. Call. "Apes save tools for future use", Science 312.5776 (2006): 1038-1040.

[11] L. Chen, *etal*.. "Sensor-based activity recognition", IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42.6 (2012): 790-808.

[12] N. Li, *etal*., "A Machine Learning Approach for Automatic Student Model Discovery," En Edm, pp. 31-40, 2011.

[13] M.Chan, *etal*., "A review of smart homes—Present state and future challenges," Computer methods and programs in biomedicine, vol. 91, no 1, p. 55-81, 2008.

[14] L. Piyathilaka, S. Kodagoda, "Human activity recognition for domestic robots", En Field and Service Robotics. Springer, Cham, p. 395-408, 2015.

[15] T. M. Mitchell. *Machine learning.* McGraw Hill series in computer science. McGraw-Hill, New York, NY, USA., 1997.

[16] D. Sánchez, "Algoritmos para la Clasificación Multinstancia", Tesis doctoral, Departamento de Ciencias de Computación e Inteligencia Artificial, Univ. de Granada, Granada, 2014.

[17] M. Reyes, "Adquisición de características de señales mioeléctricas para uso en protésica", Tesis Licenciatura en Ingeniería Mecatrónica, Departamento de Computación, Electrónica y Mecatrónica, Escuela de Ingeniería, Universidad de las Américas Puebla. Diciembre,2012.

[18] K. Englehart, B. Hudgins,"A robust, real-time control scheme for multifunction myoelectric control", IEEE transactions on biomedical engineering, 2003, vol. 50, no 7, p. 848-854.

[19] N. Thepvilojanapong, *etal*., "Recognizing bicycling states with hmm based on accelerometer and magnetometer data", En SICE Annual Conference (SICE), 2011 Proceedings of. IEEE, 2011. p. 831-832.

[20] C. Aggarwal, "Data mining: the textbook", Springer, 2015.

[21] Kelleher, D. John , M. N. Brian, and D. Aoife, "Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies", MIT Press, 2015.

[22] A. C. Contreras,"Introducción a Machine Learning", Sunqu, 2016.

[23] A. Jahangiri, *etal*., "Investigating cyclist violations at signal-controlled intersections using naturalistic cycling data", Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on. IEEE, 2016.

[24] Berrendero, "Extensiones y aplicaciones(Máquinas de vectores soporte, SVM)", 2019.

[25] J. Amat Rodrigo, "Máquinas de Vector Soporte (Support Vector Machines, SVMs)", Rstudio-pubs-static.s3.amazonaws.com, 2019. [Online]. Disponible en: https://rstudio-pubsstatic.s3.amazonaws.com/267926_04d64a3e96dc49b4ae6a2ed32fa29fe6.html. [Accedido: 17- Mar- 2019].

[26] D. Farina, R. Merletti, “Comparison of algorithms for estimation of EMG variables during voluntary isometric contractions”, J. Electromyogr. Kine-siol. 10 (2000) 337–349.

[27] J. Sánchez, “Análisis de técnicas Machine Learning para la estimación de medidas corporales”, TFG en Ingeniería Informática, Escuela de Ingeniería, Universidad Autónoma de Barcelona.

[28] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, “A practical guide to support vector classification,” National Taiwan Univ., Taipei, Taiwan, 2003.

[29] S. S. Keerthi and C.-J. Lin, “Asymptotic behaviors of support vector machines with Gaussian kernel,” Neuronal Comput., vol. 15, no. 7, pp. 1667–1689, Jul. 2003.

[30] T. Masters, “Practical neural network recipes in C++”, Morgan Kaufmann, 1993.

[31] P. Kim, “MATLAB Deep Learning: With Machine Learning, Neuronal Networks and Artificial Intelligence”, Apress, 2017.

[32] Archive.ics.uci.edu. (2016). UCI Machine Learning Repository: Activity recognition with healthy older people using a batteryless wearable sensor Data Set. [En línea] Disponible en: https://archive.ics.uci.edu/ml/datasets/Activity+recognition+with+healthy+older+people+using+a+batteryless+wearable+sensor [Accedido: 7 Jul. 2018].