

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO
GENÉTICO PARA REDUCCIÓN DE DIMENSIONALIDAD
PARA OPTIMIZAR UN CLASIFICADOR OBJETIVO”

NICOLÁS ALONSO ROMÁN VALLEJO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Nicolás Rojas Morales
Profesor Correferente: José Luis Martí

Septiembre - 2024

DEDICATORIA

Dedico esta memoria a mis padres, quienes me llevaron a aprender más y desarrollar mis habilidades y que cada logro era con su apoyo. A mis amigos por ser la alegría de ir a las clases y el apoyo mutuo.

AGRADECIMIENTOS

Quiero comenzar agradeciendo a mis padres por su apoyo incondicional y por creer en mis capacidades, incluso en los momentos más difíciles. Gracias por estar siempre ahí para recordarme que nada es imposible de superar, incluso cuando los obstáculos parecían insuperables.

A mis compañeros y amigos, con quienes he compartido este viaje desde el inicio de la carrera a través de Discord, gracias por su compañía constante y por el apoyo mutuo que nos hemos brindado. Espero que pronto todos completemos esta etapa y que sigamos manteniendo ese contacto de vez en cuando.

Agradezco especialmente a mi profesor guía, Nicolás Rojas, por su orientación y paciencia durante el desarrollo de mi memoria, incluso cuando me encontraba perdido con algunos conceptos. También, gracias por haber apoyado a mi hermano en momentos en los que él enfrentó dificultades.

Finalmente, a todos los profesores que me enseñaron no solo conocimientos técnicos, sino también valores profesionales y personales. Gracias por recordarme que la educación no es solo sobre materias, sino también sobre ser buenas personas y profesionales.

RESUMEN

Resumen—La maldición de la dimensionalidad es un fenómeno que se manifiesta como un crecimiento exponencial de la complejidad computacional y la dificultad de aprendizaje a medida que aumenta el número de características en los datos utilizados en el campo del machine learning. Este trabajo propone como alternativa la aplicación de un algoritmo genético (AG) como método de selección de características para la reducción de atributos en modelos de clasificación, con el fin de mejorar el rendimiento general de las predicciones. La implementación de este algoritmo se llevó a cabo en el contexto del proyecto Human Place, utilizando redes neuronales y random forest como modelos de clasificación. Como parte de la validación, el algoritmo se aplicó a un segundo caso de estudio, para los datos de human activity recognition. Los resultados de esta investigación mostraron que el AG mejoró el rendimiento de las redes neuronales en el primer caso y fue más efectivo para el clasificador support vector machine (SVM) en el segundo caso.

Palabras Clave— Machine learning, Algoritmo genético, Clasificación, Maldición de dimensionalidad, Selección de características.

ABSTRACT

Abstract—The curse of dimensionality is a phenomenon characterized by an exponential increase in computational complexity and learning difficulty as the number of features in the data used in machine learning grows. This work proposes the use of a genetic algorithm (GA) as a feature selection method for attribute reduction in classification models, aiming to improve the overall performance of predictions. The implementation of this algorithm was carried out within the context of the Human Place project, using neural networks and random forest as classification models. As part of the validation, the algorithm was applied to a second case study involving human activity recognition dataset. The results of this research showed that the GA improved the performance of neural networks in the first case and was more effective for the support vector machine (SVM) classifier in the second case.

Keywords— Machine learning, Genetic algorithm, Classification, Curse of dimensionality, Feature selection.

GLOSARIO

ACC: Exactitud (Accuracy). Métrica que mide el total de entradas que fueron clasificadas correctamente por un modelo.

AG: Algoritmo genético.

ERR: Tasa de error (Error Rate). Proporción de predicciones incorrectas respecto al total de predicciones realizadas.

FN: Falsos Negativos (False Negatives). Métrica que mide la cantidad de falsos negativos en la predicción de un conjunto de entradas.

FP: Falsos Positivos (False Positives). Métrica que mide la cantidad de falsos positivos en la predicción de un conjunto de entradas.

HAR: Human Activity Recognition.

KNN: K vecinos más próximos (K-nearest neighbors).

LDA: Análisis discriminante lineal (Linear Discriminant Analysis).

ML: Machine learning.

NN: Redes neuronales (Neural Networks).

NPV: Valor Predictivo Negativo (Negative Predictive Value). Métrica que mide la proporción de negativos predichos que son verdaderamente negativos.

PPV: Precisión (Positive Predictive Value). Métrica que mide la proporción de positivos predichos que son verdaderamente positivos.

RF: Bosque aleatorio (Random Forest).

RFE: Eliminación Recursiva de Características (Recursive Feature Elimination).

Recall: Métrica para evaluar la capacidad de un modelo de clasificar correctamente las entradas con presencia de una clase.

SVM: Máquinas de vectores de soporte (Support Vector Machine).

TN: Verdaderos Negativos (True Negatives). Métrica que mide la cantidad de verdaderos negativos en la predicción de un conjunto de entradas.

TP: Verdaderos Positivos (True Positives). Métrica que mide la cantidad de verdaderos positivos en la predicción de un conjunto de entradas.

TPR: Sensibilidad (True Positive Rate). Métrica que mide la proporción de verdaderos positivos detectados respecto al total de positivos reales.

TNR: Especificidad (True Negative Rate). Métrica que mide la proporción de verdaderos negativos detectados respecto al total de negativos reales.

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
GLOSARIO	V
ÍNDICE DE FIGURAS	VIII
ÍNDICE DE TABLAS	VIII
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	3
1.1 Contexto	3
1.2 Objetivo de solución	4
1.2.1 Objetivo general	4
1.2.2 Objetivos específicos	4
1.2.3 Alcance	4
CAPÍTULO 2: ESTADO DEL ARTE	6
2.1 Métricas	8
2.2 Dimensionalidad	11
2.3 Algoritmos genéticos	16
2.4 Configuración de parámetros	18
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN	21
3.1 Implementación - Caso Human Place	22
3.1.1 Representación:	22
3.1.2 Población inicial:	22
3.1.3 Función de Evaluación:	24
3.1.4 Proceso de selección:	24
3.1.5 Cruzamiento:	25
3.1.6 Mutación:	26
3.1.7 Reemplazo:	26
3.2 Implementación - Caso HAR	29
3.2.1 Representación:	29
3.2.2 Población inicial:	29
3.2.3 Función de Evaluación:	30
3.2.4 Proceso de selección:	30
3.2.5 Cruzamiento:	31
3.2.6 Mutación:	31
3.2.7 Reemplazo:	32

CAPÍTULO 4: RESULTADOS	34
4.1 Caso de estudio Human Place	34
4.1.1 Primera implementación - Redes Neuronales	36
4.1.2 Primera implementación - Random Forest	42
4.1.3 Tiempos de ejecución	45
4.2 Caso de estudio Human Activity Recognition	47
4.2.1 Primera implementación	49
4.2.2 Segunda implementación	55
4.2.3 Tiempos de ejecución	59
CAPÍTULO 5: CONCLUSIONES	61
REFERENCIAS BIBLIOGRÁFICAS	67

ÍNDICE DE FIGURAS

1.1	Árbol del problema.	4
2.1	Diagrama de flujo del marco generalizado de clasificación.	7
2.2	Proceso de selección de características	13
2.3	Proceso de selección de características	15
3.1	Ejemplo de selección por <i>roulette wheel</i>	25
3.2	Ejemplo de cruzamiento en un punto	26

ÍNDICE DE TABLAS

1	Matriz de confusión.	9
2	Métricas de rendimiento basadas en una matriz de confusión.	9
3	Extracto del dataset de <i>Breast Cancer detection Wisconsin</i> [William <i>et al.</i> , 1995]	12
4	Ejemplos de representación de atributos como cromosomas	22
5	Ejemplos de representación de atributos como cromosomas para el caso HAR	29
6	Valor de accuracy para las distintas instancias de datos sintéticos mediante Redes Neuronales y su promedio. Fuente: [Larrain <i>et al.</i> , 2022].	35
7	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: [Larrain <i>et al.</i> , 2022].	35
8	Valor de accuracy para las distintas instancias de datos sintéticos mediante Random Forest y su promedio. Fuente: [Larrain <i>et al.</i> , 2022].	35
9	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Random Forest. Fuente: Elaboración propia.	35
10	Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.	37
11	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.	37
12	Valor de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales para el mejor cromosoma por caso. Fuente: Elaboración propia.	38
13	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales para el mejor cromosoma por caso. Fuente: Elaboración propia.	38
14	Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.	39
15	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.	40
16	Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.	40

17	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.	41
18	Exactitud del mejor cromosoma para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.	42
19	Falsos negativos para los mejores cromosomas en las distintas instancias de datos sintéticos (Población 40, 20 Generaciones). Fuente: Elaboración propia.	42
20	Promedio de exactitud para las distintas instancias de datos sintéticos mediante <i>Random Forest</i> . Fuente: Elaboración propia.	44
21	Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante <i>Random Forest</i> . Fuente: Elaboración propia.	44
22	Promedio de exactitud para las distintas instancias de datos sintéticos mediante <i>Random Forest</i> para más generaciones y tamaño de población. Fuente: Elaboración propia.	44
23	Tasa de Falsos Negativos para más generaciones y tamaño de población. Fuente: Elaboración propia.	45
24	Tiempos de ejecución para NN, menor cantidad de generaciones y población. Fuente: Elaboración propia.	46
25	Tiempos de ejecución para NN, mayor cantidad de generaciones y población. Fuente: Elaboración propia.	46
26	Tiempos de ejecución para RF, 10 generaciones y 20 de población. Fuente: Elaboración propia.	46
27	Tiempos de ejecución para RF, 20 generaciones y 40 de población. Fuente: Elaboración propia.	47
28	Métrica de recall por clase de diferentes clasificadores con reconocimiento de actividad humana usando 6 atributos. Fuente: [Chen <i>et al.</i> , 2020].	48
29	Exactitud de clasificación de diferentes clasificadores con reconocimiento de actividad humana. Fuente: [Chen <i>et al.</i> , 2020].	48
30	Rendimiento del clasificador Naive Bayes utilizando el conjunto de datos HAR con características seleccionadas por K-means. Fuente: [Ismi <i>et al.</i> , 2016].	49
31	Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, primera implementación. Fuente: Elaboración propia.	50
32	Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, Fuente: Elaboración propia, ...	51
33	Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, primera implementación con mayor población y generaciones. Fuente: Elaboración propia.	52
34	Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, primera implementación con mayor población y generaciones. Fuente: Elaboración propia.	55
35	Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, segunda implementación. Fuente: Elaboración propia.	56

36	Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, segunda implementación. Fuente: Elaboración propia.	57
37	Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, segunda implementación, mayor población y generaciones. Fuente: Elaboración propia.	58
38	Tiempos de ejecución para el caso HAR. Fuente: Elaboración propia.	59
39	Tiempos de ejecución para el caso HAR, Naive Bayes. Fuente: Elaboración propia.	60

INTRODUCCIÓN

Para el campo de *machine learning*, el aumento de la dimensionalidad en los datos presenta complicaciones al momento de realizar las tareas de los modelos utilizados, generando consecuencias en los tiempos de computación y en la redundancia e irrelevancia de ciertas características que poseen los datos. Bajo esta premisa se propone la aplicación de un modelo de selección de características a través de un algoritmo genético, el cual se inspira en la idea de la variación genética y la selección natural dentro de una población, donde el más fuerte, para este caso, aquellos con mayores aptitudes, son quienes sobreviven e incrementan el rendimiento de los modelos. Esta memoria se presenta como una manera de mejorar el rendimiento de los clasificadores utilizados por el proyecto Human Place de salud mental para los estudiantes chilenos.

Inicialmente, se plantea el problema a tratar, donde se menciona la maldición de dimensionalidad, el cual ha incrementado en la actualidad debido a la cantidad de datos que se obtienen en esta era de la tecnología. Este fenómeno se expresa como el crecimiento exponencial en la complejidad de los datos, a medida que incrementa la dimensión. Al observar el comportamiento de distintos clasificadores, como SVM o RF, se aplican algoritmos de reducción de dimensionalidad, como puede ser PCA o LDA, los cuales buscan obtener la información más importante de este conjunto de datos para reducir la cantidad de atributos con las que tienen que trabajar estos clasificadores. Entonces, se propone la utilización de un algoritmo genético como método de reducción de dimensionalidad por sus características de exploración y el cómo se pueden alterar los parámetros de este algoritmo para adaptarse a distintos contextos.

En el segundo capítulo, se presentan investigaciones que aplicaron métodos de selección de características para mejorar la clasificación, como la categorización de frutas según características visuales y la clasificación binaria en diagnósticos médicos. Además, se exploran métodos de clasificación multietiqueta para asignar múltiples etiquetas a entidades. Para evaluar el rendimiento de los modelos, se introducen métricas clave como la exactitud, la precisión, el recall y la especificidad, así como métricas adicionales para problemas multietiqueta, tales como la Hamming loss y la exactitud de clasificación. Estas métricas permiten un análisis detallado de la efectividad de los modelos en la clasificación y etiquetado de datos.

La dimensionalidad en *machine learning* se refiere al número de atributos en un conjunto de datos, y el aumento exponencial de estos atributos puede complicar el procesamiento de estos. La maldición de dimensionalidad plantea retos significativos como la sobrecarga computacional y la dificultad en la extracción de información relevante. Para abordar estos desafíos, se emplean técnicas de reducción de dimensionalidad, que incluyen tanto la selección de características, que filtra y elimina atributos irrelevantes, o la extracción de características, que genera nuevas características a partir de los datos originales. Estos enfoques permiten manejar grandes volúmenes de datos de manera más eficaz y mejorar el rendimiento de los modelos al reducir la complejidad y redundancia.

En este contexto, los algoritmos genéticos surgen como una herramienta potente para la optimización, inspirándose en la variación genética y la selección natural. Los AG utilizan una población de cromosomas para aplicar operadores genéticos como el cruzamiento y la mutación, simulando la adaptación natural para encontrar soluciones óptimas. El proceso incluye la inicialización de la población, la evaluación de aptitud, la selección de los cromosomas más aptos, la recombinación de estos y la mutación para mantener la diversidad. Sin embargo, los AG pueden enfrentar problemas de convergencia y la necesidad de ajustar parámetros como la probabilidad de mutación para evitar que se asemeje a una búsqueda aleatoria.

La configuración de parámetros es crucial en los AG, ya que permite una búsqueda rápida y eficiente de combinaciones que mejoran el rendimiento. Estos algoritmos pueden encontrar soluciones óptimas o casi óptimas en espacios de búsqueda grandes, aunque son sensibles a los cambios en sus parámetros. Por ejemplo, en estudios recientes, los AG han demostrado mejorar el rendimiento de clasificadores y reducir la dimensionalidad en datos complejos, destacándose en la optimización de características y la mejora de la precisión del modelo.

En el capítulo 3 se presenta la propuesta de solución, detallando la implementación del algoritmo genético para la selección de características en cada caso. Para Human Place, se selecciona un número aleatorio de características, manteniendo aquellas que deben presentarse siempre según las indicaciones del proyecto. Esto genera una población con diferentes aptitudes en sus cromosomas, y luego se evalúan estos cromosomas a lo largo de un número determinado de generaciones, conservando el mejor caso para identificar las características más relevantes para un mejor rendimiento. En el segundo caso, dado que se requería un número específico de características, se genera una población considerando esta restricción y se aplica el procedimiento del algoritmo genético previamente descrito.

En el capítulo 4 se presentan los resultados, demostrando que la aplicación del algoritmo genético como método para reducir la dimensionalidad fue efectiva en una variedad de clasificadores, incluyendo redes neuronales para el caso Human Place y SVM o LDA para el segundo caso. El capítulo 5 ofrece un resumen de los logros de esta memoria, junto con reflexiones sobre el análisis futuro y posibles formas de mejorar el algoritmo en sus parámetros, sugiriendo así posibles trabajos futuros.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

1.1. Contexto

En la actualidad, el incremento en la dimensionalidad de los datos ha generado desafíos significativos en el campo de la minería de datos y el aprendizaje automático, o *machine learning* (ML) por sus siglas en inglés. A medida que los conjuntos de datos adquieren dimensiones más elevadas, surgen problemas como la redundancia de atributos y la irrelevancia de ciertas características. Estos desafíos no solo aumentan la carga computacional requerida para las tareas de ML [Zebari *et al.*, 2020], sino que también se relacionan con un fenómeno conocido como la “maldición de la dimensionalidad”, que se manifiesta como un crecimiento exponencial en la complejidad a medida que la dimensión aumenta [Verleysen y François, 2005].

Un clasificador es un algoritmo que asigna un conjunto de características de entrada a categorías o clases predefinidas [Jain *et al.*, 2000]. Para abordar los desafíos de redundancia de atributos y la irrelevancia de ciertas características en los clasificadores, se emplean algoritmos de reducción de dimensionalidad, como *Principal Component Analysis* (PCA) o *Linear Discriminant Analysis* (LDA), que ayudan a capturar la información relevante de los datos originales de alta dimensión. Además, existen otras técnicas, como ISOMAP y Autoencoders [Huang *et al.*, 2018], que también son opciones para reducir la dimensionalidad de los datos.

Los Algoritmos Genéticos se utilizan para la selección de características, ya que son eficaces en la exploración de soluciones diversas y evitan quedar atrapados en óptimos locales. Estos algoritmos son robustos y pueden aplicarse a diferentes problemas [Sivanandam *et al.*, 2008].

Los pasos de los Algoritmos Genéticos incluyen la inicialización de una población, la evaluación de la aptitud de los individuos, la aplicación de operadores genéticos como selección, cruzamiento y mutación, y la preparación de la población para la siguiente generación. Se verifica si se ha alcanzado una solución óptima o se cumplen los criterios de parada [Alhijawi y Awajan, 2023].

El Árbol del Problema en la Figura 1 indica como problema principal el bajo rendimiento de los clasificadores en contextos de alta dimensionalidad. Este problema es causado por relaciones espurias y datos complejos que contienen una alta cantidad de atributos. Como consecuencia, esto lleva al sobreajuste dificultando la interpretación, la dispersión de datos, ralentiza el proceso de clasificación y puede introducir complejidad innecesaria. Abordar estas causas es esencial para mejorar el rendimiento de los clasificadores en datos de alta dimensionalidad.

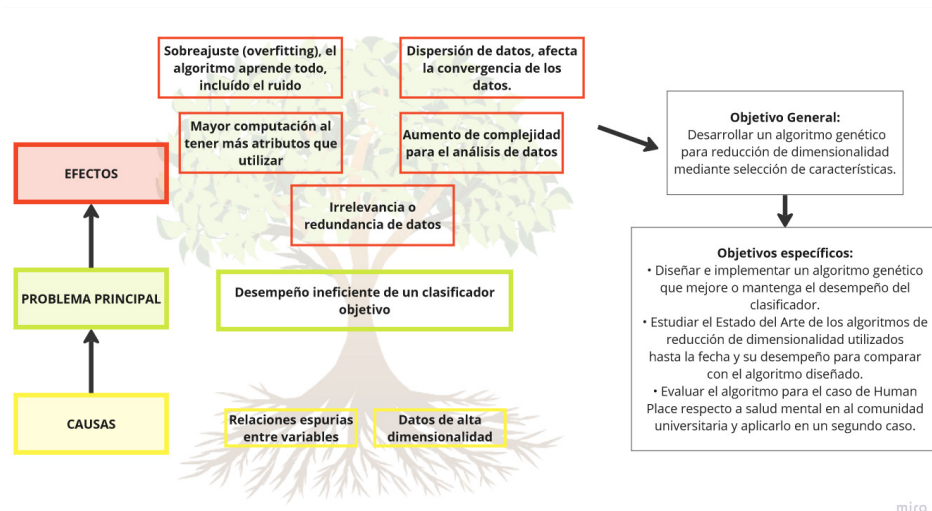


Figura 1.1: Árbol del problema.
Fuente: Elaboración propia.

1.2. Objetivo de solución

1.2.1. Objetivo general

El objetivo general de este tema de memoria es el desarrollo de un Algoritmo Genético para reducción de dimensionalidad mediante selección de características.

1.2.2. Objetivos específicos

1. Estudiar el Estado del Arte de los algoritmos de reducción de dimensionalidad utilizados hasta la fecha.
2. Diseñar e implementar un Algoritmo Genético que mejore o mantenga el desempeño del clasificador.
3. Evaluar el algoritmo para el caso de Human Place respecto a salud mental en la comunidad universitaria y aplicarlo en un segundo caso existente en la literatura.

1.2.3. Alcance

El algoritmo genético a diseñar e implementar busca poder aplicarse al caso del trabajo de Gonzalo Larrain B. [Larrain *et al.*, 2022], donde desarrollaron una red neuronal que predice

4 posibles perturbaciones en las personas que sufren estrés causado por estudios o trabajo, para así mejorar el rendimiento del clasificador.

Posteriormente, se aplicará a un segundo caso de reconocimiento de actividades humanas de la literatura para determinar si se mantienen o no las métricas de calidad medidas entre distintos modelos y el algoritmo genético. La principal métrica a mejorar o mantener es la *accuracy* o exactitud. Además, se compararán los verdaderos positivos y falsos negativos siempre que sea factible para una evaluación más detallada.

CAPÍTULO 2

ESTADO DEL ARTE

Un problema de clasificación se refiere al proceso de asignar entidades a clases que inicialmente no están definidas, de modo que los individuos en una misma clase estén de alguna manera cercanos entre sí. Esto implica la elección de a cuál de las diversas clases definidas se debe asignar una nueva entidad [Cormack, 1971].

Un ejemplo de este problema es el abordado por Sumaira Ghazal et al., en su estudio *Analysis of visual features and classifiers for Fruit classification problem*. En esta investigación, se exploró la clasificación de frutas utilizando técnicas de aprendizaje automático basadas en características visuales. Mediante el análisis de atributos como el color, la textura y la forma, lograron extraer características que permitieron identificar el tipo de fruta representada en cada imagen, como manzanas de diferentes variedades: dorada, braeburn o roja.

En el artículo anterior, se presenta un diagrama de flujo que ilustra el proceso de clasificación (consulte la Figura 2.1). Este proceso se divide en tres etapas fundamentales.

En la primera etapa, se prepara la imagen de entrada para enfocarse en la región de interés, eliminando detalles innecesarios. Para lograr esto, se aplican técnicas de segmentación de imágenes, como la conversión a escala de grises, lo que permite evaluar la intensidad de los colores. Esta fase busca normalizar los datos de entrada.

En el siguiente paso, se identifican y extraen características relevantes de la región de la fruta. Estas características pueden incluir aspectos relacionados con el color, textura y forma. El objetivo es identificar lo que hace que cada fruta sea única. En este caso, los autores proponen una combinación de características, que es efectiva para clasificar las frutas.

Finalmente, se utilizan las características extraídas para asignar etiquetas a las frutas. Se aplican diferentes clasificadores, como *Support Vector Machine* (SVM), *K-Nearest Neighbor* KNN y Random Forest, para determinar la etiqueta más adecuada para cada fruta. Esta etapa permite evaluar cuál de los clasificadores logra el mejor rendimiento en función de los datos reales.

Este enfoque proporciona una visión general de cómo se aborda un problema de clasificación, desde la preparación de los datos hasta la identificación de características y la asignación de etiquetas, considerando la fase de entrenamiento del modelo, donde este se ajusta al conjunto de datos para realizar la clasificación y la fase de prueba, donde se evalúa el rendimiento del modelo a través de un conjunto de datos determinado para las pruebas [Ghazal et al., 2021].

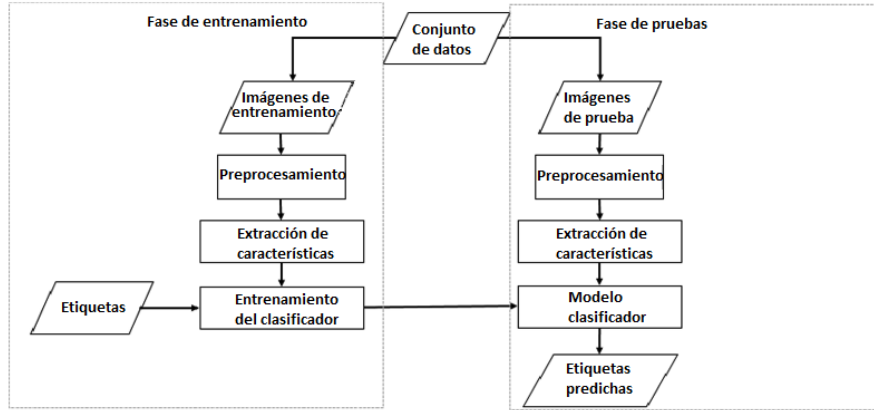


Figura 2.1: Diagrama de flujo del marco generalizado de clasificación.

Fuente: [Ghazal *et al.*, 2021].

Problemas de clasificación binaria Otra de las variantes al problema tratado anteriormente, es el problema de clasificación binaria, donde la tarea es, en un conjunto de datos, las entidades son asociadas a una de las dos clases existentes. Ejemplos de la vida real de este problema es son los diagnósticos médicos, para determinar si el paciente presenta o no alguna condición o enfermedad en particular [Jeatrakul y Wong, 2009].

El problema de clasificación binaria se define de la siguiente manera. Se tiene que $X = R^d$ es el espacio de características de entrada y $L = \{0, 1\}$ es el conjunto de dos posibles etiquetas. Dado un conjunto de entrenamiento $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ donde $x_i \in X$ y $y_i \in L$, se busca aprender una función $h : X \rightarrow L$ a partir de D capaz de predecir la etiqueta de una instancia no visitada [Gonçalves *et al.*, 2013].

Problemas de clasificación multietiquetas Realizando cambios a la expresión anterior, con $L > 2$, el problema se torna en una clasificación multietiquetas, donde se busca asociar con múltiples etiquetas de manera simultánea a las distintas entidades. Donde ejemplos de este problema es la categorización de texto o la categorización musical, detallando más esta última, una banda como Scorpions puede catalogarse como rock y balada [Tsoumakas *et al.*, 2006].

Utilizando misma formulación del problema binario, en la clasificación multietiquetas se tiene que $X = R^d$ es el espacio de características de entrada y $L = \{l_1, \dots, l_q\}$ un conjunto de q posibles etiquetas. Dado un conjunto de entrenamiento $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\} \{x_i \in X, y_i \subseteq L\}$, se busca aprender una función $h : X \rightarrow 2^L$ a partir de D capaz de predecir el conjunto de etiquetas de una instancia no visitada [Gonçalves *et al.*, 2013].

Técnicas para Problemas de Clasificación Existen dos enfoques principales para abordar el problema de clasificación multietiqueta, métodos dependientes del algoritmo y métodos independientes del algoritmo. Los métodos dependientes modifican o adaptan el algoritmo de clasificación de una sola etiqueta para manejar problemas de múltiples etiquetas, por ejemplo, esto puede darse en árboles de decisión, KNN [Zhang y Zhou, 2007] y redes bayesianas [Bielza *et al.*, 2011].

Por otra parte, los métodos independientes realizan la transformación del problema multietiqueta a uno o varios problemas de clasificación binaria, permitiendo así el uso de bosques aleatorios o SVM [Gonçalves *et al.*, 2013].

Las salidas de los modelos de clasificación pueden ser continuos si se utiliza, por ejemplo, el clasificador bayesiano ingenuo o continuo en un árbol de decisión. Pero estos resultados deben ser analizados posteriormente para obtener una interpretación correcta de los datos [Tharwat, 2020].

Los árboles de decisión son una representación gráfica de un modelo de toma de decisiones en forma de árbol, donde cada nodo interno representa una característica (atributo), cada rama representa el resultado de una prueba y cada hoja representa un resultado de clasificación o decisión final. Las hojas en un árbol de decisión son los nodos terminales que no se dividen más y representan las predicciones o decisiones finales del modelo para una instancia dada [Quinlan, 1990].

Una de las formas de utilizar el árbol de decisión para afrontar salidas continuas es proporcionando probabilidades de pertenencia a una clase en lugar de decisiones categóricas. Un ejemplo de esto es aplicar probabilidades de hoja, donde en lugar de tomar una decisión categórica en un nodo hoja, se asignan probabilidades a cada clase según la distribución de las instancias de entrenamiento que llegan a ese nodo, permitiendo capturar la incertidumbre inherente en la clasificación de instancias. Otro método es la probabilidad por ruta, en el que se recorre cada nodo desde la raíz del árbol y se multiplican las probabilidades asociadas a cada rama para obtener la probabilidad de que la instancia alcance un nodo hoja específico, considerando así la probabilidad acumulativa de todas las decisiones tomadas a lo largo de la ruta desde la raíz hasta el nodo hoja [Quinlan, 1990].

2.1. Métricas

En la literatura, existen diferentes métricas para evaluar el rendimiento de modelos de *machine learning* aplicados en problemas de clasificación. Aunque estas fueron propuestas para problemas de clasificación binaria, algunas de estas métricas pueden ser generalizadas para más de dos clases [Tharwat, 2020].

Tanto para problemas de clasificación binaria como problemas de clasificación de más de dos clases, el rendimiento de los clasificadores viene dado por la *confusion matrix* o matriz

de confusión, la cual es el medio principal para evaluar errores en este tipo de problemas. Esta matriz compara las clases obtenidas por la predicción y las observadas, comparando si estas fueron clasificadas correctamente y cuántas incorrectamente, inspeccionando así errores en cada clase [Beauxis-Aussalet y Hardman, 2014]. Una representación de la matriz de confusión se ilustra en la Tabla 1.

Tabla 1: Matriz de confusión.

Fuente: [Room, 2019].

		Valor actual	
Valor previsto		Positivos	Negativos
	Positivos	TP	FP
	Negativos	FN	TN

Las métricas clave derivadas de la matriz de confusión incluyen los *true positives* (TP), que representan la proporción de casos positivos correctamente identificados; los *false positives* (FP), que son la proporción de casos negativos incorrectamente clasificados como positivos; los *true negatives* (TN), que indican la proporción de casos negativos correctamente clasificados; y los *false negatives* (FN), que son la proporción de casos positivos incorrectamente clasificados como negativos [Santra y Christy, 2012].

A partir de la matriz de confusión, se pueden obtener las siguientes métricas:

Tabla 2: Métricas de rendimiento basadas en una matriz de confusión.

Fuente: [De Diego *et al.*, 2022].

Símbolo	Métrica	Fórmula
ACC	Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
ERR	Error	$\frac{FP+FN}{TP+TN+FP+FN}$
PPV	Precision	$\frac{TP}{TP+FP}$
TPR	Recall	$\frac{TP}{TP+FN}$
TNR	Specificity	$\frac{TN}{TN+FP}$

Definiendo cada métrica, ACC, o *accuracy*, corresponde a la proporción de instancias correctamente predichas entre todas las instancias en el conjunto de datos. El error es la proporción de predicciones incorrectas realizadas por el modelo de clasificación. La precisión, o PPV, indica la proporción de predicciones verdaderamente positivas entre todas las predicciones positivas realizadas por el modelo. TNR se refiere a las predicciones verdaderamente negativas entre todas las instancias negativas reales en el conjunto de datos. TPR, o *recall*, corresponde a la proporción de predicciones verdaderamente positivas entre todas las instancias positivas reales en el conjunto de datos [De Diego *et al.*, 2022].

Para problemas multietiquetas, existen dos grupos de métricas de evaluación, las métricas basadas en ejemplos y métricas basadas en etiquetas.

Para este primer grupo, S corresponderá a un conjunto de datos de evaluación de múltiples etiquetas, que consiste de $|S|$ ejemplos multietiquetas $(x_i, Y_i), i = 1..|S|, Y_i \subseteq L$. [Zhang y Zhou, 2013]. En la clasificación multietiqueta, cada ejemplo x_i representa una entidad, como una canción tomando el caso de la banda Scorpions, que puede estar asociada con un conjunto de etiquetas Y_i . Aquí, Y_i sería el conjunto de géneros musicales a los que pertenece la canción x_i , siendo un subconjunto del total de etiquetas L . Estos géneros podrían ser rock, balada o metal. Por ejemplo, algunas de estas canciones podrían ser categorizadas simultáneamente como rock y baladas, reflejando así la naturaleza multietiqueta de la clasificación [Tsoumakas et al., 2006].

Una de las métricas que se puede obtener es *Hamming loss* que cuenta el número de etiquetas para las cuales nuestra predicción es incorrecta y la normaliza entre valores de 0 a 1. Sea h un clasificador multietiqueta y $Z_i = h(x_i)$ el conjunto de etiquetas predicho por h para el ejemplo x_i , la ecuación de esta pérdida corresponderá a:

$$\text{Hamming loss}(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{|Y_i \Delta Z_i|}{|L|} \quad (1)$$

donde Δ denota la diferencia simétrica entre dos conjuntos. En este caso, al número de etiquetas que están en Y_i pero no en Z_i (y viceversa) para así calcular las etiquetas que difieren entre el conjunto de etiquetas verdaderas y el conjunto de etiquetas predichas.

Esta métrica entregará un valor más bajo cuando el clasificador refleje un mejor rendimiento, puesto que las predicciones del clasificador estarán más cercanas a las etiquetas verdaderas [Destercke, 2014].

Una métrica más estricta, donde las etiquetas predichas deben coincidir exactamente con las etiquetas reales para entregar un valor positivo, es la Exactitud de Clasificación o *Classification Accuracy*, la cual viene dada por:

$$\text{Exactitud de Clasificación}(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} I(Z_i = Y_i) \quad (2)$$

donde $I(\text{true}) = 1$ para el caso positivo y $I(\text{false}) = 0$ para el negativo [Zhu et al., 2005]. Las últimas métricas de la categoría basada en ejemplos son las siguientes [Godbole y Sarawagi, 2004]:

$$\text{Precisión}(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{|Y_i \cap Z_i|}{|Z_i|} \quad (3)$$

$$\text{Sensibilidad}(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{|Y_i \cap Z_i|}{|Y_i|} \quad (4)$$

$$F1 - score(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{2|Y_i \cap Z_i|}{|Z_i| + |Y_i|} \quad (5)$$

$$Exactitud(h, S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \quad (6)$$

donde la precisión, exactitud y sensibilidad mantienen su significado como en los problemas de clasificación binarios. Mientras que (5) corresponde al *F1-score* el cual es un indicador unidimensional usado para medir el equilibrio entre la precisión y la sensibilidad en la evaluación del rendimiento de un sistema, entregando un valor entre 0 y 1, donde 1 representa un rendimiento perfecto y 0 un pobre rendimiento en la clasificación [Huang *et al.*, 2015].

Para las métricas basadas en etiquetas, se puede hacer uso de cualquier métrica usada en la evaluación binaria como la exactitud, precisión o sensibilidad para calcular estas métricas para todas las etiquetas se deben obtener dos operaciones de promedio, macropromedio (*macro-averaging*) y micropromedio (*micro-averaging*) [Zhang y Zhou, 2013].

$$M_{\text{macro}} = 1 - \frac{1}{|L|} \sum_{\lambda=1}^L M(TP_{\lambda}, FP_{\lambda}, TN_{\lambda}, FN_{\lambda}) \quad (7)$$

$$M_{\text{micro}} = M \left(\frac{1}{|L|} \sum_{\lambda=1}^L TP_{\lambda}, \frac{1}{|L|} \sum_{\lambda=1}^L FP_{\lambda}, \frac{1}{|L|} \sum_{\lambda=1}^L TN_{\lambda}, \frac{1}{|L|} \sum_{\lambda=1}^L FN_{\lambda} \right) \quad (8)$$

Las dos ecuaciones anteriores hacen uso de las métricas binarias de verdaderos positivos y negativos y falsos positivos y negativos, para posteriormente realizar las operaciones de promediado con las ecuaciones (7) y (8).

Macro-averaging toma el promedio ponderado de M, siendo M el tipo de evaluación binaria utilizada, para todas las etiquetas según su tamaño, calculando el promedio para cada etiqueta y luego promediar esos promedios, donde el peso es el tamaño de cada etiqueta.

Para *micro-averaging*, se suman todas las métricas binarias TP, FP, TN y FN para todas las etiquetas y se calcula M usando esas sumas [Koyejo *et al.*, 2015].

2.2. Dimensionalidad

La dimensionalidad en *machine learning* hace referencia a la cantidad de atributos o características que posee un conjunto de datos. En el contexto actual de la explosión de datos en diversas áreas de investigación, como el manejo de bases de datos, minería de datos, análisis de redes sociales y aprendizaje automático, se ha vuelto evidente que el crecimiento

de los algoritmos no puede mantenerse al ritmo del aumento de los datos. Este desajuste lleva a que muchos algoritmos se vuelvan computacionalmente intratables y, por ende, inviables en la práctica [Gao *et al.*, 2017]. En particular, los datos de alta dimensionalidad, que constituyen una forma de “Big Data”, presentan desafíos adicionales, como la redundancia de atributos o características y la falta de relevancia de algunas de ellas en la extracción de información. Este problema de la “maldición de la dimensionalidad” se ha convertido en un obstáculo central a superar en la búsqueda de soluciones efectivas para la selección de características y la reducción de la dimensionalidad [Zebari *et al.*, 2020].

En la Tabla 3, se muestra un extracto del dataset utilizado para la detección de cáncer de mama (*Breast Cancer detection Wisconsin*). Cada columna de la tabla representa un atributo o característica del conjunto de datos, que en total comprende 31 atributos. Estos atributos son cruciales para la clasificación y diagnóstico en estudios de aprendizaje automático aplicados a la detección temprana de cáncer de mama.

Tabla 3: Extracto del dataset de *Breast Cancer detection Wisconsin*[William *et al.*, 1995]

id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean
1	M	17.99	10.38	122.8	1001	0.1184	0.2776	0.3001	0.1471	0.2419
2	M	20.57	17.77	132.9	1326	0.08474	0.07864	0.0869	0.07017	0.1812
3	M	19.69	21.25	130	1203	0.1096	0.1599	0.1974	0.1279	0.2069
4	M	11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414	0.1052	0.2597
5	M	20.29	14.34	135.1	1297	0.1003	0.1328	0.198	0.1043	0.1809
6	M	12.45	15.7	82.57	477.1	0.1278	0.17	0.1578	0.08089	0.2087
7	M	18.25	19.98	119.6	1040	0.09463	0.109	0.1127	0.074	0.1794
8	M	13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366	0.05985	0.2196
9	M	13	21.82	87.5	519.8	0.1273	0.1932	0.1859	0.09353	0.235

La maldición de la dimensionalidad, originalmente mencionada por Richard Bellman al referirse al problema que se encuentra al optimizar y analizar en espacios de alta dimensionalidad, se manifiesta en el hecho de que, a medida que la dimensión de un espacio aumenta, la cantidad de datos o combinaciones posibles crece exponencialmente, lo que dificulta la obtención de datos suficientes y la diferenciación de puntos en el espacio [Bellman, 1957]. Estos desafíos han llevado al desarrollo de enfoques y algoritmos específicos para abordar la alta dimensionalidad, que se explorarán en las secciones posteriores de esta memoria.

Este problema se ha tratado en la literatura a través de dos tipos de técnicas, la selección de características y la extracción de características.

Selección de características

Esta primera agrupación de técnicas puede utilizarse para la reducción de dimensionalidad, realizando la selección de aquellos atributos más importantes y relevantes del conjunto de datos y, a la vez, eliminando aquellos considerados redundantes e irrelevantes. Estas técnicas ofrecen diversas ventajas, como reducir el tamaño de los datos a trabajar, disminuir el almacenamiento que se requiere, mejorar la precisión de las predicciones, evitar el

sobreajuste y reducir el tiempo de ejecución y entrenamiento del modelo de aprendizaje automático. El problema que se puede presentar es la posible pérdida de información, ya que algunas características deben ser excluidas en el proceso de elección del subconjunto de características [Gao et al., 2017].

En general, este tipo de algoritmos tienen dos fases, la generación de subconjuntos, donde se deben crear estos a partir de las características del conjunto de datos y determinar. El siguiente paso es la evaluación de los subconjuntos, donde se debe verificar si el subconjunto generado es de mejor calidad o no. La Figura 2.2 visualiza el cómo es el flujo de este proceso. Estas operaciones se pueden realizar de cinco formas; filtrado (*filter*), incrustado (*embedded*), envoltura (*wrapped*), híbrido (*hybrid*) o conjunto (*ensemble*).

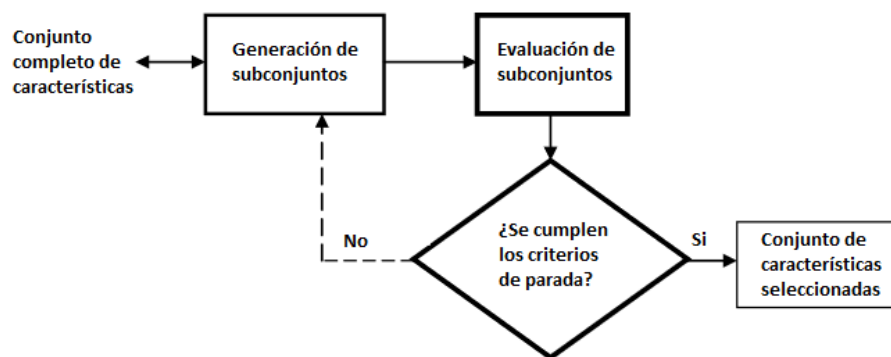


Figura 2.2: Proceso de selección de características
Fuente: [S Velliangiri y Thankumar, 2019].

Los métodos de filtrado son eficientes desde una perspectiva computacional y son independientes del clasificador. Estos métodos evalúan la relevancia de las características en función de sus propias estadísticas, como la correlación para posteriormente seleccionar las características más relevantes.

Por otro lado, los métodos de incrustado utilizan algoritmos de aprendizaje específicos para evaluar la calidad de los subconjuntos de características. Realizan una búsqueda en el espacio de subconjuntos y evalúan cada uno de ellos utilizando este algoritmo. Esta búsqueda puede basarse en estrategias que añaden progresivamente las características más relevantes o eliminan aquellas menos significativas, evaluando en cada paso cómo cada característica afecta al modelo existente. Aunque suelen ser más costosos en términos computacionales, proporcionan resultados más precisos que los métodos de filtrado.

En el caso de los métodos de envoltura, la selección de características se integra como parte del proceso de construcción del modelo. Estos métodos utilizan algoritmos de aprendizaje específicos y pueden realizar la selección durante el entrenamiento, utilizando por ejemplo árboles de decisión [Padmaja y Vishnuvardhan, 2016].

Para los métodos híbridos, se combinan tanto los métodos de filtro como los de envoltura, donde el primer paso es usar los de filtro para descartar la mayoría de las características irrelevantes del conjunto inicial y luego aplicar el de envoltura para seleccionar las características del subconjunto filtrado para así optimizar la precisión del entrenamiento [Leung y Hung, 2008].

Para la última técnica que se ha desarrollado, la técnica de conjunto busca abordar los problemas de inestabilidad observados en otras técnicas de selección de características cuando ocurren pequeñas perturbaciones en el conjunto de entrenamiento a través del sub muestreo. Este método se ejecuta en varios subconjuntos y las características obtenidas se fusionan en un subconjunto más estable [Lazar *et al.*, 2012].

Existe un caso donde la proporción entre las clases no es similar, lo que lleva a que el conjunto de datos se encuentre desbalanceado. Ejemplo de esto en un problema de clasificación binaria, las dos clases se encuentran en una razón de 5:1, lo que puede llevar a predicciones sesgadas, favoreciendo la clase mayoritaria. Además, esto puede resultar en métricas de evaluación engañosas como la exactitud, ya que en un caso donde el 95 % de los datos son de una clase, un modelo que siempre prediga esta única clase no será representativo para la clase minoritaria [Maldonado *et al.*, 2014].

Este último problema puede surgir durante la selección de características en un problema de clasificación, esto debido a que la distribución de las clases tiende demasiado a una clase. Una investigación utilizó una adaptación de SVM para seleccionar así atributos relevantes y discriminar entre clases que se encuentren bajo la condición de datos desbalanceados. Los resultados mostraron que la versión adaptada del algoritmo SVM, utilizando una función de pérdida balanceada, mejoró significativamente la precisión y el rendimiento general en conjuntos de datos desbalanceados [Maldonado *et al.*, 2014].

Extracción de características

La extracción de características es un proceso en el que se generan nuevas características o atributos a partir de un conjunto de datos, reduciendo así el número de recursos necesarios para el procesamiento de datos sin perder información relevante. Para muchos extractores de características, cuanto mayor es la dimensionalidad de entrada de un conjunto de datos, más probable es que ocurra el sobreajuste. Este sobreajuste ocurre cuando un modelo de machine learning se ajusta demasiado a los datos de entrenamiento específicos, capturando también el ruido y la aleatoriedad que pueden existir en los datos. Esto lleva a que el modelo tenga un rendimiento excelente en los datos de entrenamiento, pero su capacidad para generalizar y predecir con precisión nuevos datos disminuya. [Liu y Gillies, 2016]. La Figura 2.3 visualiza el cómo es el flujo de este proceso.

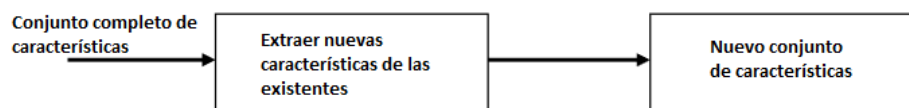


Figura 2.3: Proceso de selección de características

Fuente: [S Velliangiri y Thankumar, 2019].

Algunos algoritmos extracción de características pueden ser lineales como no lineales, y entre ellos, algunos de los métodos más conocidos incluyen el Análisis de Componentes Principales (PCA), el Escalamiento Multidimensional (MDS), el Mapeo Isométrico (ISOMAP), el Incrustado Local Lineal (LLE), el Análisis de Discriminante Lineal (LDA) y la Indexación Semántica Latente (LSI). Estos métodos son utilizados para realizar la transformación de características en el proceso de extracción de características [Zebari *et al.*, 2020].

PCA es una técnica de reducción de dimensionalidad. En esta técnica, se extrae información de diversos conjuntos de datos. El objetivo es producir un conjunto de características de menor dimensionalidad a partir del conjunto de datos original. En PCA, es crucial determinar el número de componentes principales. El número de componentes principales a elegir entre todos los componentes principales debe representar los datos de la mejor manera posible [Shah y Patel, 2016].

MDS es un algoritmo utilizado para visualizar y comprender datos al preservar las distancias entre pares de puntos de datos. Su objetivo principal es encontrar una disposición de puntos en un espacio de menor dimensionalidad, de manera que las distancias entre estos puntos coincidan lo más posible con las distancias originales proporcionadas como entrada. En resumen, MDS transforma datos de alta dimensionalidad en una representación más simple y de menor dimensionalidad, manteniendo las relaciones espaciales entre los puntos de datos [Pai *et al.*, 2022].

ISOMAP es una técnica de reducción no lineal de dimensionalidad que se utiliza para descubrir la estructura oculta de datos que están en una variedad no lineal. Su objetivo principal es preservar las distancias en la variedad entre pares de puntos de datos. Utiliza un grafo de vecindad para aproximar estas distancias, donde los puntos cercanos en el espacio de entrada tienen una buena aproximación de distancia euclidiana, mientras que para puntos distantes se suman distancias a través de conexiones de vecindad. Isomap proporciona una representación en un espacio euclidiano de menor dimensión que captura de manera efectiva la geometría interna de los datos, siendo útil tanto para visualización como para tareas de clasificación cuando los datos son de alta dimensión y tienen una estructura no lineal [Geng *et al.*, 2005].

LLE es un método no supervisado que reduce la dimensionalidad de datos asumiendo que están distribuidos sobre una superficie suave y no lineal en un espacio de muchas dimensiones. Preserva las estructuras geométricas locales del conjunto de datos original al reconstruir

cada punto como una combinación lineal de sus vecinos más cercanos. La fortaleza de LLE radica en su capacidad para manejar datos no lineales de manera eficiente, utilizando matrices dispersas. Esto puede resultar en tiempos y espacios computacionales menores en comparación con otras técnicas de extracción de características [Anowar *et al.*, 2021].

LDA es un método de reducción de dimensionalidad ampliamente utilizado en minería de datos y aplicaciones de aprendizaje automático. Su objetivo principal es transformar un conjunto de datos con múltiples características en un espacio de menor dimensión, mientras maximiza la separación entre las clases presentes en los datos originales. Esto ayuda a reducir los costos computacionales al trabajar con conjuntos de datos complejos y mejora la capacidad de los modelos para realizar tareas de clasificación al mantener la estructura discriminativa de los datos.

LSI es una técnica de reducción de dimensionalidad cuyo objetivo es proyectar los documentos en dimensiones semánticas latentes. El espacio semántico latente tiene menos dimensiones que en el espacio original. Esta técnica se basa en la aplicación de la descomposición de valores singulares (SVD) a una matriz [Shah y Patel, 2016].

SVD es una técnica matemática que descompone una matriz en tres componentes fundamentales: vectores singulares izquierdos, valores singulares y vectores singulares derechos. Se utiliza ampliamente para reducir la dimensionalidad de datos [Abdi, 2007].

2.3. Algoritmos genéticos

Los algoritmos genéticos (AG) tienen sus raíces desde la década de 1950, donde se estudiaron los sistemas evolutivos para poder aplicarlos como una herramienta de optimización para resolver problemas de ingeniería. Esta idea se inspiró en la variación genética y la selección natural para así desarrollar métodos de evolución de una población de posibles soluciones a un problema. El primero en mencionar los algoritmos genéticos fue John Holland en la década de 1960, inventó y desarrolló este concepto que, a diferencia de los evolutivos, este buscaba formalizar el fenómeno de la adaptación tal como se produce en la naturaleza y explorar cómo los mecanismos de adaptación de la naturaleza pueden simularse en sistemas informáticos.

Los AG se definen como un método que utiliza una población de “cromosomas”, que son cadenas de unos y ceros o “bits”, para posteriormente aplicar la selección natural mediante operadores inspirados en la genética, como el cruce o mutación. Cada cromosoma está compuesto por “genes” que son instancias de un “alelo” particular. El proceso de selección usualmente da preferencia a los cromosomas más aptos en la población para que estos se reproduzcan para así generar una descendencia más apta. El cruce intercambia partes de dos cromosomas y la mutación cambia aleatoriamente los valores de los alelos en el cromosoma.

La innovación de Holland era significativa, pues daba pie al marco de trabajo de los algo-

ritmos genéticos siguientes. esto a través de los conceptos de población, mutación y cruzamiento [Mitchell, 1998].

Un aspecto esencial de los AG es la población, puesto que estos algoritmos dependen de una población de soluciones candidatas, donde el tamaño de estas, que puede ser determinado por el usuario, puede influir en el rendimiento y escalabilidad de los AG. Si una población es pequeña, podría resultar en una convergencia prematura, pero si es muy grande, tener un tiempo de computación extenso.

Luego de haber definido estas poblaciones y elegir un parámetro que indique la calidad de las soluciones, se puede encontrar la mejor solución siguiendo los siguientes pasos, donde los más destacables son la selección, la recombinación o cruzamiento y la mutación, ya que estos son operadores que buscan mantener la diversidad de la población [Zeebaree *et al.*, 2017]:

1. Inicialización: Las soluciones candidatas iniciales de la población se generan, en su mayoría, aleatoriamente en todo el espacio de búsqueda.
2. Evaluación: Se evalúa la aptitud de las soluciones candidatas una vez terminada la inicialización.
3. Selección: Las mejores soluciones son seleccionadas, esto mediante distintos métodos, como puede ser la selección de la ruleta, donde cada solución candidata tiene una probabilidad de ser seleccionada proporcional a su aptitud, la selección por ranking, donde las soluciones candidatas son ordenadas según su aptitud y se eligen en función de su aptitud o el método del torneo, donde aleatoriamente se seleccionan conjuntos de soluciones y se eligen aquellas más aptas dentro de ese conjunto. Apuntar únicamente a la selección por aptitud puede generar que en la búsqueda uno se encuentre siempre en un óptimo local, en cambio, al permitir aquellas soluciones inferiores, puede potenciar la exploración de todo el espacio de búsqueda, es decir, diversificar.
4. Recombinación: En este paso se combinan partes de dos más soluciones para obtener nuevas y mejores soluciones, es decir, descendencia.
5. Mutación: Cuando dos o más cromosomas padres se operan mediante la recombinación, puede producirse una modificación de sus alelos por mutación, esto siendo una forma local y aleatoria de solución, realizando así un recorrido aleatorio en la vecindad de la solución candidata. Existen diversas formas de realizar la mutación que se explicarán posteriormente.
6. Reemplazo: La descendencia generada por los pasos anteriores reemplaza a la población parental original. Algunos métodos de reemplazo son el elitismo, donde se seleccionan únicamente los más aptos, intensificando en el espacio de solución en el que uno se encuentre, el borrar todo generacionalmente, que reemplaza toda la generación actual por la descendencia o el borrar n -últimos, donde aquellos n peores individuos son eliminados de la población actual y reemplazados por n individuos de la nueva descendencia [Alhijawi y Awajan, 2023].

Sobre los pasos mencionados, el reemplazo podría impactar negativamente en la diversidad de la población, puesto que los nuevos descendientes tienen alelos idénticos a los de sus padres. Por ello, la mutación se emplea para poder cambiar el alelo de los hijos de manera aleatoria.

Este operador tiene una probabilidad de mutación o P_m que se mantiene tan baja como sea posible para evitar que el AG se comporte como una búsqueda aleatoria. Algunas de las técnicas de mutación usadas son [Alhijawi y Awajan, 2023]:

- Mutación de punto único: Selecciona un gen al azar y lo cambia a un valor aleatorio. Esta mutación afecta solo a un gen en el cromosoma, cambiándolo por completo a un valor aleatorio.
- Mutación por inversión: Invierte toda la cadena de bits, bit por bit, con probabilidad P_m
- Mutación de variable aleatoria: Selecciona un número aleatorio ($K > 1$) de genes al azar y los cambia a valores aleatorios. Esta mutación implica seleccionar K genes aleatorios y cambiar sus valores por completo a valores aleatorios.
- Mutación uniforme: Cambia el valor del gen seleccionado a un valor uniformemente aleatorio seleccionado entre los límites superiores e inferiores para ese gen. Esta mutación altera el valor de un gen seleccionado según una distribución uniforme dentro de los límites definidos para ese gen.

2.4. Configuración de parámetros

Las ventajas de utilizar estos algoritmos comienzan por la búsqueda rápida y eficiente de combinaciones de parámetros que mejoran el rendimiento, siendo estos parámetros, los genes y combinaciones genéticas de los individuos utilizados para generar estas nuevas combinaciones. Esta forma de entregar nuevas soluciones tiene un alto rendimiento en un corto período de tiempo.

Hay pruebas considerables, tanto experimentales como teóricas, de que los AG pueden encontrar combinaciones de parámetros de alto rendimiento. Sin embargo, la desventaja que estos tienen es la posible convergencia a soluciones subóptimas, donde para solucionar este problema, se debe de aplicar un mayor grado de mutación que, en consecuencia, podría generar que el algoritmo se vuelva una búsqueda aleatoria [De Jong, 1988].

Entonces, los AG pueden optimizar una amplia variedad de funciones objetivas y encontrar soluciones óptimas o casi óptimas en espacios de búsqueda grandes y complejos, pero estos son sensibles al mismo cambio de parámetros.

Un caso de aplicación de algoritmos genéticos recientes fue realizado por Waleed Ali y Faisal Saeed, donde utilizando un filtro de selección de características, se seleccionan las características más relevantes de los *microarrays* utilizados para posteriormente aplicar un algoritmo genético para mejorar y optimizar las características seleccionadas previamente.

Dentro de la experimentación realizada en este caso, se obtuvo que para un clasificador *Nai-ve Bayes* (NB), después de aplicar los métodos híbridos de algoritmos genéticos, se logró una mejora en el *F-measure* de 85.39 %, 91.16 % y 83.02 % para los métodos IG-GA, IGR-GA Y CS-GA respectivamente en comparación con el clasificador NB independiente. Estas técnicas se pueden explorar más en profundidad en su investigación.

También se comparó el clasificador kNN, donde al emplear el método IG-GA propuesto, lograron una mejora en el *F-measure* de 83.65 % en comparación con el clasificador kNN independiente [Ali y Saeed, 2023].

Una aplicación de algoritmo genético para selección de características fue realizado por W. Siedlecki y J. Sklansky, quienes diseñaron un algoritmo para reducir la dimensionalidad en datos multidimensionales [Siedlecki y Sklansky, 1989]. En su investigación, buscaron el subconjunto de características más pequeño o el menos costoso en un algoritmo de *branch and bound* (B&B). Utilizando el error del clasificador para medir el rendimiento, definieron un subconjunto como factible si la tasa de error de este clasificador estaba por debajo del umbral de factibilidad. La naturaleza de la optimización restringida de su problema de búsqueda, que obliga a encontrar soluciones óptimas dentro de ciertos límites, como mantener el error del clasificador por debajo de un umbral específico, hacía que la selección de características no pudiera ser manejada fácilmente por algoritmos genéticos. Por lo tanto, implementaron una función de penalización para modificar el puntaje en su función de puntuación total aplicada a cada solución, orientando así al algoritmo genético hacia la selección de subconjuntos de características que cumplieran con los criterios de rendimiento y penalizando a aquellos que no los cumplieran [Siedlecki y Sklansky, 1989].

En la experimentación, el número inicial de características superaba los 20, se comparó la implementación del AG con métodos no exhaustivos como búsqueda secuencial y *branch and bound*, donde el AG mostró mejores resultados que B&B para datos con 24 características. Para los datos reales, con 30 dimensiones, el AG superó a la búsqueda secuencial, pero no se probó con B&B por la complejidad, encontrando subconjuntos de características más pequeños después de explorar un número similar de posibles soluciones.

La ventaja del AG es notoria cuando se aplica en espacios de características iniciales grandes, ya que los procedimientos secuenciales tienen complejidad temporal cuadrática y la búsqueda genética exhibe un aumento ligeramente superior al lineal en la complejidad temporal [Siedlecki y Sklansky, 1989].

En un caso de estudio realizado por Alexandre Plastino et al., sobre un algoritmo genético para optimizar el orden de etiquetas en cadenas de clasificadores multietiqueta, se observó que, en experimentos preliminares, se producía un sobreajuste al fijar un gran nú-

mero de generaciones para la ejecución del AG. Para enfrentar este problema y reducir la carga computacional, se limitó el número de generaciones del algoritmo genético en un rango de 15 a 20, dependiendo del tamaño del conjunto de datos. Esto demuestra que una de las limitaciones de los AG es la cantidad de generaciones necesarias para su ejecución [Gonçalves *et al.*, 2013].

Un estudio realizado por Feng Tan *et al.* evaluó tres métodos de selección de características: basado en la entropía, estadística T y eliminación recursiva de características SVM (SVM-RFE). Además, propusieron un marco basado en un algoritmo genético para combinar múltiples criterios de selección de características, demostrando su eficacia en la mejora de la selección de subconjuntos de características.

Los métodos basados en entropía evalúan la importancia de cada característica en función de la incertidumbre que reduce sobre la variable objetivo. T-Statistics utiliza pruebas estadísticas para seleccionar características en función de su capacidad para distinguir entre clases. SVM-RFE, por otro lado, es una técnica iterativa que utiliza máquinas de vectores de soporte para eliminar características de menor importancia.

En los experimentos se utilizaron datos de cáncer de próstata, donde el algoritmo genético logró una precisión de prueba del 94,1 %. Esto es comparable a la precisión máxima de la estadística T (97,1 %) y los métodos basados en entropía (97,1 %). Sin embargo, AG logró esta precisión con un número significativamente menor de funciones seleccionadas. Por ejemplo, AG logró una precisión del 94,1 % con solo cinco funciones, mientras que las estadísticas T y los métodos basados en entropía requirieron de un mayor número de características para lograr una precisión similar.

Además, AG seleccionó un subconjunto de ocho características en la base de datos ionosférica y logró la precisión más alta de 90,31 %, superando a SVM-RFE y T-Statistics para el mismo tamaño de subconjunto. Esto resalta la eficiencia de AG en la creación de subconjuntos de funciones más compactos y efectivos.

Los resultados de la investigación sugieren que los algoritmos genéticos son buenos en la selección de características debido a su capacidad para combinar múltiples criterios de selección, reducir la redundancia y seleccionar características que son complementarias entre sí. Esta capacidad de identificar subconjuntos de características efectivas, sugiere que los AG pueden generalizar mejor datos previamente desconocidos, mejorando la precisión y la generalización de los modelos de aprendizaje automático. La capacidad de ajustar parámetros específicos del AG, como puede ser su probabilidad de mutación, método de cruzamiento o selección, permite la optimización para diferentes problemas y conjuntos de datos, lo que lo convierte en una herramienta para mejorar el rendimiento del modelo de clasificación [Tan *et al.*, 2008].

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

En esta sección se exploran diversas estrategias de implementación de un algoritmo genético, tanto en el caso de estudio de Human Place como en el conjunto de datos de reconocimiento de actividades humanas (HAR). En el contexto del primer caso, se llevó a cabo una investigación previa que empleó modelos de Redes Neuronales (NN) y Random Forest (RF) para clasificar las alteraciones en la salud mental de estudiantes, abordando aspectos físicos, emocionales, cognitivos y organizacionales. Esta investigación sirvió de base para enfrentar las alteraciones que afectan al individuo y establecer planes de acción adecuados según la naturaleza de dichas alteraciones [Larrain *et al.*, 2022]. Asimismo, se propuso abordar la predicción de alteraciones como un problema de clasificación multi-etiqueta, lo que implicó la aplicación de una transformación binaria para abordar el problema desde otra perspectiva.

Respecto al segundo caso de estudio, este registró 561 atributos relacionados con la capacidad de determinar la actividad que realizaba una persona dentro de 6 posibles clases: caminar, subir escaleras, bajar escaleras, sentarse, estar de pie o acostarse. Debido a la alta dimensionalidad del conjunto de datos, la selección de características es crucial para mejorar la exactitud y el rendimiento de los modelos de clasificación, así como para mitigar los efectos de la maldición de la dimensionalidad. En esta investigación, se realizó la selección de características a través de distintos modelos de clasificación, como *Random Forest*, SVM, KNN y *Linear Discriminant Analysis* (LDA), con el objetivo de identificar el conjunto de atributos más relevantes y observar qué modelo entrega la mayor exactitud con una cantidad reducida de atributos [Chen *et al.*, 2020]. Para este fin, se utilizaron métodos de selección de características como *varImp()* de Random Forest, Boruta y Eliminación Recursiva de Características (RFE), comparando los resultados obtenidos en términos de exactitud y recall. Este enfoque de selección de características no solo busca simplificar los modelos y reducir el tiempo de entrenamiento, sino también mejorar la generalización y evitar el sobreajuste.

Con ambos contextos definidos, se propone la utilización de distintas variaciones del algoritmo genético en función de mantener o mejorar la calidad de las soluciones para los modelos utilizados en cada caso de estudio y poder determinar qué atributos pueden no ser utilizados para la clasificación en los respectivos problemas.

3.1. Implementación - Caso Human Place

Para cada implementación del algoritmo genético, se definirá lo siguiente: la representación de los cromosomas, el método utilizado para generar la población inicial, la función de evaluación para cada candidato a solución, el proceso de selección de los padres mediante el método de la ruleta (*Roulette Wheel*), el cruzamiento de los genes para aquellos seleccionados, la mutación de los cromosomas y el proceso de reemplazo de la población.

3.1.1. Representación:

En términos generales, el cromosoma en el algoritmo genético se define como una secuencia de ceros y unos que indican si un atributo dado se incluirá en el modelo de Redes Neuronales o en el de Random Forest. Esta representación permite mostrar de manera clara y flexible qué atributos serán considerados por el modelo, ya que cada bit del cromosoma puede activar o desactivar un atributo específico.

Por ejemplo, una secuencia que consiste únicamente de unos señala que se han seleccionado todos los atributos para la toma de decisiones, lo que refleja el enfoque original de estos modelos de clasificación, donde se consideran todas las características disponibles. En contraste, la presencia de ceros en la secuencia indica que el atributo correspondiente no será considerado en el modelo. Esta representación se puede visualizar en la Tabla 4, donde los atributos no considerados se muestran en gris, lo que representa un valor de cero en el cromosoma, y en blanco en caso contrario.

Tabla 4: Ejemplos de representación de atributos como cromosomas

Fuente: Elaboración Propia.

Cromosomas	Atributo1	Atributo2	Atributo3	Atributo4	Atributo5	Atributo6
1111111	1	1	1	1	1	1
1100111	1	1	0	0	1	1
0001111	0	0	0	1	1	1
1010101	1	0	1	0	1	1
1111100	1	1	1	1	1	0

3.1.2. Población inicial:

Dado que el Algoritmo Genético (AG) opera en base a una población de soluciones, se inicia generando una población inicial compuesta por p_s cromosomas únicos. Sin embargo, se realizó una excepción en esta fase del proceso al mantener ciertos atributos inmutables, en conformidad con las restricciones impuestas por las preguntas que no deben ser modificadas. Estas preguntas, entregadas por el equipo Human Place, establecen que “no se pueden

alterar las preguntas claves entregadas por el equipo de Human Place”[Larrain *et al.*, 2022]. El pseudocódigo 1 comienza definiendo la variable N, que representa el número de atributos en cada cromosoma (línea 1). A continuación, se establece “TamañoPoblacion”, que indica el número total de cromosomas que compondrán la población inicial (línea 2). Se define “UnchangeAttr” como una lista de atributos que no deben ser modificados durante el proceso (línea 3). Luego, se inicializa “Poblacion” como una lista vacía que almacenará los cromosomas generados (línea 4). La variable “InitCromos” se establece como un cromosoma inicial compuesto exclusivamente por valores “1” (línea 5).

Posteriormente, se entra en un bucle *While* que se ejecuta mientras el tamaño de la población generada sea menor que el “TamañoPoblacion” deseado (línea 6). Dentro de este bucle, se crea un nuevo cromosoma, “NuevoCromosoma”, que inicialmente es una copia de “InitCromos” (línea 7). Luego, se inicia un bucle *For* que se ejecuta un número aleatorio de veces, determinado por un valor aleatorio entre el tamaño de los cromosomas y sin considerar los atributos inmutables (línea 8). En cada iteración del bucle *For*, se selecciona una posición aleatoria dentro del cromosoma (línea 9). Si esta posición no pertenece a los atributos inmutables definidos en “UnchangeAttr” (línea 10), el valor en esa posición se alterna entre 0 y 1 (línea 11). Esta estrategia permite una exploración más amplia de posibles soluciones, ya que no se limita a cambiar solo un bit a la vez, sino que se efectúan múltiples cambios, lo que favorece una exploración más exhaustiva del espacio de posibles soluciones. Finalmente, el “NuevoCromosoma” generado se añade a la “Poblacion” (línea 13). Este proceso se repite hasta que la población alcanza el tamaño deseado (línea 14).

Algorithm 1 Generar Población

```
1: variable N ← Número de atributos
2: variable TamañoPoblacion ←  $\in \mathbb{N}^+ - \{0\}$ 
3: variable UnchangeAttr ← [attrX,attrY...]  $\exists \{attr1,attr2,attr3...,attrN\}$ 
4: variable Poblacion ← [ ]
5: variable InitCromos ← “11111...N”
6: while Size(Poblacion) < TamañoPoblacion do
7:   variable NuevoCromosoma = InitCromos
8:   for _ do in range(random(1,Size(InitCromos)-Size(UnchangeAttr))):
9:     variable posicion_new_bit ← random(0, Size(InitCromos))
10:    if posicion_new_bit not in UnchangeAttr then
11:      NuevoCromosoma[posicion_new_bit] ← “0” if NuevoCromosoma[posicion_new_bit] == “1”, else “1”
12:    end if
13:  end for
14:  Poblacion.append(NuevoCromosoma)
15: end while
```

3.1.3. Función de Evaluación:

Teniendo una población de cromosomas, obtenemos un valor de aptitud para cada uno de estos cromosomas usando un modelo de Redes Neuronales o de RF a través de la exactitud y los falsos negativos obtenidos del modelo utilizado, donde se tiene un valor para cada alteración, esto sumado al grupo de control, que corresponde a la clase sin alteraciones [Larrain *et al.*, 2022].

Luego, se obtiene un puntaje total por cada cromosoma, considerando las 5 alteraciones presentes. Esto se obtiene multiplicando la exactitud obtenida por un cromosoma en las alteraciones (control, física, emocional, cognitiva y organizacional) por un valor de penalización asociado. La penalización adopta una forma cuadrática, como se aprecia en la ecuación (9), lo que implica que el mayor crecimiento en la penalización se produce con la presencia de al menos un falso negativo, y a medida que aumenta el número de falsos negativos, el crecimiento del valor de la penalización será menor.

Este puntaje total acumulado se representa en la ecuación (10), donde $PuntajeTotal_x$ es el puntaje total obtenido por el cromosoma x en las alteraciones, $Exactitud_i$ es la exactitud del modelo para la alteración i , $Penalización_i$ es la penalización asociada a esta, y z es la cantidad de alteraciones, que para el caso de *Human Place* es de 5.

Esto significa que, además de considerar la exactitud del modelo, se tiene en cuenta la capacidad del modelo para identificar correctamente los casos positivos, reflejada en la penalización aplicada a los falsos negativos. Asimismo, la importancia de la exactitud en el cálculo del puntaje total resalta la relevancia de la correcta predicción de positivos y negativos del modelo en el proceso de evaluación. Los puntajes totales obtenidos para cada cromosoma se almacenan para su posterior análisis y selección en el algoritmo genético.

$$Penalización = \frac{1}{(FN + 1)^2} \quad (9)$$

$$PuntajeTotal_x = \sum_{i=1}^z (Exactitud_i \cdot Penalización_i) \quad (10)$$

3.1.4. Proceso de selección:

Teniendo evaluado cada cromosoma con este puntaje, realizamos la selección de N_{padres} padres a través de probabilidades ponderadas a cada posible padre, obteniendo esta probabilidad dividiendo el puntaje de cada padre por la suma total de todos los otros puntajes. Así, aquellos padres con mayor puntaje tienen una mayor probabilidad de ser elegidos, fomentando la evolución de la población en el algoritmo genético, potenciando aquellos con mejor

desempeño. Esta selección es conocida como *roulette wheel*, donde al observar el ejemplo de la Figura 3.1, se puede notar que ciertos cromosomas tienen una mayor porción de la ruleta según su valor de aptitud, por lo que es más probable que sean seleccionados para ser los futuros padres.

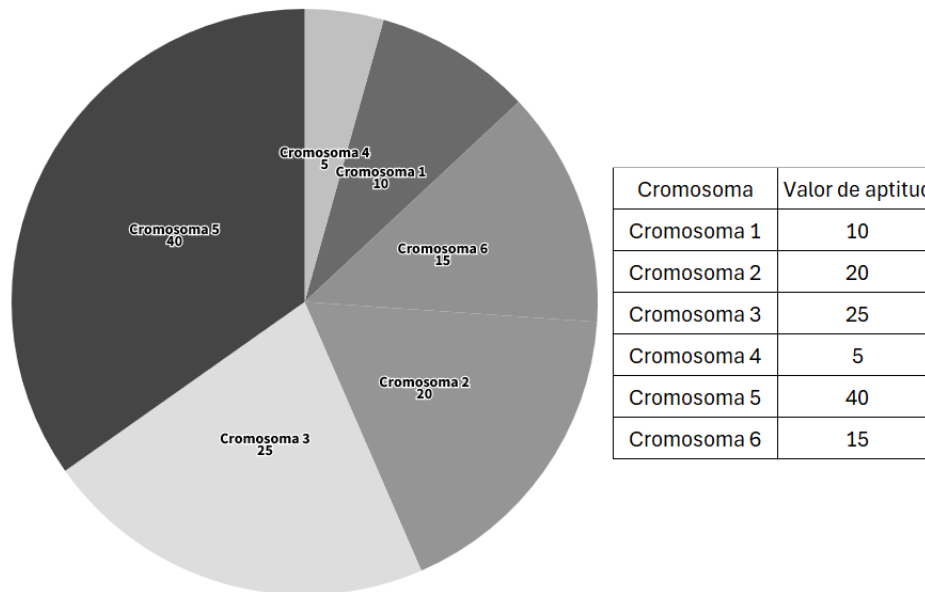


Figura 3.1: Ejemplo de selección por *roulette wheel*
Fuente: Elaboración propia.

3.1.5. Cruzamiento:

Ahora, mediante cruzamiento en un punto, se realizará el intercambio de genes entre los padres seleccionados en distintos puntos de cruzamiento hasta tener la cantidad de hijos (*offspring*) que se indiquen. Es importante destacar que se evita que el punto de corte coincida con posiciones predefinidas como no modificables. Una vez establecido este punto, el cual se obtiene seleccionando un punto aleatorio dentro de los genes del cromosoma, se genera un nuevo hijo combinando las secciones de cromosomas de ambos padres. Ejemplo de este tipo de cruzamiento se puede observar en la Figura 3.2 Este enfoque garantiza que el hijo herede una combinación diversa de genes de ambos progenitores. Notar que al realizar múltiples cruzamientos en una misma selección de padres, esto puede llevar a una pérdida de diversidad en la población, pero debido al tamaño del cromosoma tratado en este caso, y el cómo se implementó el cruce, se puede variar si se toma primero los genes del primer padre para la primera sección o las del segundo y viceversa.

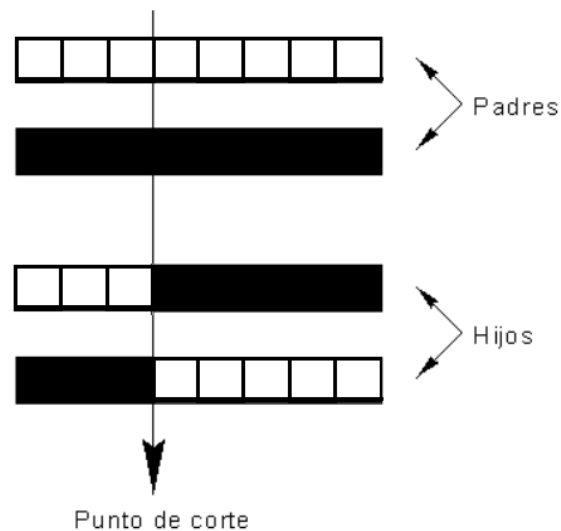


Figura 3.2: Ejemplo de cruzamiento en un punto
Fuente: [Kuri y Galaviz, 2002]

3.1.6. Mutación:

Para esta nueva generación, se asigna una probabilidad p_m de aplicar una mutación a un número aleatorio de sus genes mutables. Esto implica introducir cambios aleatorios en los genes dentro de su rango binario, alternando entre los valores 0 y 1. El propósito es diversificar la población y generar individuos nuevos [De Jong, 1988].

3.1.7. Reemplazo:

Finalmente, para sustituir a la población anterior con la nueva generación, se emplea un proceso de reemplazo total, donde todos los individuos de la población anterior son completamente reemplazados por los nuevos individuos. Este proceso se repite durante N generaciones, donde finalmente almacenamos todos los resultados obtenidos a lo largo de estas generaciones para poder analizar cómo fue el rendimiento en general de los cambios en los atributos, así como en los distintos conjuntos de datos disponibles.

Para entender mejor esta implementación, se presenta el pseudocódigo 2 donde se describe el proceso del algoritmo genético para entrenar y evaluar modelos de NN o RF utilizando distintos datasets y conjuntos de hiperparámetros. Comienza inicializando una lista de *datasets* representados por archivos CSV (línea 1) y luego carga los hiperparámetros del modelo desde un archivo (línea 2), definiendo parámetros esenciales como el número de capas, neuronas, épocas, tamaño de lote, tasa de aprendizaje, entre otros.

El tamaño de la población inicial de cromosomas (línea 3), el número máximo de generacio-

nes (línea 4) y la generación actual (línea 5) se establecen junto con la probabilidad de mutación (línea 6) y la cantidad de padres a seleccionar para la reproducción (línea 7). El algoritmo itera sobre cada dataset (línea 8), leyendo cada archivo CSV para generar una población utilizando el pseudocódigo 1 (línea 10) y declara una lista para ir rastreando los cromosomas evaluados(línea 11).

Utilizando un ciclo que itera desde 0 hasta la cantidad máxima de generaciones (línea 12), se iteran los cromosomas de la población generada (línea 13). Si un cromosoma no ha sido evaluado previamente (línea 14), se entrena y evalúa el modelo utilizando los datos, los hiperparámetros correspondientes, y el cromosoma (líneas 15-19). Las métricas de rendimiento obtenidas se guardan en un registro (línea 18), y el cromosoma se marca como visitado (línea 20). Este proceso se repite para cada tipo de hiperparámetro (línea 15).

Después de evaluar la población actual, se seleccionan los mejores padres basados en su aptitud (línea 23), se realiza el método de cruzamiento en un punto para generar descendencia (línea 24) y se aplica la mutación a esta descendencia (línea 25). La *offspring* mutada se evalúa de manera similar a la población inicial (líneas 26-30), y se seleccionan los nuevos cromosomas para la siguiente generación (línea 36).

Este ciclo de selección, cruzamiento, mutación y evaluación continúa hasta que se alcanza el número máximo de generaciones. Finalmente, se repite el proceso para cada dataset (línea 9), asegurando que todos los conjuntos de datos sean utilizados para entrenar y evaluar los modelos con el enfoque del AG.

Algorithm 2 Flujo Algoritmo Genético - Primera implementación

```
1: Datasets  $\leftarrow$  [csv1, csv2...]
2: hiperparametros  $\leftarrow$  LeerArchivo(hyperparameters)  $\triangleright$  Hiperparámetros del modelo de NN/RF
3: Npoblacion  $\leftarrow$  Tamaño de la población
4: MaxGeneraciones  $\leftarrow$  Número máximo de generaciones
5: Generaciones  $\leftarrow$  0
6:  $p_m$   $\leftarrow$  Probabilidad de mutación
7: NpadresSeleccionados  $\leftarrow$  Cantidad de padres a seleccionar
8: for Archivo in Datasets do
9:   Datos  $\leftarrow$  LeerComoDataframe(Archivo)
10:  poblacion  $\leftarrow$  GenerarPoblacion(Npoblacion, Datos)
11:  CromosVisitados  $\leftarrow$  []
12:  while Generaciones < MaxGeneraciones do
13:    for cromosoma in poblacion do
14:      if cromosoma not in en CromosVisitados then
15:        for tipo in ['control', 'fisical', 'emotional', 'cognitive', 'organizational'] do
16:          TipoHyperparam  $\leftarrow$  hiperparametros[tipo]
17:          metricas  $\leftarrow$  TrainEvalModelo(Datos, TipoHyperparam, cromosoma)
18:          GuardarRegistro(cromosoma, tipo, metricas)
19:        end for
20:        CromosVisitados  $\leftarrow$  agregar(cromosoma)
21:      end if
22:    end for
23:    PadresSeleccionados  $\leftarrow$  SeleccionarPadres(poblacion, NpadresSeleccionados)
24:    Offspring  $\leftarrow$  CruzamientoUnPunto(PadresSeleccionados)
25:    OffspringMutados  $\leftarrow$  Mutacion(Offspring,  $p_m$ )
26:    for cromosoma in OffspringMutados do
27:      if cromosoma not in CromosVisitados then
28:        for tipo in ['control', 'fisical', 'emotional', 'cognitive', 'organizational'] do
29:          TipoHyperparam  $\leftarrow$  hiperparametros[tipo]
30:          metricas  $\leftarrow$  TrainEvalModelo(Datos, TipoHyperparam, cromosoma)
31:          GuardarRegistro(cromosoma, tipo, metricas)
32:        end for
33:        CromosVisitados  $\leftarrow$  agregar(cromosoma)
34:      end if
35:    end for
36:    poblacion  $\leftarrow$  seleccionar_nuevos_cromosomas(poblacion, OffspringMutados)
37:    Generaciones  $\leftarrow$  Generaciones + 1
38:  end while
39: end for
```

3.2. Implementación - Caso HAR

Se definen los mismos pasos a seguir que en la primera implementación del caso Human Place, pero con distintos métodos. Sin embargo, la principal diferencia es en la representación de los datos. En este caso, se tienen 561 atributos. Por lo tanto, en lugar de modificar cadenas extensas de ceros y unos, se guardarán los índices de los atributos que se utilizarán, de acuerdo con la cantidad de atributos objetivos deseados.

3.2.1. Representación:

En la primera instancia de investigación, se toma toda la cadena de atributos y se cambian entre unos y ceros para determinar qué atributos utilizar. Para este caso, debido a la investigación utilizada para comparar algoritmos [Chen *et al.*, 2020], se parametriza el tamaño de los cromosomas mediante *AttrSizeTarget* atributos, donde *AttrSizeTarget* corresponde a la cantidad de atributos que se busca utilizar en las distintas generaciones. Este valor se determina por la cantidad de características utilizadas en la investigación mencionada para realizar las comparaciones adecuadas bajo escenarios similares.

Para referirse de mejor manera a esta representación, consultar la Tabla 5, donde la principal diferencia es el cómo se verán los cromosomas. Estos indicarán el índice del atributo que será considerado, en lugar de la cadena de unos y ceros.

Tabla 5: Ejemplos de representación de atributos como cromosomas para el caso HAR

Fuente: Elaboración Propia.

Cromosoma	Attr1	Attr2	Attr3	...	Attr25	...	Attr561
4, 15, 25, 104, 203, 432	0	0	0	...	1	...	0
2, 7, 10, 25, 332, 504	0	1	0	...	0	...	0

3.2.2. Población inicial:

La población de cromosomas corresponde a una lista de índices de características seleccionadas aleatoriamente. El tamaño de la población es determinado por p_s y cada cromosoma tiene una longitud de *AttrSizeTarget* seleccionados aleatoriamente y sin repetición. Para cada cromosoma, se genera una permutación aleatoria de los índices de las características y se seleccionan los primeros *AttrSizeTarget* elementos de esta permutación. Este proceso asegura la diversidad inicial en la población, proporcionando una base para la evolución de las soluciones a través de las generaciones del algoritmo genético. Una descripción más detallada de este paso se puede ver en el pseudocódigo 3.

Algorithm 3 Generar Población Caso HAR

```
1: variable AttrSizeTarget  $\leftarrow$  Número de atributos a utilizar
2: variable  $p_s \leftarrow \in \mathbb{N}^+ - \{0\}$ 
3: variable Poblacion  $\leftarrow [ ]$ 
4: variable ListaAtributos  $\leftarrow [\text{attr1}, \text{attr2} \dots \text{attrN}]$  ▷ N=Cantidad atributos
5: for _ in range( $p_s$ ) do
6:   Poblacion.append(random.permutation(ListaAtributos)[0:AttrSizeTarget])
7: end for
```

3.2.3. Función de Evaluación:

Ahora, iterando por cada cromosoma de la población, se inicializa un clasificador, que puede ser KNN, RF, LDA o SVM, según el caso a comparar. Primero, se seleccionan las características correspondientes al cromosoma actual. Estas características se normalizan para asegurar que todas tengan una escala similar, lo cual es importante para evitar que características con mayores magnitudes dominen el proceso de aprendizaje del clasificador. Luego, se utiliza la técnica de validación cruzada para evaluar el clasificador. En este proceso, los datos se dividen en múltiples subconjuntos y el modelo se entrena y evalúa varias veces, cada vez con un subconjunto diferente de datos de entrenamiento y prueba. La métrica utilizada para esta evaluación es la exactitud. Al final, se obtiene la media de las puntuaciones de validación cruzada, la cual se usa como el valor de aptitud de cada individuo.

3.2.4. Proceso de selección:

Para seleccionar los p_s padres, se emplea la estrategia de selección lineal por rango. En este proceso, los individuos de la población se ordenan según su aptitud y se les asigna un rango correspondiente. Esto implica que los individuos con mejores valores de aptitud tienen una mayor probabilidad de ser seleccionados, aunque aún existe la posibilidad de que se elijan individuos menos aptos. Esta estrategia resulta especialmente beneficiosa cuando se trabaja con un número fijo de características en los cromosomas, lo cual limita su tamaño.

Al comparar la diversidad genética, calculada como la diversidad genética promedio dentro de la población de padres, entre el método de selección por ruleta y el método por rango, se observó que este último proporcionó resultados superiores al mantener una mayor variedad de características en la población seleccionada. Esto asegura una exploración más exhaustiva del espacio de soluciones, ayudando a evitar el estancamiento en óptimos locales. Una vez calculadas las probabilidades de selección, los padres se eligen aleatoriamente de la población ponderando estas probabilidades.

3.2.5. Cruzamiento:

Para realizar el cruzamiento de esta selección de padres, se emplea uno de los métodos implementados: cruzamiento uniforme, cruzamiento en un punto o cruzamiento aleatorio múltiple. Estos métodos permiten generar descendencia intercambiando los genes entre los padres seleccionados para crear nuevos individuos. En el proceso, se itera sobre cada gen de los padres y, para cada método como el cruzamiento uniforme o el cruzamiento aleatorio múltiple, se elige aleatoriamente de cuál de los dos padres tomar dicho gen, determinando esto mediante la generación de un número aleatorio. Este intercambio se lleva a cabo para todos los genes seleccionados, lo que resulta en nuevos individuos que combinan características de ambos padres.

3.2.6. Mutación:

Para cada individuo generado en el cruzamiento, se genera un número aleatorio por cada gen para determinar si se aplica la mutación, en función de si el número es menor que la tasa de mutación p_m . Si esta condición se cumple, la mutación consiste en reemplazar el valor del gen por la diferencia entre el valor máximo de los genes del individuo y el valor actual del gen. Este paso introduce variabilidad en la descendencia, ayudando a explorar nuevas áreas del espacio de soluciones y evitando que la población se estanque en óptimos locales.

Para ilustrar mejor esta mutación, se presenta el siguiente ejemplo donde se toman dos cromosomas de la población:

Cromosoma 1: [3, 7, 5, 4, 9]

Cromosoma 2: [6, 1, 8, 4, 2]

El máximo valor en estas listas es 9. Si aplicamos la mutación con una tasa de mutación del 50 % y se aplica a los individuos, estos resultan en:

Cromosoma 1 mutado: [3, 2, 5, 4, 9] (mutación en el segundo elemento)

Cromosoma 2 mutado: [6, 1, 0, 4, 2] (mutación en el tercer elemento)

Este ejemplo ilustra cómo la mutación invierte algunos valores en relación con el máximo encontrado en cada lista. En el caso del cromosoma 1, se seleccionó el valor en la segunda posición para ser mutado. El valor mutado se obtiene como $9 - 7 = 2$, lo que resulta en la mutación del cromosoma 1 original [3, 7, 5, 4, 9] a [3, 2, 5, 4, 9]. Este método introduce variabilidad dentro de la población en un algoritmo genético.

3.2.7. Reemplazo:

Como último paso, la población actual se reemplaza con la nueva generación de individuos obtenida a partir de la mutación. Este corresponde a un reemplazo generacional completo donde luego se va conservando las mejores aptitudes obtenidas a lo largo de las N_g generaciones.

Para ilustrar mejor el flujo de esta implementación, se presenta el pseudocódigo 4 que detalla el algoritmo genético para el caso HAR. Se inicia definiendo varias variables esenciales. Se establece *AttrSizeTarget* como el número de atributos a utilizar (línea 1), p_s como el número de padres a seleccionar (línea 2), *Poblacion* como una lista vacía (línea 3) y *ListaAtributos* como una lista de todos los atributos disponibles (línea 4). *Ngeneraciones* se define como un número positivo mayor a 1 que representa la cantidad de generaciones a iterar (línea 5), *MejorFitnessAllIters* comienza con un valor inicial de cero (línea 6) y buscará conservar el mejor valor a través de todas las iteraciones, *FitValues* es una lista vacía para almacenar los valores de aptitud (línea 7), y p_m es la probabilidad de mutación (línea 8). A continuación, se carga el archivo CSV que contiene los datos necesarios (línea 9).

Luego, se genera la población inicial utilizando el algoritmo 3, que toma como parámetros *AttrSizeTarget*, p_s , *Poblacion* y *ListaAtributos* (línea 10). Se inicia un bucle para iterar a través del número de generaciones definidas en *Ngeneraciones* (línea 11). Dentro de este ciclo, se evalúan los valores de aptitud de la población actual utilizando la función de evaluación *FunEval* y se almacenan en *FitValues* (línea 12). Notar que se puede precisar la métrica que se desea evaluar, para este caso en particular se toma la exactitud o *accuracy* en el pseudocódigo para la demostración del flujo del algoritmo. Los padres se seleccionan mediante la selección lineal por rango *LinealRankSelection* basada en los valores de aptitud (línea 13).

Los hijos se generan mediante un proceso de cruzamiento que puede variar entre uniforme, múltiple o en un punto (línea 14). Los hijos mutados se obtienen aplicando la función *Mutacion* a los hijos generados (línea 15). La población actual se reemplaza completamente por los hijos mutados (línea 16). Se determina el mejor valor de aptitud en la generación actual y se almacena en *MejorFitValue* (línea 17).

Si *MejorFitValue* es mayor que el último valor almacenado en *MejorFitnessAllIters*, este se agrega a la lista *MejorFitnessAllIters* (líneas 18-20). El bucle continúa hasta completar el número de generaciones especificado.

Algorithm 4 Flujo Algoritmo Genético - Caso HAR

```
1: variable AttrSizeTarget  $\leftarrow$  Número de atributos a utilizar
2: variable  $p_s \leftarrow \in \mathbb{N}^+ - \{0\}$ 
3: variable Poblacion  $\leftarrow [ ]$ 
4: variable ListaAtributos  $\leftarrow [\text{attr1}, \text{attr2} \dots \text{attrN}]$   $\triangleright N = \text{Cantidad atributos}$ 
5: variable Ngeneraciones  $\leftarrow \in \mathbb{N}^+ > 1$ 
6: variable MejorFitnessAllIters  $\leftarrow [0]$ 
7: variable FitValues  $\leftarrow [ ]$ 
8: variable  $p_m \leftarrow \text{Prob. Mutacion}$ 
9: variable data  $\leftarrow \text{Cargar archivo CSV}$ 
10: Poblacion = Algoritmo3(AttrSizeTarget,  $p_s$ , Poblacion, ListaAtributos)  $\triangleright \text{Generar población}$ 
11: for iter in range(Ngeneraciones) do
12:   FitValues  $\leftarrow \text{FunEval}(\text{Poblacion}, \text{data}, \text{AttrSizeTarget}, \text{metrica} = \text{"Accuracy"}), \text{modelo} = \text{"KNN/RF/SVM/LDA"}$ 
13:   Padres  $\leftarrow \text{LinealRankSelection}(\text{Poblacion}, \text{FitValues}, p_s)$ 
14:   Offspring  $\leftarrow \text{Cruzamiento}(\text{Padres}, \text{modoCruzamiento} = M \rightarrow [\text{Uniforme}, \text{Multiple}, \text{UnPunto}])$ 
15:   MutatedOffspring  $\leftarrow \text{Mutacion}(\text{Offspring}, p_m)$ 
16:   Poblacion  $\leftarrow \text{MutatedOffspring}$   $\triangleright \text{Reemplazo total de la población}$ 
17:   MejorFitValue  $\leftarrow \max(\text{FitValues})$ 
18:   if MejorFitValue > MejorFitnessAllIters[-1] then
19:     MejorFitnessAllIters.append(MejorFitValue)
20:   end if
21: end for
```

CAPÍTULO 4

RESULTADOS

Este capítulo busca presentar los resultados obtenidos a lo largo de las implementaciones realizadas del algoritmo genético, detallando si los cambios propuestos a este algoritmo entregaron resultados distintos, de igual o mejor calidad que los originales. Se analizan las métricas de calidad, con un enfoque particular en la exactitud, así como la comparación de verdaderos positivos y falsos negativos entre diferentes modelos con la selección de características del algoritmo genético. Estos resultados proporcionan una visión detallada del rendimiento del algoritmo genético en la tarea de resolver problemas de clasificación en contextos diversos.

En particular, se revisarán las métricas de exactitud y falsos negativos para el caso de Human Place en los datos sintéticos generados, de acuerdo con la investigación realizada por [Larrain *et al.*, 2022]. Esto se hace con el objetivo de reducir la cantidad de características utilizadas para determinar la presencia de alguna perturbación en el sujeto. Los datos sintéticos se generaron creando versiones alternativas de los registros originales, con alteraciones aleatorias para introducir diferentes niveles de ruido, asegurando así una evaluación robusta de los modelos. Además, se aplicará tanto RF como redes neuronales para evaluar el rendimiento de los modelos.

Para el caso de reconocimiento de actividades humanas, utilizando el estudio de [Chen *et al.*, 2020], se compara la calidad de las distintas soluciones según las métricas de exactitud y recall, para establecer si el uso del algoritmo genético para la selección de características mejoró o mantuvo la calidad.

Con el fin de proporcionar comparaciones con distintos números de atributos con el mismo conjunto de datos HAR, se toma el estudio de [Ismi *et al.*, 2016], donde, a través de k-means, se seleccionaron de 10 a 70 características para ser evaluadas posteriormente por un clasificador bayesiano ingenuo. Realizando de esta manera, la comparativa de selección de características por algoritmo genético en lugar de k-means.

4.1. Caso de estudio Human Place

Para este caso de estudio se presenta el análisis y resultados obtenidos por parte de la investigación Human Place [Larrain *et al.*, 2022] con la aplicación de redes neuronales o *random forest* como uno de los métodos de resolución del problema de clasificación. Posteriormente, se entregan los resultados obtenidos con la aplicación del algoritmo genético como el método de selección de características previo al uso del modelo de redes neuronales para poder realizar las comparaciones entre ambos métodos y determinar si las distintas métricas tuvieron un mejor rendimiento o se mantuvo.

Tabla 6: Valor de accuracy para las distintas instancias de datos sintéticos mediante Redes Neuronales y su promedio. Fuente: [Larrain *et al.*, 2022].

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional
20Q	0.920	0.957	0.940	0.953	0.937
30Q	0.880	0.930	0.963	0.917	0.930
40Q	0.867	0.930	0.973	0.873	0.890
50Q	0.823	0.930	0.950	0.840	0.887
60Q	0.803	0.967	0.937	0.843	0.890
AVG	0.859	0.943	0.953	0.885	0.907

Tabla 7: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: [Larrain *et al.*, 2022].

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional
20Q	9.6 %	4.0 %	6.0 %	4.3 %	6.3 %
30Q	12.6 %	7.3 %	3.6 %	8.6 %	7.0 %
40Q	14.6 %	7.0 %	2.3 %	13.0 %	11.0 %
50Q	19.3 %	7.0 %	4.6 %	16.0 %	11.3 %
60Q	20.0 %	3.3 %	6.6 %	15.6 %	11.0 %
AVG	15.22 %	5.12 %	4.62 %	11.52 %	9.32 %

Tabla 8: Valor de accuracy para las distintas instancias de datos sintéticos mediante Random Forest y su promedio. Fuente: [Larrain *et al.*, 2022].

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional
20Q	0.980	0.947	0.933	0.923	0.850
30Q	0.973	0.923	0.937	0.953	0.940
40Q	1.000	0.920	0.933	0.977	0.967
50Q	0.987	0.913	0.933	0.980	0.977
60Q	0.980	0.910	0.927	0.977	1.000
AVG	0.984	0.923	0.933	0.962	0.947

Tabla 9: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Random Forest. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional
20Q	0.0 %	4.7 %	4.7 %	7.0 %	14.3 %
30Q	0.0 %	7.0 %	4.7 %	3.3 %	5.3 %
40Q	0.0 %	7.3 %	4.7 %	0.7 %	3.3 %
50Q	0.0 %	7.7 %	4.3 %	0.3 %	0.7 %
60Q	0.0 %	8.3 %	4.3 %	0.0 %	0.0 %
AVG	0.0 %	7.0 %	4.5 %	2.3 %	4.7 %

4.1.1. Primera implementación - Redes Neuronales

Las variaciones aplicadas para esta primera implementación fueron las siguientes:

- Número de generaciones: 10 - 20
- Tamaño de población: 20 - 40
- Selección de padres: Selección por ruleta
- Cruzamiento: Cruzamiento en un punto.
- Probabilidad de mutación utilizada: 1 % y 10 %.
- Reemplazo de población: Reemplazo total.

En la tabla 6 se presentan los valores de exactitud obtenidos por la investigación original utilizando redes neuronales en diversos conjuntos de datos sintéticos. Los valores de exactitud oscilan entre 0.803 y 0.973, con un promedio general que varía entre 0.859 y 0.953 según la alteración evaluada. Los datos muestran un desempeño consistentemente alto en las métricas Físico y Emocional, mientras que las métricas Control y Cognitivo presentan una disminución en el rendimiento con el aumento de la perturbación en los datos.

En comparación, los resultados obtenidos mediante la implementación del algoritmo genético como método de selección de características antes de aplicar redes neuronales, presentados en la tabla 10, muestran una mejora notable en la exactitud promedio. Los valores oscilan entre 0.910 y 0.977, con un promedio general entre 0.918 y 0.976 según la alteración. Esta mejora se observa en todas las categorías, destacando especialmente en la métrica Control, que muestra una exactitud promedio significativamente superior a la obtenida en el estudio original.

La tasa de falsos negativos, mostrada en la tabla 7, revela patrones similares. Los valores de la investigación original varían entre 2.3 % y 20.0 %, con un promedio general entre 4.62 % y 15.22 %. Los resultados obtenidos con la aplicación del algoritmo genético, como se observa en la tabla 11, presentan una reducción considerable en la tasa de falsos negativos, con valores que oscilan entre 2.0 % y 8.1 % y un promedio general que varía entre 2.1 % y 7.3 %. Esta reducción es particularmente notable en las perturbaciones de tipo Control y Cognitivo, que presentan una mayor reducción en comparación con las otras métricas.

Tabla 10: Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.977	0.920	0.977	0.937	0.972	1 %
30Q	0.953	0.923	0.976	0.928	0.968	1 %
40Q	0.931	0.919	0.973	0.925	0.954	1 %
50Q	0.922	0.918	0.977	0.925	0.936	1 %
60Q	0.920	0.910	0.975	0.923	0.926	1 %
AVG	0.941	0.918	0.976	0.928	0.951	

Tabla 11: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.0 %	4.3 %	2.2 %	8.1 %	1.1 %	1 %
30Q	5.3 %	11.2 %	2.2 %	4.1 %	0.0 %	1 %
40Q	3.2 %	4.2 %	2.1 %	7.4 %	1.2 %	1 %
50Q	1.3 %	5.2 %	2.2 %	7.3 %	3.1 %	1 %
60Q	1.1 %	8.4 %	2.1 %	7.4 %	5.1 %	1 %
60Q	2.2 %	6.7 %	2.2 %	6.9 %	2.1 %	

Al considerar los mejores cromosomas para cada caso, presentados en la tabla 12, se observa una mejora adicional en la exactitud de las redes neuronales. Los valores de exactitud para los mejores cromosomas varían entre 0.912 y 1.000, superando en muchos casos a los valores promedio presentados anteriormente. Esta mejora es particularmente notable en los tipos Emocional y Organizacional, donde se alcanzan valores cercanos al 1.000 en múltiples instancias.

La tasa de falsos negativos para los mejores cromosomas, mostrada en la Tabla 13, también refleja esta mejora. Los valores varían entre 0.0 % y 8.4 %, con reducciones significativas en comparación con los promedios generales. En particular, la alteración de Control muestra una tasa de falsos negativos de 0.0 % para el caso 20Q, destacando la efectividad del mejor cromosoma en la mejora de la exactitud de la clasificación.

Sin embargo, es importante notar que la tasa de falsos negativos para la alteración Física presenta valores relativamente altos, especialmente en los casos 30Q y 60Q, con tasas de 11.2 % y 8.4 % respectivamente. Esto sugiere que, a pesar de la mejora general en la exactitud, la selección de características realizada mediante el algoritmo genético podría no haber optimizado completamente las características relevantes para esta perturbación.

El alto valor de falsos negativos en el tipo Física podría indicar que el conjunto de características seleccionadas para este caso considera características que no aportan información relevante para la red neuronal, lo que llevaría a tener dificultades en la correcta clasificación.

Esto podría ser el resultado de la reducción en el número de características, que aunque puede haber mejorado el rendimiento general, también podría haber eliminado características importantes para la alteración Física.

Una observación relevante es la cantidad de características seleccionadas por los mejores cromosomas para cada caso. En comparación con el modelo original que utiliza 83 características, los cromosomas seleccionados utilizaron desde 41 a 70 características, como 20Q que obtuvo sus resultados con 61 características, 50Q con 70 y 40Q con 41. Esto indica una reducción en el número total de características utilizadas. La reducción puede ser beneficiosa al eliminar características irrelevantes o redundantes, contribuyendo a una mayor eficiencia del modelo y una menor tasa de falsos negativos.

Por lo tanto, los resultados obtenidos mediante la implementación del algoritmo genético como método de selección de características demuestran una mejora sustancial en la exactitud y la reducción de la tasa de falsos negativos en comparación con los resultados de la investigación original. Los mejores cromosomas por caso destacan aún más estas mejoras, evidenciando la capacidad del algoritmo genético para optimizar las características relevantes y mejorar el rendimiento del modelo de redes neuronales en la clasificación de datos sintéticos.

Tabla 12: Valor de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales para el mejor cromosoma por caso. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	1.000	0.956	0.978	0.912	0.989	1 %
30Q	0.945	0.879	0.978	0.956	1.000	1 %
40Q	0.956	0.956	0.978	0.923	0.989	1 %
50Q	0.989	0.945	0.978	0.923	0.967	1 %
60Q	0.989	0.912	0.978	0.923	0.945	1 %
AVG	0.975	0.929	0.978	0.927	0.978	

Tabla 13: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales para el mejor cromosoma por caso. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	2.0 %	7.3 %	2.1 %	5.7 %	2.5 %	1 %
30Q	4.4 %	7.0 %	2.1 %	6.5 %	2.8 %	1 %
40Q	6.2 %	7.3 %	2.4 %	6.8 %	4.2 %	1 %
50Q	6.9 %	7.5 %	2.0 %	6.8 %	5.7 %	1 %
60Q	7.3 %	8.1 %	2.2 %	7.0 %	6.7 %	1 %
AVG	5.4 %	7.4 %	2.2 %	6.6 %	4.4 %	

Ahora, aplicando una mutación del 10 % para promover la variabilidad genética, se dispone de las tablas 14 y 15 como los resultados obtenidos en la experimentación de esta imple-

mentación. En esta primera tabla, se puede observar que la exactitud promedio resultó en una ligera disminución para los distintos casos.

Por ejemplo, la exactitud promedio en el tipo Control disminuyó de 0.965 (con $p_m = 1\%$) a 0.956 (con $p_m = 10\%$). Un comportamiento similar se observa en las alteraciones Física y Cognitiva, donde los valores de exactitud también disminuyen. Sin embargo, en la perturbación Emocional, la exactitud se mantiene constante en 0.975, indicando que en este caso particular, el aumento en la probabilidad de mutación no tuvo un impacto negativo significativo.

Tabla 14: Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.973	0.913	0.977	0.929	0.975	10 %
30Q	0.953	0.918	0.975	0.927	0.959	10 %
40Q	0.941	0.911	0.975	0.928	0.951	10 %
50Q	0.955	0.930	0.974	0.928	0.964	10 %
60Q	0.957	0.928	0.975	0.924	0.959	10 %
AVG	0.956	0.920	0.975	0.927	0.962	

Respecto a la tasa de falsos negativos, la tabla 15 muestra que la tasa de falsos negativos sufrió variaciones al aumentar la probabilidad de mutación. En general, las tasas de falsos negativos son más altas con $p_m = 10\%$ en comparación con $p_m = 1\%$.

Tomando un caso en particular, la tasa de falsos negativos en el tipo Física aumentó de un promedio de 5.9 % a 7.3 %. En el tipo Control muestra un aumento similar, pasando de 2.9 % a 4.4 %. Este aumento en la tasa de falsos negativos sugiere que una mayor probabilidad de mutación puede introducir una variabilidad que, aunque puede explorar más el espacio de búsqueda, también puede llevar a una menor estabilidad y exactitud en la selección de características.

El aumento de la probabilidad de mutación a 10 % en el algoritmo genético no ha demostrado ser beneficioso en términos de mejorar la exactitud promedio de las redes neuronales para las distintas alteraciones. En distintas perturbaciones, la exactitud disminuyó ligeramente, como en el tipo Control, donde pasó de 0.965 con $p_m = 1\%$ a 0.956 con $p_m = 10\%$, y en el tipo Físico, donde disminuyó de 0.930 a 0.920. El tipo Cognitivo también experimentó una reducción en la exactitud, de 0.938 a 0.927, y el tipo Organizacional mostró una pequeña disminución, de 0.964 a 0.962. Además, la tasa de falsos negativos aumentó en varias perturbaciones. Por ejemplo, en el tipo Control, la tasa de falsos negativos subió de 2.9 % a 4.4 %, y en el tipo Físico, aumentó de 5.9 % a 7.3 %. En el tipo Cognitivo, la tasa de falsos negativos pasó de 5.8 % a 6.6 %, y en el tipo Organizacional, de 2.8 % a 3.5 %. Aunque una mayor variabilidad genética puede ser útil en algunos casos para evitar mínimos locales, en este caso particular, generó más ruido que beneficios en el proceso de selección de características.

Tabla 15: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	2.5 %	7.9 %	2.1 %	6.4 %	2.3 %	10 %
30Q	4.3 %	7.5 %	2.2 %	6.6 %	3.8 %	10 %
40Q	5.3 %	8.0 %	2.2 %	6.6 %	4.4 %	10 %
50Q	4.2 %	6.4 %	2.4 %	6.7 %	3.2 %	10 %
60Q	3.9 %	6.5 %	2.3 %	6.9 %	3.7 %	10 %
AVG	4.4 %	7.3 %	2.2 %	6.6 %	3.5 %	

Con el objetivo de observar cómo se comporta esta implementación con el aumento de la población y las generaciones desde 20 y 10 a 40 y 20 respectivamente, se dispone de las siguientes tablas que presentan el promedio de exactitud de las distintas evaluaciones y su tasa de negativos, como también el análisis de los resultados obtenidos por el mejor cromosoma de entre todas las generaciones.

Tabla 16: Promedio de exactitud para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.975	0.907	0.976	0.934	0.973	1 %
30Q	0.960	0.909	0.974	0.931	0.971	1 %
40Q	0.967	0.898	0.976	0.930	0.966	1 %
50Q	0.945	0.927	0.973	0.935	0.945	1 %
60Q	0.930	0.922	0.975	0.927	0.934	1 %
AVG	0.955	0.913	0.974	0.931	0.958	

Como se detalla en la tabla 16, al comparar los valores de exactitud con los de la tabla 10, se observa un aumento en los promedios o AVG de los tipos de perturbaciones Control, Cognitivo y Organizacional, pasando de 0.941, 0.928 y 0.958 a 0.955, 0.931 y 0.958 respectivamente. En cambio, para las perturbaciones de tipo Físico y Emocional, no se observó un aumento, obteniéndose valores de 0.913 y 0.974 en comparación con los valores previos de 0.918 y 0.976. Estas diferencias, aunque menores que las observadas en la primera comparación, indican que el aumento en el tamaño de la población y el número de generaciones puede tener un impacto positivo en ciertas perturbaciones, mientras que en otras el impacto es limitado o incluso negativo.

Al analizar la tasa de falsos negativos en la tabla 17 en comparación con la tabla 11, se observa una notable reducción en el caso 30Q para la perturbación Física, donde la tasa disminuyó del 11.2 % al 8.3 %, y el promedio general mejoró del 8.4 % al 8.0 %. Estas mejoras son destacables, junto con las observadas en el tipo Cognitivo, donde la tasa de falsos negativos pasó

de un promedio del 7.4 % al 6.2 %, y en el tipo Organizacional, que disminuyó del 5.1 % al 3.8 %.

Sin embargo, para la perturbación de tipo Control, la tasa de falsos negativos aumentó del 1.1 % al 4.1 %. Esta diferencia, aunque notable, puede ser atribuida al aumento de la población y las generaciones, llevó a que el modelo se enfoque más en ajustar otros tipos de perturbaciones, dejando menos capacidad para la clasificación precisa de los casos de tipo Control. En cuanto a la perturbación Emocional, el aumento del 0.2 % en la tasa de falsos negativos es considerado aceptable y no refleja un cambio significativo en el desempeño del modelo.

Tabla 17: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	2.3 %	8.5 %	2.2 %	6.0 %	2.5 %	1 %
30Q	3.6 %	8.3 %	2.4 %	6.3 %	2.6 %	1 %
40Q	3.0 %	9.3 %	2.2 %	6.2 %	3.1 %	1 %
50Q	5.2 %	6.7 %	2.4 %	6.0 %	4.9 %	1 %
60Q	6.4 %	7.1 %	2.3 %	6.6 %	6.0 %	1 %
AVG	4.1 %	8.0 %	2.3 %	6.2 %	3.8 %	

En una última observación de los resultados, al analizar los mejores cromosomas de la primera implementación con la menor población, se obtuvieron valores promedio de exactitud de 0.975, 0.929, 0.978, 0.927 y 0.978 para las perturbaciones de Control, Físico, Emocional, Cognitivo y Organizacional, respectivamente. En contraste, al aumentar la población y las generaciones, como se muestra en la tabla 18, estos promedios aumentan a 0.996, 0.976, 0.932 y 0.980 para Control, Emocional, Cognitivo y Organizacional, mientras que la exactitud para el tipo Físico se redujo a 0.919.

Aumentar la población y las generaciones puede haber introducido una mayor complejidad en el modelo, que podría haber afectado negativamente la exactitud en la clasificación del tipo Físico. Además, la naturaleza del tipo Físico puede hacer que sea más susceptible a las variaciones en el rendimiento, debido a la complejidad de las características asociadas a esta perturbación. En resumen, mientras que el aumento en población y generaciones ha mejorado la exactitud en la mayoría de los casos, la variabilidad en el tipo Físico sugiere que este tipo de perturbación podría beneficiarse de ajustes adicionales o de técnicas específicas para manejar su variabilidad.

La tasa de falsos negativos de esta implementación con mayor cantidad de generaciones y población con el mejor cromosoma, como se muestra en 19, muestra que para la perturbación de tipo Control, todos los casos, excepto 50Q, que obtuvo un 1.1 %, presentan una tasa de 0.0 %, indicando que no hubo presencia de falsos negativos en la mayoría de los casos.

Comparado con los resultados previos con menos población, que mostraban valores entre 2.0 % y 7.3 %, esta mejora es notable para el tipo Control.

En cuanto a las demás perturbaciones, se observa que para el tipo Físico hay un empeoramiento en su valor promedio de falsos negativos, aumentando del 7.4 % al 8.1 %. Para los tipos Emocional y Cognitivo, las variaciones son menores, con un aumento del 0.3 % y una disminución del 0.1 %, respectivamente. Por otro lado, el tipo Organizacional muestra una mejora significativa, reduciendo su tasa de falsos negativos del 4.4 % al 1.9 %.

Estos resultados sugieren que, aunque la implementación con mayor población y generaciones ha logrado una mejora sustancial en la exactitud para algunos tipos de perturbaciones, la tasa de falsos negativos en el tipo Físico ha aumentado, posiblemente debido a una mayor complejidad en la clasificación de este tipo específico. La mejora en el tipo Organizacional y la reducción en el tipo Control destacan la eficacia del aumento en la población y las generaciones en estos casos, mientras que las variaciones menores en Emocional y Cognitivo indican un impacto más equilibrado en estos tipos.

Tabla 18: Exactitud del mejor cromosoma para las distintas instancias de datos sintéticos mediante Redes Neuronales con mayor población y generaciones (Población 40, 20 Generaciones). Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.989	0.901	0.978	0.912	1.0	1 %
30Q	1.0	0.934	0.978	0.934	0.989	1 %
40Q	1.0	0.901	0.967	0.945	0.967	1 %
50Q	0.989	0.945	0.978	0.934	0.978	1 %
60Q	1.0	0.912	0.978	0.934	0.967	1 %
AVG	0.996	0.919	0.976	0.932	0.980	

Tabla 19: Falsos negativos para los mejores cromosomas en las distintas instancias de datos sintéticos (Población 40, 20 Generaciones). Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.0 %	9.8 %	2.3 %	8.7 %	0.0 %	1 %
30Q	0.0 %	6.5 %	2.2 %	6.6 %	1.1 %	1 %
40Q	0.0 %	9.8 %	3.3 %	5.5 %	3.3 %	1 %
50Q	1.1 %	5.5 %	2.6 %	6.5 %	2.2 %	1 %
60Q	0.0 %	8.7 %	2.2 %	6.5 %	3.3 %	1 %
AVG	0.2 %	8.1 %	2.5 %	6.7 %	1.9 %	

4.1.2. Primera implementación - Random Forest

Las variaciones aplicadas para esta implementación fueron las siguientes:

- Número de generaciones: 10 - 20
- Tamaño de población: 20 - 40
- Selección de padres: Selección por ruleta
- Cruzamiento: Cruzamiento en un punto.
- Probabilidad de mutación utilizada: 1 % y 10 %.
- Reemplazo de población: Reemplazo total.

Observando los resultados de los resultados originales para *Random Forest* en la tabla 8, se aprecia un rendimiento alto en la exactitud promedio para todas las instancias de datos sintéticos. Los valores resultados oscilaron entre 0.923 y 1.000, con un promedio general de 0.947. En particular, los datos del conjunto 40Q y 50Q presentaron las mejores exactitudes, alcanzando 1.000 y 0.987, respectivamente. Este rendimiento superior indica que el modelo de Random Forest es muy eficaz para las características de los datos sintéticos utilizados en estos casos, especialmente en las dimensiones Emocional, Cognitiva y Organizacional.

Por otro lado, los resultados obtenidos con la primera implementación del algoritmo genético, como se muestra en la tabla 20, la exactitud promedio varía entre 0.389 y 0.864, con un promedio general de 0.739. Esta baja de la métrica es más notoria en la dimensión Física, donde esta se reduce desde 0.923 hasta 0.389. También se observan reducciones en las demás métricas, tomando el promedio de cada una para el caso 20Q por poner un caso en particular, se tiene que desde 0.980, 0.933, 0.923 y 0.850 para el tipo Contro, Emocional, Cognitivo y Organizacional, se redujo hasta 0.717, 0.737, 0.669 y 0.822 respectivamente.

Esta significativa reducción en la exactitud con la aplicación del AG sugiere que la variabilidad genética no resultó beneficiosa, sugirieron que al remover características del modelo llevó a una mayor dificultad para poder discriminar entre las diferentes clases y afectando la exactitud.

Esta situación también se ve reflejada en la tasa de falsos negativos, donde en los resultados mostrados en la tabla 11 se observan valores por sobre el 10 % para el tipo Control, sobre el 54 % para la perturbación Física y sobre 30 % para la Emocional. Es bueno notar que para la perturbación Cognitiva y Organizacional, este valor se mantuvo cercano lo mostrado en la tabla 9 con valores de 2.1 % y 5.6 % frente a 2.3 % y 4.7 % respectivamente.

Tabla 20: Promedio de exactitud para las distintas instancias de datos sintéticos mediante *Random Forest*. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.717	0.389	0.737	0.669	0.822	1 %
30Q	0.734	0.438	0.555	0.671	0.815	1 %
40Q	0.737	0.478	0.654	0.669	0.838	1 %
50Q	0.745	0.564	0.727	0.667	0.845	1 %
60Q	0.760	0.657	0.864	0.670	0.860	1 %
AVG	0.739	0.505	0.707	0.669	0.836	

Tabla 21: Tasa de falsos negativos para las distintas instancias de datos sintéticos mediante *Random Forest*. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	10.8 %	59.4 %	18.6 %	1.7 %	5.3 %	1 %
30Q	10.4 %	57.4 %	34.1 %	1.9 %	5.9 %	1 %
40Q	9.4 %	56.3 %	38.7 %	2.3 %	6.6 %	1 %
50Q	10.4 %	54.0 %	40.3 %	2.1 %	6.2 %	1 %
60Q	9.9 %	49.8 %	34.7 %	2.8 %	4.1 %	1 %
AVG	10.0 %	55.4 %	33.3 %	2.1 %	5.6 %	

El incremento de la población y generaciones no resultó en ninguna mejora notoria en las métricas evaluadas de exactitud y tasa de falsos negativos. Esto se puede ver en la tabla 22, donde los promedios son similares a la primera implementación con menor población y generaciones. Tomando la perturbación Física, Emocional, Cognitiva y Organizacional, se obtuvieron valores de 0.527, 0.701, 0.672 y 0.835 respectivamente frente a los valores de 0.505, 0.707, 0.669 y 0.836 de la tabla anterior de exactitud.

Tabla 22: Promedio de exactitud para las distintas instancias de datos sintéticos mediante *Random Forest* para más generaciones y tamaño de población. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	0.734	0.403	0.721	0.674	0.821	1 %
30Q	0.754	0.460	0.550	0.673	0.819	1 %
40Q	0.758	0.514	0.637	0.671	0.831	1 %
50Q	0.747	0.601	0.740	0.670	0.847	1 %
60Q	0.761	0.658	0.857	0.670	0.859	1 %
AVG	0.751	0.527	0.701	0.672	0.835	

Tabla 23: Tasa de Falsos Negativos para más generaciones y tamaño de población. Fuente: Elaboración propia.

Data-set	Control	Físico	Emocional	Cognitivo	Organizacional	p_m
20Q	8.1 %	57.6 %	20.6 %	1.9 %	5.3 %	1 %
30Q	8.7 %	55.9 %	32.9 %	1.8 %	5.7 %	1 %
40Q	7.4 %	55.1 %	38.8 %	2.0 %	7.1 %	1 %
50Q	7.7 %	51.3 %	41.0 %	2.1 %	5.9 %	1 %
60Q	8.7 %	45.7 %	30.6 %	2.2 %	4.5 %	1 %
AVG	8.1 %	53.1 %	32.8 %	2.0 %	5.7 %	

4.1.3. Tiempos de ejecución

Luego de realizar las ejecuciones de cada implementación y variar la cantidad de iteraciones, se obtuvieron las tablas 24, 25, 26, y 27. Estas tablas presentan los tiempos de ejecución correspondientes a las implementaciones de los clasificadores NN y RF, bajo configuraciones de 10 y 20 generaciones con 20 individuos, así como de 20 generaciones con 40 individuos en la población.

En la configuración con 10 generaciones y una población de 20 individuos para las redes neuronales, el tiempo promedio de ejecución fue de 5200.89 segundos (cerca de 1.4 horas). Por otro lado, al aumentar el número de generaciones a 20 y el tamaño de la población a 40 individuos, el tiempo promedio de ejecución se incrementó significativamente a 17309.75 segundos (4.8 horas). Esto indica que la configuración con más generaciones y una mayor población resulta en tiempos de ejecución considerablemente más largos.

Además, se evaluó el impacto de las modificaciones en el código sin utilizar el AG, pero con la reducción de atributos, es decir, con el mejor cromosoma obtenido. Al usar menos atributos, el tiempo promedio de ejecución fue de 26.12 segundos. En contraste, utilizando todos los atributos, el tiempo promedio fue de 26.87 segundos. Esta reducción en el número de atributos resultó en una mejora del 2.77 % en el tiempo de ejecución, demostrando una optimización en los tiempos que tomó la clasificación.

Tabla 24: Tiempos de ejecución para NN, menor cantidad de generaciones y población. Fuente: Elaboración propia.

Data-set	Número de generaciones	Tamaño población	Tiempo de ejecución [s]
20Q	10	20	5186.22
30Q	10	20	5210.47
40Q	10	20	5159.35
50Q	10	20	5245.89
60Q	10	20	5202.54
AVG	-	-	5200.89

Tabla 25: Tiempos de ejecución para NN, mayor cantidad de generaciones y población. Fuente: Elaboración propia.

Data-set	Número de generaciones	Tamaño población	Tiempo de ejecución [s]
20Q	20	40	17280.00
30Q	20	40	17345.78
40Q	20	40	17205.31
50Q	20	40	17402.56
60Q	20	40	17315.12
AVG	-	-	17309.75

Para la implementación con RF, con 10 generaciones y una población de 20 individuos, los tiempos oscilan entre 157.52 y 162.78 segundos, promediando 160.73 segundos. Esta configuración es más eficiente debido a su menor complejidad. En cambio, al incrementar estos dos parámetros, los tiempos aumentan significativamente, resultando en un rango desde 672.45 a 685.11 segundos, con un promedio de 678.52 segundos. Esto debido la mayor carga computacional que requiere realizar las múltiples evaluaciones de cada individuo. Aunque este ajuste puede mejorar la exploración del espacio de soluciones, también incrementa considerablemente el tiempo de ejecución.

Tabla 26: Tiempos de ejecución para RF, 10 generaciones y 20 de población. Fuente: Elaboración propia.

Data-set	Número de generaciones	Tamaño población	Tiempo de ejecución [s]
20Q	10	20	159.34
30Q	10	20	162.78
40Q	10	20	157.52
50Q	10	20	160.11
60Q	10	20	161.90
AVG	-	-	160.73

Tabla 27: Tiempos de ejecución para RF, 20 generaciones y 40 de población. Fuente: Elaboración propia.

Data-set	Número de generaciones	Tamaño población	Tiempo de ejecución [s]
20Q	20	40	678.23
30Q	20	40	685.11
40Q	20	40	672.45
50Q	20	40	680.92
60Q	20	40	675.87
AVG	-	-	678.52

4.2. Caso de estudio Human Activity Recognition

En esta sección de resultados, se presenta un análisis sobre el rendimiento de diversos modelos de clasificación y algoritmos de selección de características, aplicados al caso de reconocimiento de actividades humanas (HAR). Para este estudio, se evaluaron los modelos SVM, LDA, KNN y RF utilizando diferentes métodos de selección de características, tales como BORUTA, RFE y RF, cada uno con una cantidad de atributos definida. Específicamente, se hace referencia a las métricas de recall por cada clase y exactitud obtenidas en la clasificación, según el caso de estudio [Chen *et al.*, 2020]. En la tabla 28 se detallan los valores de recall para cada combinación de modelo y método de selección de características, mientras que la tabla 29 resume los resultados de exactitud bajo las mismas condiciones. Estos resultados permiten comparar la efectividad de los distintos enfoques en la mejora del rendimiento de los modelos de clasificación en el contexto de HAR.

Para la última comparación con clasificador NB con características seleccionadas por k-means, se dispone de la tabla 30 para hacer la comparación por exactitud en diferentes cantidades de atributos haciendo uso del algoritmo genético.

Tabla 28: Métrica de recall por clase de diferentes clasificadores con reconocimiento de actividad humana usando 6 atributos. Fuente: [Chen *et al.*, 2020].

Evaluación	Clase: LAYING	Clase: SITTING	Clase: STANDING	Clase: WALKING	Clase: WALKING DOWNSTAIRS	Clase: WALKING UPSTAIRS
RF + SVM						
Recall	1.0000	0.8872	0.8650	0.8571	0.7919	0.7617
RF + LDA						
Recall	1.0000	0.7354	0.9051	0.8408	0.8223	0.61682
RF + KNN						
Recall	1.0000	0.9339	0.9161	0.8776	0.8426	0.8131
RF + RF						
Recall	1.0000	0.9455	0.9270	0.9020	0.8985	0.9019

Tabla 29: Exactitud de clasificación de diferentes clasificadores con reconocimiento de actividad humana. Fuente: [Chen *et al.*, 2020].

Método	Exactitud	N_atributos
SVM	0.9796	561
Boruta + SVM	0.8692	6
RFE + SVM	0.8331	6
RF + SVM	0.8685	6
LDA	0.9823	561
Boruta + LDA	0.7786	6
RFE + LDA	0.705	6
RF + LDA	0.8297	6
KNN	0.9748	561
Boruta + KNN	0.864	6
RFE + KNN	0.8385	6
RF + KNN	0.904	6
RF	0.9857	561
Boruta + RF	0.8924	6
RFE + RF	0.9394	6
RF + RF	0.9326	6

Tabla 30: Rendimiento del clasificador Naive Bayes utilizando el conjunto de datos HAR con características seleccionadas por K-means. Fuente: [Ismi *et al.*, 2016].

N_atributos	Exactitud
10	69.40
20	76.81
30	79.56
40	79.33
50	81.38
60	79.84
70	83.35

4.2.1. Primera implementación

Las variaciones aplicadas para esta primera implementación fueron las siguientes:

- Número de generaciones: 10 - 20
- Tamaño de población: 20 - 40
- Selección de padres: Selección lineal por rango
- Cruzamiento: Cruzamiento Uniforme.
- Probabilidad de mutación utilizada: 1 %, 5 % y 10 %.
- Reemplazo de población: Reemplazo total.

En la tabla 31, se presenta la mediana de exactitud de distintos métodos de clasificación y el mejor valor de esta métrica entre paréntesis, utilizando esta primera implementación del AG con diferentes probabilidades de mutación (P_m):

Tabla 31: Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, primera implementación. Fuente: Elaboración propia.

Evaluación	$P_m = 1\%$	$P_m = 5\%$	$P_m = 10\%$	N_atributos
AG+KNN	0.791 (0.838)	0.812 (0.858)	0.785 (0.831)	6
AG+RF	0.827 (0.867)	0.842 (0.845)	0.851 (0.871)	6
AG+SVM	0.826 (0.873)	0.824 (0.883)	0.832 (0.861)	6
AG+LDA	0.787 (0.843)	0.788 (0.849)	0.785 (0.813)	6
AG+NB	0.806 (0.842)	0.796 (0.856)	0.794 (0.840)	10
AG+NB	0.814 (0.856)	0.806 (0.844)	0.819 (0.847)	20
AG+NB	0.827 (0.839)	0.824 (0.851)	0.826 (0.848)	30
AG+NB	0.828 (0.844)	0.832 (0.849)	0.828 (0.847)	40
AG+NB	0.828 (0.846)	0.831 (0.853)	0.831 (0.846)	50
AG+NB	0.828 (0.840)	0.830 (0.848)	0.831 (0.852)	60
AG+NB	0.832 (0.847)	0.829 (0.841)	0.833 (0.844)	70

Al comparar la tabla 31 con la tabla 29 se puede realizar la comparación de la exactitud en los distintos modelos aplicados.

Para el algoritmo KNN, la exactitud se reduce a 0.904 cuando se utiliza el método RF para la selección de características. En el caso del algoritmo genético (AG) combinado con KNN, la exactitud es de 0.812 con una probabilidad de mutación del 5 %. Al analizar cada una de las ejecuciones, el valor máximo alcanzado en 10 iteraciones del algoritmo es una exactitud de 0.857. Por lo tanto, se puede concluir que el uso de AG con KNN presenta un rendimiento mejor que el de RFE y cercano al de Boruta en la selección de características, aunque inferior al de RF. Esto sugiere que, aunque el algoritmo genético puede mejorar la exactitud de KNN en comparación con RFE, no es tan efectivo como el método de selección de características basado en RF. Esto debido a que RF tiene una menor variación en su calidad a costo de computación.

En cuanto a la selección de características mediante Bosque Aleatorio (RF), los resultados obtenidos con el algoritmo genético (AG) fueron inferiores en términos de exactitud. Sin embargo, al aumentar la probabilidad de mutación, la calidad de los resultados mejoró, lo que indica que la mayor diversidad en la búsqueda de soluciones evita caer en óptimos locales. En general, AG tuvo un peor desempeño que RF en la selección de características.

Respecto a AG+LDA, este muestra un rendimiento más estable, alcanzando una exactitud de 0.788 con una probabilidad de mutación del 5 %, superando la calidad obtenida por RFE+LDA con 0.705 y Boruta+LDA con 0.7786. No obstante, RF+LDA volvió a presentar el mejor desempeño. Esto indica que, aunque el AG puede proporcionar mejoras significativas en el rendimiento de LDA en comparación con RFE y Boruta, el método RF mantiene un mejor desempeño.

Finalmente, al aplicar SVM, se observa el mejor desempeño en Boruta+SVM y RF+SVM con

valores de 0.8692 y 0.8685 respectivamente, mientras que AG+SVM obtuvo exactitudes de 0.826, 0.824 y 0.832 para los diferentes porcentajes de mutación. Aunque no mostró una mejora numérica significativa, mantuvo un desempeño similar a los demás métodos. Considerando las exactitudes por iteración, uno de los resultados más altos es 0.8827 con una probabilidad de mutación del 5 %, lo que indica un desempeño constante. La ligera inferioridad del AG es debido a la aleatoriedad en la selección de características, que no siempre captura las relaciones más importantes entre las variables, como lo hacen los métodos más estructurados como Boruta y RF que evalúan la importancia de los atributos, seleccionando así los caminos más relevantes.

Respecto a la métrica de recall, la tabla 28 entrega los resultados obtenidos por la investigación, y la tabla 32 representa los valores obtenidos por cada clase con la implementación del AG utilizando cada clasificador anterior.

Tabla 32: Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, Fuente: Elaboración propia,

Evaluación	Clase: LAYING	Clase: SITTING	Clase: STANDING	Clase: WALKING	Clase: WALKING DOWNSTAIRS	Clase: WALKING UPSTAIRS
AG + SVM						
$P_m = 1\%$	0.8208	0.7724	0.4849	0.7961	0.7564	0.7600
$P_m = 5\%$	0.8474	0.7737	0.4909	0.8339	0.7497	0.7366
$P_m = 10\%$	0.7981	0.8148	0.5020	0.8778	0.7674	0.7774
AG + LDA						
$P_m = 1\%$	0.7977	0.7364	0.5164	0.7204	0.7743	0.7228
$P_m = 5\%$	0.8291	0.7061	0.4524	0.6809	0.7688	0.7057
$P_m = 10\%$	0.8758	0.7160	0.4499	0.7152	0.7379	0.7472
AG + KNN						
$P_m = 1\%$	0.6609	0.7220	0.4576	0.5048	0.6629	0.6764
$P_m = 5\%$	0.6696	0.7105	0.5048	0.5555	0.6637	0.6901
$P_m = 10\%$	0.7023	0.7499	0.5060	0.6040	0.6493	0.7075
AG + RF						
$P_m = 1\%$	0.7227	0.7124	0.5928	0.7040	0.7472	0.7416
$P_m = 5\%$	0.7121	0.7507	0.5454	0.6431	0.7304	0.7365
$P_m = 10\%$	0.7176	0.7352	0.6014	0.6994	0.7430	0.7397

En la tabla 28, RF combinado con otros clasificadores tiende a tener un recall más alto y consistente a través de todas las clases. Esto sugiere que RF es particularmente efectivo para capturar y distinguir las características relevantes para la clasificación de actividades humanas. Respecto a la implementación del AG, se observa en la tabla 32 que los valores de recall varían con diferentes probabilidades de mutación. Las mejores probabilidades de mutación se encuentran alrededor del 10 %, aunque todavía no alcanzan el rendimiento de los modelos de la tabla 28. Los modelos basados en algoritmos genéticos muestran mayor variabilidad en

su desempeño, lo que puede atribuirse al uso de los operadores genéticos como la mutación y el cruzamiento, que introducen variaciones aleatorias en el proceso de optimización.

Para ofrecer un mayor análisis de resultados con una mayor población y número de generaciones, se dispone de la tabla 33 que detalla el comportamiento de las métricas con una mayor cantidad de generaciones y una población más grande.

Tabla 33: Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, primera implementación con mayor población y generaciones. Fuente: Elaboración propia.

Evaluación	$P_m = 1\%$	$P_m = 5\%$	$P_m = 10\%$	N_atributos
AG+KNN	0.825 (0.865)	0.818 (0.866)	0.819 (0.847)	6
AG+RF	0.859 (0.881)	0.842 (0.845)	0.851 (0.871)	6
AG+SVM	0.853 (0.864)	0.830 (0.853)	0.837 (0.847)	6
AG+LDA	0.814 (0.830)	0.804 (0.828)	0.812 (0.839)	6
AG+NB	0.821 (0.848)	0.811 (0.843)	0.812 (0.856)	10
AG+NB	0.826 (0.838)	0.839 (0.863)	0.840 (0.858)	20
AG+NB	0.840 (0.861)	0.844 (0.853)	0.835 (0.847)	30
AG+NB	0.842 (0.849)	0.841 (0.851)	0.840 (0.850)	40
AG+NB	0.845 (0.864)	0.844 (0.860)	0.835 (0.856)	50
AG+NB	0.841 (0.853)	0.841 (0.849)	0.841 (0.858)	60
AG+NB	0.840 (0.852)	0.830 (0.852)	0.835 (0.854)	70

En la tabla 33, AG+RF alcanza la mayor mediana de exactitud (0.859) con una probabilidad de mutación del p_m del 1 %, y también muestra el mejor valor de exactitud (0.881) en el mismo contexto. AG+KNN presenta una mediana de exactitud que varía entre 0.818 y 0.825, con los mejores valores de exactitud oscilando entre 0.847 y 0.866. AG+SVM muestra una mediana de exactitud que varía de 0.830 a 0.853, con los mejores valores de exactitud entre 0.847 y 0.864. AG+LDA presenta una mediana de exactitud menor, oscilando entre 0.804 y 0.814, con los mejores valores de exactitud alcanzando hasta 0.839.

Al comparar los resultados de la tabla 31 con los de la tabla 33, notamos varias diferencias y similitudes en el rendimiento de los métodos de clasificación al variar la población y el número de generaciones en el algoritmo genético.

AG+KNN muestra una mejora significativa en la mediana de exactitud en la tabla 33, alcanzando 0.825 frente a 0.791 con p_m del 1 %. AG+RF mantiene una mediana de exactitud similar entre ambos contextos, con un pequeño incremento en la tabla 33 (0.859 frente a 0.827 con p_m del 1 %). AG+SVM presenta una mejora en la mediana de exactitud en la tabla 33, con 0.853 frente a 0.826 con p_m del 1 %. AG+LDA también muestra una mediana de exactitud mejorada, alcanzando 0.814 frente a 0.787 con p_m del 1 %.

Para AG+NB, la mediana de exactitud y los mejores valores en ambos contextos presentó

una leve mejora, como es el caso de AG+B con 10 atributos, donde inicialmente se obtuvo un 0.806 de mediana de exactitud y con una mayor cantidad de población y generaciones, esto incrementó a 0.821. Este tipo de comportamiento se reflejó en las otras instancias. Esto indica que el aumento de evaluaciones mejora la exactitud en la clasificación.

En los mejores valores de exactitud, AG+RF muestra el mejor rendimiento con un valor de exactitud de 0.881. AG+KNN y AG+SVM presentan mejoras en los valores de exactitud mayores en esta nueva iteración con mayor población y generaciones. AG+LDA también mejora en términos del mejor valor de exactitud, con 0.839 frente a 0.843.

Un aspecto importante a considerar es la diferencia entre la mediana y el mejor valor de exactitud. Una menor diferencia entre estos valores indica una menor variación en los datos, lo cual sugiere una mayor consistencia en el rendimiento del modelo. En la tabla 33, AG+RF muestra una diferencia relativamente pequeña entre la mediana (0.859) y el mejor valor (0.881) con p_m del 1 %, lo que sugiere una alta consistencia en su rendimiento. En cambio, AG+KNN presenta una diferencia mayor en la tabla 31 con p_m del 1 %, lo que indica una mayor variabilidad en los resultados obtenidos.

Comparando con los resultados presentados en la tabla 29, se observa que RF sigue siendo el modelo que ofrece los mejores resultados con 6 atributos. Sin embargo, la aplicación del AG como método de selección de características no logró alcanzar resultados similares o superiores. El mejor caso obtenido de 0.881 con la combinación AG+RF al 1 % de p_m contrasta con el valor de 0.9326 obtenido por RF+RF, indicando un desempeño inferior para AG en este caso.

En cuanto a los otros métodos, AG+KNN logró un mejor resultado con una exactitud de 0.866 al 5 % de p_m , en comparación con los valores de 0.8385 y 0.864 obtenidos por RFE+KNN y Boruta+KNN, respectivamente. Por otro lado, AG+LDA superó a Boruta+LDA y RFE+LDA con un valor máximo de 0.839 al 10 % de p_m , frente a los valores de 0.7786 y 0.705 obtenidos por estos métodos. Sin embargo, RF+LDA, con un valor de 0.8279, ha demostrado un mejor rendimiento general. A pesar de esto, AG+LDA mostró un valor de exactitud superior, tanto al 10 % de mutación con 0.839 como al 1 % con 0.830. Esta ligera mejora en la exactitud sugiere que la variabilidad puede contribuir a encontrar un valor óptimo superior haciendo uso de los operadores genéticos de AG. Por último, AG+SVM obtuvo el mejor resultado con una exactitud de 0.864 al 1 % de probabilidad de mutación, superando a RFE+SVM, que alcanzó 0.8331. No obstante, no superó a Boruta+SVM y RF+SVM, que lograron exactitudes de 0.8692 y 0.8685.

El análisis de los resultados revela que AG+RF destaca como el mejor método con una mediana de exactitud de 0.859 y un mejor valor de 0.881 al 1 % de p_m . Este método muestra la mayor consistencia en su rendimiento, con una pequeña diferencia entre la mediana y el mejor valor de exactitud. Los métodos AG+KNN y AG+SVM también muestran mejoras notables, con AG+KNN alcanzando una mediana de 0.825 y AG+SVM una mediana de 0.853, ambos superiores a sus versiones anteriores. AG+LDA, aunque mejora respecto a contextos anteriores, presenta valores más bajos en comparación con AG+RF.

Luego de revisar los resultados, RF sigue siendo el modelo con mejor rendimiento con 6 atributos. Sin embargo, el AG, especialmente en su combinación con RF, no logra igualar el rendimiento de RF+RF, pero ayuda a mejorar la exactitud en otros métodos de clasificación como LDA. AG+KNN y AG+LDA muestran mejoras respecto a métodos de selección tradicionales como RFE y Boruta, aunque AG+SVM, a pesar de sus mejoras, no supera a Boruta+SVM y RF+SVM. Estos resultados sugieren que, aunque AG puede mejorar la exactitud en ciertos contextos, los métodos clásicos como RF siguen siendo más robustos en términos de rendimiento general.

Al analizar los resultados de recall en la tabla 34 muestra mejoras notables en varios métodos de clasificación respecto a la tabla 32 cuando se aumenta la población y el número de generaciones en la primera implementación. El método AG+SVM destaca por un significativo aumento en el recall en todas las clases, especialmente con un 1 % de p_m . Por ejemplo, el recall en la clase *LAYING* incrementa de 0.8208 a 0.9560, y en la clase *WALKING* de 0.7961 a 0.9069. Estos resultados sugieren que AG+SVM ha logrado un desempeño más robusto y consistente con una mayor capacidad computacional.

En contraste, AG+LDA presenta un desempeño más heterogéneo. Aunque se observa una mejora en algunas clases, como en *LAYING* donde el recall sube de 0.7977 a 0.9316, otras clases muestran una disminución, como en *WALKING*, que baja de 0.7204 a 0.7047. Esto indica que el aumento en la población y generaciones no beneficia de manera uniforme a todas las clases con AG+LDA, y su desempeño es variable dependiendo del contexto.

El método AG+KNN muestra mejoras más modestas en el recall. Aunque se observa un aumento en la clase *LAYING*, que pasa de 0.6609 a 0.6888 con p_m del 1 %, y en la clase *WALKING* de 0.5048 a 0.5931, las mejoras no son tan marcadas como las observadas con AG+SVM. Esta tendencia sugiere que, aunque hay una ligera mejora en el recall para clases menos representadas, AG+KNN no alcanza los mismos niveles de desempeño que AG+SVM.

Finalmente, AG+RF también muestra una mejora en el recall, aunque en menor cantidad en comparación con otros métodos. Por ejemplo, el recall en la clase *LAYING* incrementa de 0.7227 a 0.6464 con p_m del 1 %, y en la clase *WALKING* de 0.7040 a 0.6737. Aunque hay una tendencia de mejora, los aumentos son relativamente modestos, lo que sugiere que AG+RF sigue siendo competitivo, pero no alcanza el mismo nivel de mejora significativo observado en AG+SVM.

Tabla 34: Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, primera implementación con mayor población y generaciones. Fuente: Elaboración propia.

Evaluación	Clase: LAYING	Clase: SITTING	Clase: STANDING	Clase: WALKING	Clase: WALKING DOWNSTAIRS	Clase: WALKING UPSTAIRS
AG + SVM						
$P_m = 1\%$	0.9560	0.8390	0.5991	0.9069	0.7497	0.8406
$P_m = 5\%$	0.9167	0.8107	0.5847	0.8955	0.7411	0.7626
$P_m = 10\%$	0.9021	0.8188	0.5346	0.9118	0.7406	0.8568
AG + LDA						
$P_m = 1\%$	0.9316	0.6943	0.4754	0.7047	0.7712	0.7003
$P_m = 5\%$	0.8519	0.6933	0.4470	0.7088	0.7509	0.7466
$P_m = 10\%$	0.8896	0.7082	0.5491	0.7747	0.7557	0.7068
AG + KNN						
$P_m = 1\%$	0.6888	0.7078	0.5149	0.5931	0.6609	0.6690
$P_m = 5\%$	0.7017	0.7223	0.4911	0.5305	0.6379	0.6960
$P_m = 10\%$	0.7432	0.7421	0.5049	0.5938	0.6636	0.6957
AG + RF						
$P_m = 1\%$	0.6464	0.6620	0.5953	0.6737	0.7232	0.7248
$P_m = 5\%$	0.6740	0.7373	0.5382	0.6277	0.7265	0.7419
$P_m = 10\%$	0.7354	0.7628	0.5669	0.6947	0.7465	0.7860

4.2.2. Segunda implementación

Las variaciones aplicadas para esta primera implementación fueron las siguientes:

- Número de generaciones: 10 - 20
- Tamaño de población: 20 - 40
- Selección de padres: Selección lineal por rango.
- Cruzamiento: Cruzamiento aleatorio múltiple.
- Probabilidad de mutación utilizada: 1 %, 5 % y 10 %.
- Reemplazo de población: Reemplazo total.

En la tabla 35, se presenta la mediana de exactitud de distintos métodos de clasificación y el mejor valor de esta métrica entre paréntesis, utilizando esta primera implementación del AG con diferentes probabilidades de mutación (P_m):

Tabla 35: Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, segunda implementación. Fuente: Elaboración propia.

Evaluación	$P_m = 1\%$	$P_m = 5\%$	$P_m = 10\%$	N_atributos
AG+KNN	0.786 (0.829)	0.795 (0.825)	0.793 (0.886)	6
AG+RF	0.820 (0.842)	0.828 (0.848)	0.817 (0.858)	6
AG+SVM	0.803 (0.886)	0.817 (0.842)	0.792 (0.844)	6
AG+LDA	0.794 (0.831)	0.788 (0.840)	0.791 (0.809)	6
AG+NB	0.789 (0.837)	0.786 (0.836)	0.789 (0.830)	10
AG+NB	0.812 (0.850)	0.808 (0.837)	0.812 (0.845)	20
AG+NB	0.831 (0.852)	0.829 (0.855)	0.825 (0.839)	30
AG+NB	0.830 (0.843)	0.828 (0.844)	0.835 (0.849)	40
AG+NB	0.828 (0.840)	0.824 (0.839)	0.822 (0.838)	50
AG+NB	0.829 (0.852)	0.818 (0.834)	0.826 (0.862)	60
AG+NB	0.823 (0.843)	0.822 (0.844)	0.826 (0.845)	70

Realizando la comparación con la tabla 31 de la primera implementación, para el método de KNN, se observa que la primera implementación con cruzamiento uniforme muestra una mejora en la exactitud al incrementar la probabilidad de mutación del 1 % al 5 %, alcanzando un valor máximo de 0.812 (0.858). Sin embargo, al aumentar P_m al 10 %, la exactitud disminuye ligeramente. En la segunda implementación con cruzamiento aleatorio múltiple, la exactitud es más consistente, pero no supera los valores obtenidos con cruzamiento uniforme.

Para el método de RF, la primera implementación con cruzamiento uniforme muestra una exactitud creciente con el aumento de P_m , alcanzando un valor máximo de 0.851 (0.871) con $P_m=10\%$. En contraste, la segunda implementación con cruzamiento aleatorio múltiple presenta valores de exactitud de 0.820 (0.842) con $P_m=1\%$, 0.828 (0.848) con $P_m=5\%$ y 0.817 (0.858) con $P_m=10\%$. Estos resultados indican un comportamiento variable y no alcanzan la misma efectividad que la primera implementación.

En el caso del método SVM, la primera implementación con cruzamiento uniforme presenta una alta exactitud en todos los valores de P_m , especialmente notable con $P_m=5\%$ (0.824 (0.883)). La segunda implementación muestra valores de exactitud de 0.803 (0.886) con $P_m=1\%$, 0.817 (0.842) con $P_m=5\%$ y 0.792 (0.844) con $P_m=10\%$, lo que indica un rendimiento inferior en comparación con la primera implementación.

Para el método LDA, ambas implementaciones muestran una tendencia similar en cuanto a la exactitud. En la primera implementación, los valores son 0.787 (0.843) con $P_m=1\%$, 0.788 (0.849) con $P_m=5\%$ y 0.785 (0.813) con $P_m=10\%$. En la segunda implementación, los valores son 0.794 (0.831) con $P_m=1\%$, 0.788 (0.840) con $P_m=5\%$ y 0.791 (0.809) con $P_m=10\%$. Los resultados son similares, con una ligera ventaja para la primera implementación con cruzamiento uniforme.

El método NB para la primera implementación con cruzamiento uniforme muestra una lige-

ra superioridad en los valores de exactitud en comparación con la segunda implementación. Por ejemplo, con 10 atributos y $P_m=1\%$, la exactitud es 0.806 (0.842), mientras que en la segunda implementación, con cruzamiento aleatorio múltiple, es 0.789 (0.837). Además, con 30 atributos y $P_m=1\%$, la primera implementación alcanza 0.827 (0.839), mientras que la segunda muestra 0.831 (0.852), indicando que los resultados son bastante competitivos, pero aún muestran una ligera ventaja para la primera implementación en algunos casos.

En general, la implementación con cruzamiento uniforme tiende a mostrar una mejor exactitud en la mayoría de los métodos de clasificación, especialmente en KNN, RF y SVM. El cruzamiento aleatorio múltiple no logra mantener la misma consistencia en los resultados de exactitud, indicando que el cruzamiento uniforme puede ser una estrategia más efectiva en este contexto específico de algoritmos genéticos aplicados a distintos métodos de clasificación

Tabla 36: Recall por clase de distintos métodos de clasificación para distintas probabilidades de mutación utilizando 6 atributos, segunda implementación. Fuente: Elaboración propia.

Evaluación	Clase: LAYING	Clase: SITTING	Clase: STANDING	Clase: WALKING	Clase: WALKING DOWNSTAIRS	Clase: WALKING UPSTAIRS
AG + SVM						
$P_m = 1\%$	0.8902	0.7604	0.4881	0.7736	0.7399	0.7390
$P_m = 5\%$	0.9253	0.7885	0.5173	0.8187	0.7722	0.7464
$P_m = 10\%$	0.8961	0.7749	0.5343	0.8587	0.7151	0.7258
AG + LDA						
$P_m = 1\%$	0.8993	0.6833	0.4082	0.6483	0.7301	0.7031
$P_m = 5\%$	0.7951	0.7045	0.4572	0.6793	0.7476	0.6765
$P_m = 10\%$	0.8044	0.7061	0.4066	0.6873	0.7386	0.7097
AG + KNN						
$P_m = 1\%$	0.6617	0.7076	0.4732	0.5051	0.6393	0.6803
$P_m = 5\%$	0.7159	0.7476	0.5083	0.5983	0.6511	0.6920
$P_m = 10\%$	0.6550	0.7393	0.4422	0.5031	0.6564	0.7274
AG + RF						
$P_m = 1\%$	0.7213	0.7119	0.5934	0.7036	0.7476	0.7412
$P_m = 5\%$	0.7117	0.7503	0.5460	0.6427	0.7308	0.7361
$P_m = 10\%$	0.7180	0.7348	0.6020	0.6990	0.7434	0.7393

Realizando el análisis del valor de recall de la tabla 36 se puede determinar que el uso de la segunda implementación tuvo efectos variados. Por ejemplo, para SVM Y LDA, el recall para la clase LAYING con SVM aumenta de 0.8208 en la primera implementación con cruzamiento uniforme a 0.8902 en la segunda implementación con cruzamiento múltiple aleatorio a $P_m=1\%$. De manera similar, el recall para la clase STANDING con SVM mejora de 0.4849 a 0.4881 y para la clase STANDING con LDA aumenta de 0.5164 a 0.5343 con el aumento de la

probabilidad de mutación. Estos resultados sugieren que el cruzamiento múltiple aleatorio puede ser beneficioso para identificar instancias positivas en ciertas clases, especialmente en situaciones donde el modelo necesita mejorar su capacidad para detectar clases menos representadas.

Sin embargo, también se observa una mayor variabilidad en el recall debido al cruzamiento múltiple aleatorio. En algunos casos, como con el método KNN, el recall para la clase LAYING se mantiene casi constante o incluso disminuye ligeramente en comparación con la primera implementación. Además, para la clase WALKING DOWNSTAIRS con SVM, el recall disminuye de 0.7564 a 0.7399 al aumentar la probabilidad de mutación. Esto indica que, aunque el cruzamiento múltiple aleatorio puede ofrecer mejoras para ciertas clases, también puede introducir variabilidad en el rendimiento, no beneficiando a todos los métodos o clases de manera uniforme.

El cruzamiento múltiple aleatorio suele mejorar el recall para algunos métodos como SVM y LDA, especialmente en clases menos representadas, pero introduce mayor variabilidad. En contraste, el cruzamiento uniforme ofrece un rendimiento más estable y consistente, aunque sin las mismas mejoras en recall en todos los casos. La elección entre ambos métodos depende de la necesidad de mejorar el recall frente a mantener una estabilidad general en el rendimiento.

Por último, observando el comportamiento de esta implementación incrementando el tamaño de la población y la cantidad de generaciones, se obtiene la siguiente tabla 37.

Tabla 37: Exactitud de distintos métodos de clasificación para distintas probabilidades de mutación, segunda implementación, mayor población y generaciones. Fuente: Elaboración propia.

Evaluación	$P_m = 1\%$	$P_m = 5\%$	$P_m = 10\%$	N_atributos
AG+KNN	0.820 (0.847)	0.821 (0.867)	0.816 (0.846)	6
AG+RF	0.847 (0.869)	0.844 (0.865)	0.837 (0.855)	6
AG+SVM	0.841 (0.860)	0.872 (0.897)	0.854 (0.875)	6
AG+LDA	0.805 (0.864)	0.809 (0.863)	0.805 (0.852)	6
AG+NB	0.821 (0.831)	0.807 (0.841)	0.817 (0.857)	10
AG+NB	0.838 (0.859)	0.844 (0.858)	0.831 (0.858)	20
AG+NB	0.836 (0.850)	0.837 (0.845)	0.837 (0.856)	30
AG+NB	0.833 (0.851)	0.838 (0.845)	0.837 (0.855)	40
AG+NB	0.839 (0.845)	0.839 (0.847)	0.833 (0.841)	50
AG+NB	0.832 (0.846)	0.840 (0.851)	0.832 (0.840)	60
AG+NB	0.845 (0.849)	0.837 (0.855)	0.839 (0.843)	70

Al comparar los resultados de la primera y segunda implementación del algoritmo genético con el aumento en la población y las generaciones, pero con métodos de cruzamiento distintos, se obtienen diferencias en la exactitud de estas implementaciones. En general, la

segunda implementación, que utiliza el cruzamiento múltiple aleatorio, tiende a mostrar mejoras en la exactitud en comparación con la primera implementación que utilizó cruzamiento uniforme.

Por ejemplo, en la primera implementación, el método SVM con $P_m=5\%$ muestra una exactitud de 0.830, mientras que en la segunda implementación, con el mismo parámetro de mutación, la exactitud aumenta a 0.872. Esto sugiere que el cruzamiento múltiple aleatorio podría estar favoreciendo una exploración más efectiva del espacio de soluciones, lo que resulta en una mayor exactitud.

Por otra parte, para el RF, la exactitud con $P_m=10\%$ en la primera implementación es 0.871, mientras que en la segunda implementación es 0.855. Aunque hay una ligera disminución en este caso, la variabilidad en los resultados establece que el impacto del método de cruzamiento puede ser dependiente del método de clasificación y los parámetros específicos utilizados.

Mientras que el aumento en la población y generaciones mejora el desempeño general, el cambio en el método de cruzamiento tiene un efecto en los resultados y en los métodos de clasificación utilizados, mas no es tan significativo como el aumento de la población y las generaciones.

4.2.3. Tiempos de ejecución

Con la ejecución de ambas implementaciones, se obtienen los tiempos de la tabla 38 para los clasificadores RF, KNN, LDA y SVM, los cuales apuntaban a una cantidad de 6 atributos, mientras que la tabla 39 representa los tiempos que tomó para ambas implementaciones pero con las diferentes cantidades de atributos utilizados por *Naive Bayes*.

Tabla 38: Tiempos de ejecución para el caso HAR. Fuente: Elaboración propia.

Clasificador	Número de generaciones	Tamaño población	Tiempo de ejecución [s]	N_atributos
RF	10	20	806.23	6
RF	20	40	3389.46	6
KNN	10	20	27.88	6
KNN	20	40	99.59	6
LDA	10	20	12.15	6
LDA	20	40	35.82	6
SVM	10	20	382.49	6
SVM	20	40	1574.36	6

El costo en tiempo de ejecución de los clasificadores combinados con un algoritmo genético varía notablemente en función del modelo utilizado y la configuración del algoritmo. El clasificador RF presenta los tiempos de ejecución más elevados, con 806.23 segundos para 10

Tabla 39: Tiempos de ejecución para el caso HAR, Naive Bayes. Fuente: Elaboración propia.

Clasificador	Número de generaciones	Tamaño población	Tiempo de ejecución [s]	N_atributos
NB	10	20	13.59	10
NB	10	20	52.06	20
NB	10	20	52.30	30
NB	10	20	54.94	40
NB	10	20	56.89	50
NB	10	20	57.53	60
NB	10	20	57.93	70
NB	20	40	45.64	10
NB	20	40	208.38	20
NB	20	40	210.50	30
NB	20	40	221.28	40
NB	20	40	228.22	50
NB	20	40	229.70	60
NB	20	40	234.69	70

generaciones y una población de 20, incrementándose drásticamente a 3389.46 segundos con 20 generaciones y una población de 40. Esto indica que RF es altamente demandante en términos de tiempo de computación.

Por otro lado, KNN muestra un desempeño más eficiente, con tiempos de 27.88 segundos para 10 generaciones y 20 de población, y 99.59 segundos con 20 generaciones y 40 de población. El clasificador LDA es aún más rápido, con solo 12.15 segundos para la configuración más simple y 35.82 segundos para la más compleja, colocándose como el más eficiente en términos de tiempo de ejecución. SVM entre estos clasificadores previos, con tiempos de 382.49 y 1574.36 segundos para las configuraciones ya mencionadas.

En cuanto al clasificador NB, los tiempos de ejecución también dependen de la configuración del algoritmo genético y el número de atributos seleccionados. Para 10 generaciones y una población de 20, los tiempos se encuentran entre 13.59 segundos con 10 atributos y 57.93 segundos con 70 atributos, indicando un incremento moderado. Sin embargo, al aumentar las generaciones a 20 y la población a 40, hay un notable salto de 45.64 segundos con 10 atributos a 208.38 segundos con 20 atributos. Una posible razón para esta gran diferencia es que el conjunto de datos con 10 atributos permite una optimización más rápida del algoritmo genético, debido a su menor complejidad, mientras que el incremento a 20 atributos añade un mayor número de combinaciones posibles, resultando en un aumento en el tiempo de procesamiento.

CAPÍTULO 5

CONCLUSIONES

En el presente documento se presenta un estudio de la aplicación de un algoritmo genético como método de selección de características para reducción de dimensionalidad para dos contextos de problemas de clasificación distintos. En el primer caso de Human Place, se hace uso de un modelo de clasificación de Redes Neuronales y, como una variación, se aplica la clasificación por *Random Forest*, mientras que en el segundo contexto de *human activity recognition*, se hizo uso de RF, KNN, SVM, LDA y NB como los modelos de clasificación.

Para ambos casos, se experimentó con diferentes configuraciones en el número de generaciones y el tamaño de la población: 10 y 20 generaciones, y 20 y 40 individuos en la población. En cuanto a las probabilidades de mutación, el caso Human Place empleó valores de 1 % y 10 %, mientras que el caso HAR probó con 1 %, 5 %, y 10 %. Se aplicaron también distintos métodos de selección de padres: selección lineal por rango para el caso HAR y probabilidades ponderadas para Human Place en NN. En el reemplazo de poblaciones, se utilizó predominantemente el reemplazo total, aunque para las redes neuronales se optó por el reemplazo mediante ruleta, buscando así probar diferentes niveles de variabilidad genética y determinar sus efectos.

Al comparar los dos contextos estudiados, se observa que, aunque ambos casos comparten el objetivo de mejorar el rendimiento de los modelos de clasificación mediante la reducción de dimensionalidad, las diferencias en la naturaleza de los datos y los desafíos específicos de cada contexto destacan la necesidad de adaptar las estrategias de selección de características. En Human Place, la clasificación de alteraciones en la salud mental se benefició particularmente de la selección de características en un contexto de problema de clasificación multietiqueta. Por otro lado, en el caso de HAR, la alta dimensionalidad, la variabilidad entre los modelos de clasificación y la falta de una evaluación más particular de las técnicas empleadas generaron impactos variados en los diferentes modelos. Esto subraya la inexistencia de una solución única para todos estos modelos de clasificación, pero demuestra la flexibilidad del algoritmo genético para ser aplicado en distintos contextos y presentar efectos positivos en la mayoría de los casos.

Los resultados obtenidos por las distintas variaciones del AG en el caso de las NN, donde se aplicaron distintas probabilidades de mutación, número de generaciones, tamaño de población y métodos distintos de reemplazo de la población que alteran la variabilidad genética, entregaron que el algoritmo genético para la selección de características mejoró significativamente la exactitud y redujo la tasa de falsos negativos en redes neuronales. La exactitud promedio con este algoritmo aumentó de un rango original de 0.803 a 0.973, a un rango de 0.910 a 0.977, mientras que la tasa de falsos negativos disminuyó de 2.3 %-20.0 % a 2.0 %-8.1 %.

Se observaron mejoras adicionales en la exactitud, especialmente en las alteraciones de tipo

Emocional y Organizacional, aunque la alteración Física presentó tasas de falsos negativos relativamente altas, indicando que la selección de características podría no haber sido óptima para esta categoría.

Aumentar la probabilidad de mutación al 10 %, ofreciendo una mayor variabilidad genética, afectó negativamente la exactitud promedio y aumentó los falsos negativos, sugiriendo que una mayor variabilidad genética puede introducir ruido, lo que reduce la eficiencia de los clasificadores. Por otro lado, incrementar la población y el número de generaciones mostró mejoras en la exactitud para perturbaciones como Control, Cognitivo y Organizacional, aunque no siempre se observó un impacto positivo en todas las perturbaciones. En particular, la perturbación Física experimentó una disminución en la exactitud.

Ahora, si observando el caso de Random Forest para el contexto de Human Place, este tiene un rendimiento alto en la exactitud promedio para las instancias de datos sintéticos, con valores que oscilaron entre 0.923 y 1.000. Este rendimiento indica que RF es muy eficaz para las características de los datos sintéticos, especialmente en las dimensiones Emocional, Cognitiva y Organizacional.

En contraste, aplicando el algoritmo genético para la selección de características, este resultó en una reducción significativa de la exactitud. La exactitud promedio disminuyó a un rango de 0.389 a 0.864, con una notable disminución en la dimensión Física. La tasa de falsos negativos también aumentó, especialmente para las dimensiones Control, Física y Emocional, mientras que las dimensiones Cognitiva y Organizacional mostraron tasas más estables.

A pesar de incrementar la población y el número de generaciones del AG, no se obtuvieron mejoras notables en la exactitud ni en la tasa de falsos negativos. Estos resultados sugieren que la reducción de características no resultó beneficiosa para la capacidad del modelo para discriminar entre clases.

El AG mostró mejoras considerables en redes neuronales, incrementando la exactitud y reduciendo la tasa de falsos negativos, lo que indica que la selección de características fue efectiva en este contexto. Sin embargo, al aplicar el AG en RF para el caso de Human Place, los resultados fueron opuestos, donde la exactitud disminuyó y la tasa de falsos negativos aumentó, particularmente en la dimensión Física. Esto sugiere que, aunque el AG puede ser beneficioso en escenarios específicos como NN, no siempre es el adecuado para otros modelos de clasificación como *Random Forest*, donde la reducción de características puede introducir ruido y afectar negativamente el rendimiento de los clasificadores.

A través de la aplicación del algoritmo genético en el segundo caso de *Human Activity Recognition*, este escenario permitió evaluar si las mejoras observadas en el caso de Human Place, podían replicarse en un contexto diferente.

Observando los clasificadores utilizados en el caso de *Human Activity Recognition*, se observa un comportamiento variado dependiendo de la configuración del algoritmo genético y el clasificador utilizado. En el caso de KNN, con RF como método de selección de características,

la exactitud es de 0.904 con 6 atributos. Al aplicar el algoritmo genético, se alcanza un valor máximo de 0.858, sin mostrar una mejora significativa en el rendimiento del clasificador. En comparación, RFE y Boruta lograron resultados de 0.838 y 0.864 respectivamente, lo que sugiere que AG superó a RFE y se acercó a Boruta. Con el aumento en el número de generaciones y la población, los resultados mejoraron a 0.865, 0.866 y 0.847 para probabilidades de mutación de 1 %, 5 % y 10 % respectivamente. Esto sugiere que ajustar los parámetros puede mejorar la calidad, aunque, como se observa con un 10 % de mutación, la variabilidad genética podría afectar negativamente el rendimiento de los clasificadores.

Respecto al comportamiento del recall en KNN y en los otros clasificadores, esta métrica en general disminuyó en la mayoría de clases, teniendo casos excepcionales para algunas probabilidades de mutación y clasificadores. Específicamente, en el caso del KNN original, el recall era 1.0 para la clase *LAYING*, 0.9339 para *SITTING*, 0.9161 para *STANDING*, y 0.8131 para *WALKING UPSTAIRS*. Después de aplicar el algoritmo genético como método de selección de características, se observaron valores de recall de 0.6696, 0.7105, 0.5048 y 0.6901, respectivamente. Aunque el AG generó una reducción general en el recall, también se notó un aumento en la exactitud del clasificador, lo cual sugiere una mejora en el rendimiento global del modelo.

Sin embargo, se identificaron casos excepcionales donde la aplicación del AG mostró mejoras notables. Por ejemplo, con AG+LDA, el recall para la clase *WALKING UPSTAIRS* aumentó de 0.6168 a valores superiores a 0.7057, alcanzando hasta 0.7466. Estos resultados destacan la capacidad del AG para mejorar el rendimiento en situaciones específicas, a pesar de la tendencia general a la reducción del recall. Esto indica que, mientras que el AG puede reducir el recall en general, también tiene el potencial de mejorar significativamente el rendimiento en ciertas clases y configuraciones del clasificador.

El ajuste de parámetros en un algoritmo genético representa un balance entre beneficios y desventajas. Por un lado, ajustar parámetros como la tasa de mutación, la probabilidad de cruzamiento y el tamaño de la población permiten que el AG se adapte mejor a las características específicas del problema, optimizando su capacidad para explorar el espacio de soluciones y evitar mínimos locales. Esto, a su vez, puede conducir a un desempeño superior del modelo, mejorando la exactitud y la eficiencia en la reducción de dimensionalidad. Sin embargo, el ajuste de parámetros también puede llevar a ciertas desventajas, como el incremento en la complejidad y el tiempo de cómputo, puesto que identificar la combinación óptima de parámetros puede requerir múltiples pruebas y evaluaciones. junto con el uso constante del modelo de clasificación por cada una de las generaciones para evaluar cada individuo de la población y obtener sus valores de aptitud respectivos. Además, un ajuste excesivo puede llevar a un sobreajuste del modelo a los datos específicos de entrenamiento, reduciendo su capacidad de generalización. Por lo tanto, es fundamental encontrar un equilibrio que maximice los beneficios del ajuste de parámetros mientras se minimizan sus desventajas, asegurando así la robustez y eficacia del AG en distintos contextos.

En contraste, para el caso de SVM se obtuvieron mejores resultados en comparación con RFE, Boruta y RF, que alcanzaron valores de 0.822, 0.869 y 0.868, respectivamente, frente

a un máximo de 0.883 con una probabilidad de mutación del 5 % y 0.873 con 1 %. Estos valores pueden considerarse *outliers*, ya que con un mayor número de iteraciones, población y generaciones, el valor de exactitud más alto encontrado fue 0.864. Aunque es posible alcanzar estos valores, se requiere utilizar los parámetros específicos del caso, ya que entre los 561 atributos disponibles, hay diversas maneras de seleccionar los 6 mencionados y no se lograron con el mayor número de iteraciones. Un ejemplo de mejora se puede observar en el recall obtenido con AG+SVM, que aumentó en todas las clases, especialmente con una probabilidad de mutación del 1 %. En particular, la clase LAYING subió de 0.8208 a 0.9560, y la clase WALKING de 0.7961 a 0.9069.

Estos *outliers* obtenidos en los resultados, lejos de ser considerados exclusivamente como un error, representan el potencial para mejorar los parámetros aplicados para el algoritmo genético en los distintos clasificadores y casos. En particular, en el contexto del segundo caso de HAR, *outliers* contribuyeron a identificar configuraciones del algoritmo genético que, aunque atípicas, lograron optimizar el rendimiento de ciertos clasificadores, como se evidenció en las mejoras observadas en el recall para clases específicas. Estos resultados sugieren que, en lugar de ser descartados, estos pueden ofrecer oportunidades valiosas para refinar el AG y explorar soluciones que, de otro modo, podrían pasar desapercibidas. Así, el manejo estratégico de estos *outliers* podría ser un aspecto clave en futuras investigaciones para maximizar la eficiencia y efectividad de los algoritmos de selección de características.

En la segunda implementación del caso HAR, al introducir una mayor variabilidad genética mediante cruzamiento aleatorio múltiple, no se observaron cambios significativos en los valores de desempeño. Sin embargo, el uso de un método que fomente la variabilidad genética permite explorar más soluciones y detectar *outliers* que podrían mejorar la evaluación del clasificador. Por ejemplo, en AG+SVM, el valor máximo obtenido en la primera implementación fue de 0.883, mientras que en la segunda alcanzó 0.897. No obstante, es importante notar que el desempeño de un algoritmo genético está condicionado por su configuración, ya que en otros casos el cruzamiento uniforme, aunque más sencillo, logró los mejores resultados.

Los resultados mencionados muestran que el rendimiento del AG varía significativamente según el modelo de clasificación utilizado y el contexto de cada investigación. Por ejemplo, mientras que AG mejoró el rendimiento de NN, tuvo un efecto adverso para RF en el primer caso. En contraste, para el segundo caso de HAR, presentó mejoras para algunos casos con RF, pero igual empeoró el rendimiento de algunos clasificadores, pero no en la misma medida que ocurrió en el primer caso de Human Place. Esto refleja la necesidad de ajustar parámetros al AG según sea el contexto de la investigación, como el contexto de Human Place tenía hiperparámetros particulares de número de estimadores, número máximo de estimadores, mínimo de muestras, máximo de profundidad y criterio para el RF, la modificación de estos junto con los cambios en el AG pueden llevar a una mejora en este modelo de AG+RF para Human Place. En cambio, el caso HAR se beneficia de cambiar parámetros del AG según cada clasificador, puesto que algunos métodos de cruzamiento pueden obtener mejor rendimiento para uno de estos, mientras que los demás se ven perjudicados.

En los tiempos de ejecución entre los clasificadores y configuraciones del algoritmo genético, en el caso de Human Place, este tiempo en las redes neuronales y RF aumentó significativamente con más generaciones y una mayor población, destacando que la complejidad del modelo impacta directamente en el tiempo requerido. La reducción de atributos mediante el AG mejoró el tiempo de ejecución en un 2.77 %, evidenciando que una selección de características más eficiente puede acelerar el proceso de clasificación.

En el caso HAR, clasificadores como LDA y KNN demostraron ser mucho más eficientes que RF y SVM en términos de tiempo. LDA fue el más rápido, seguido por KNN, mientras que RF mostró los mayores tiempos de ejecución, especialmente con configuraciones más exigentes. Naive Bayes también mostró un incremento moderado en tiempos con más atributos, debido a la mayor complejidad combinatoria.

Respecto a la variabilidad genética y el ruido que este puede introducir, se presenta como una oportunidad el estudiar más profundamente el cómo equilibrar la variabilidad genética para maximizar la exploración del espacio de búsqueda, sin perder el rendimiento en la exactitud. Esto aplicando distintas tasas de mutación, métodos de cruzamiento, selección de padres o selección de poblaciones.

Para el contexto de Human Place, se evidenció que la selección de características mediante AG tuvo un impacto negativo en la exactitud y tasa de falsos negativos, especialmente en la dimensión Física para RF. Investigaciones futuras podrían centrarse en entender mejor por qué ciertas dimensiones se ven más afectadas que otras y el cómo se podrían mitigar estos efectos. Esto se puede ampliar más, aplicando AG en distintos contextos para determinar qué características del problema pueden causar que el uso del AG con el método de clasificación en particular, genere una mejora o empeoramiento de las métricas de calidad.

Aplicando algoritmos genéticos, se espera obtener un número de características reducido que permita la aplicación del modelo de clasificación con una mayor dimensionalidad, de esta manera, mejorar la eficiencia de estos modelos a través de los resultados otorgados por el AG.

En conclusión, el estudio revela que el uso del algoritmo genético para la selección de características puede tener efectos diversos dependiendo del contexto y del modelo de clasificación utilizado. Mientras que en el caso de Human Place, el AG mejoró significativamente la exactitud y redujo la tasa de falsos negativos en redes neuronales, su aplicación en RF resultó en una disminución de la exactitud y un aumento en los falsos negativos, especialmente en la dimensión Física. Esto destaca que el AG puede ser altamente efectivo para algunos modelos y contextos, pero también puede introducir ruido y disminuir el rendimiento en otros casos. La variabilidad en los resultados obtenidos subraya la necesidad de adaptar las estrategias de selección de características y los parámetros del AG a las características específicas del problema y del clasificador.

En el contexto de Human Activity Recognition, el AG mostró un rendimiento mixto, con mejoras en algunos clasificadores como KNN y SVM, pero sin una mejora generalizada en todos

los casos. Los resultados también indicaron que la variabilidad genética y los valores atípicos pueden proporcionar oportunidades para optimizar los clasificadores, aunque también pueden complicar el ajuste del AG. Por lo tanto, es crucial considerar el equilibrio entre exploración y explotación al ajustar el AG, así como investigar más a fondo los efectos específicos de la selección de características en diferentes modelos. Este enfoque permitirá maximizar la eficiencia del AG y mejorar el rendimiento de los modelos de clasificación en diversos contextos.

REFERENCIAS BIBLIOGRÁFICAS

- [Abdi, 2007] Abdi, H. (2007). Singular value decomposition (svd) and generalized singular value decomposition. *Encyclopedia of measurement and statistics*, 907(912):44.
- [Alhijawi y Awajan, 2023] Alhijawi, B. y Awajan, A. (2023). Genetic algorithms: theory, genetic operators, solutions, and applications. *Evolutionary Intelligence*, pp. 1–12.
- [Ali y Saeed, 2023] Ali, W. y Saeed, F. (2023). Hybrid filter and genetic algorithm-based feature selection for improving cancer classification in high-dimensional microarray data. *Processes*, 11(2):562.
- [Anowar et al., 2021] Anowar, F., Sadaoui, S., y Selim, B. (2021). Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). *Computer Science Review*, 40:100378.
- [Beauxis-Aussalet y Hardman, 2014] Beauxis-Aussalet, E. y Hardman, L. (2014). Visualization of confusion matrix for non-expert users. En *IEEE Conference on Visual Analytics Science and Technology (VAST)-Poster Proceedings*, pp. 1–2.
- [Bellman, 1957] Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, New Jersey, sixth edición. A Rand Corporation Research Study.
- [Bielza et al., 2011] Bielza, C., Li, G., y Larranaga, P. (2011). Multi-dimensional classification with bayesian networks. *International Journal of Approximate Reasoning*, 52(6):705–727.
- [Chen et al., 2020] Chen, R.-C., Dewi, C., Huang, S.-W., y Caraka, R. E. (2020). Selecting critical features for data classification based on machine learning methods. *Journal of Big Data*, 7(1):52.
- [Cormack, 1971] Cormack, R. M. (1971). A review of classification. *Journal of the Royal Statistical Society: Series A (General)*, 134(3):321–353.
- [De Diego et al., 2022] De Diego, I. M., Redondo, A. R., Fernández, R. R., Navarro, J., y Moguerza, J. M. (2022). General performance score for classification problems. *Applied Intelligence*, 52(10):12049–12063.
- [De Jong, 1988] De Jong, K. (1988). Learning with genetic algorithms: An overview. *Machine learning*, 3:121–138.
- [Destercke, 2014] Destercke, S. (2014). Multilabel prediction with probability sets: the hamming loss case. En *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 15th International Conference, IPMU 2014, Montpellier, France, July 15–19, 2014, Proceedings, Part II 15*, pp. 496–505. Springer.
- [Gao et al., 2017] Gao, L., Song, J., Liu, X., Shao, J., Liu, J., y Shao, J. (2017). Learning in high-dimensional multimedia data: the state of the art. *Multimedia Systems*, 23:303–313.

- [Geng et al., 2005] Geng, X., Zhan, D.-C., y Zhou, Z.-H. (2005). Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(6):1098–1107.
- [Ghazal et al., 2021] Ghazal, S., Qureshi, W. S., Khan, U. S., Iqbal, J., Rashid, N., y Tiwana, M. I. (2021). Analysis of visual features and classifiers for fruit classification problem. *Computers and Electronics in Agriculture*, 187:106267.
- [Godbole y Sarawagi, 2004] Godbole, S. y Sarawagi, S. (2004). Discriminative methods for multi-labeled classification. En *Pacific-Asia conference on knowledge discovery and data mining*, pp. 22–30. Springer.
- [Gonçalves et al., 2013] Gonçalves, E. C., Plastino, A., y Freitas, A. A. (2013). A genetic algorithm for optimizing the label ordering in multi-label classifier chains. *IEEE 25th International Conference on Tools with Artificial Intelligence*, 25:469–476.
- [Huang et al., 2015] Huang, H., Xu, H., Wang, X., y Silamu, W. (2015). Maximum f1-score discriminative training criterion for automatic mispronunciation detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(4):787–797.
- [Huang et al., 2018] Huang, X., Wu, L., y Ye, Y. (2018). A review on dimensionality reduction techniques. *International Journal of Pattern Recognition and Artificial Intelligence*, pp. 1–23.
- [Ismi et al., 2016] Ismi, D. P., Panchoo, S., y Murinto, M. (2016). K-means clustering based filter feature selection on high dimensional data. *International Journal of Advances in Intelligent Informatics*, 2(1):38–45.
- [Jain et al., 2000] Jain, A. K., Goodman, E. D., Kuhn, L. A., Raymer, M. L., y Punch, W. F. (2000). Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4:164–171.
- [Jeatrakul y Wong, 2009] Jeatrakul, P. y Wong, K. W. (2009). Comparing the performance of different neural networks for binary classification problems. En *2009 Eighth International Symposium on Natural Language Processing*, pp. 111–115. IEEE.
- [Koyejo et al., 2015] Koyejo, O. O., Natarajan, N., Ravikumar, P. K., y Dhillon, I. S. (2015). Consistent multilabel classification. En Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., y Garnett, R., editores, *Advances in Neural Information Processing Systems*, volumen 28. Curran Associates, Inc.
- [Kuri y Galaviz, 2002] Kuri, A. y Galaviz, J. (2002). *Algoritmos Genéticos*. IPN.
- [Larrain et al., 2022] Larrain, G. B., Rojas-Morales, N., Jeneral, P. D. O., y Rogel, N. G. (2022). Towards a classifier ensemble to prevent burnout syndrome on university students. En *2022 41st International Conference of the Chilean Computer Science Society (SCCC)*, pp. 1–8. IEEE.

- [Lazar *et al.*, 2012] Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., de Schaetzen, V., Duque, R., Bersini, H., y Nowe, A. (2012). A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM transactions on computational biology and bioinformatics*, 9(4):1106–1119.
- [Leung y Hung, 2008] Leung, Y. y Hung, Y. (2008). A multiple-filter-multiple-wrapper approach to gene selection and microarray data classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(1):108–117.
- [Liu y Gillies, 2016] Liu, R. y Gillies, D. F. (2016). Overfitting in linear feature extraction for classification of high-dimensional image data. *Pattern Recognition*, 53:73–86.
- [Maldonado *et al.*, 2014] Maldonado, S., Weber, R., y Famili, F. (2014). Feature selection for high-dimensional class-imbalanced data sets using support vector machines. *Information sciences*, 286:228–246.
- [Mitchell, 1998] Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.
- [Padmaja y Vishnuvardhan, 2016] Padmaja, D. L. y Vishnuvardhan, B. (2016). Comparative study of feature subset selection methods for dimensionality reduction on scientific data. En *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pp. 31–34. IEEE.
- [Pai *et al.*, 2022] Pai, G., Bronstein, A., Talmon, R., y Kimmel, R. (2022). Deep isometric maps. *Image and Vision Computing*, 123:104461.
- [Quinlan, 1990] Quinlan, J. R. (1990). Probabilistic decision trees. En *Machine Learning*, pp. 140–152. Elsevier.
- [Room, 2019] Room, C. (2019). Confusion matrix. *Mach. Learn*, 6:27.
- [S Velliangiri y Thankumar, 2019] S Velliangiri, S. A. y Thankumar, I. (2019). A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, 165:104–111.
- [Santra y Christy, 2012] Santra, A. y Christy, C. J. (2012). Genetic algorithm and confusion matrix for document clustering. *International Journal of Computer Science Issues (IJCSI)*, 9(1):322.
- [Shah y Patel, 2016] Shah, F. P. y Patel, V. (2016). A review on feature selection and feature extraction for text classification. En *2016 international conference on wireless communications, signal processing and networking (WiSPNET)*, pp. 2264–2268. IEEE.
- [Siedlecki y Sklansky, 1989] Siedlecki, W. y Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. *Pattern recognition letters*, 10(5):335–347.
- [Sivanandam *et al.*, 2008] Sivanandam, S., Deepa, S., Sivanandam, S., y Deepa, S. (2008). *Genetic algorithms*. Springer.

- [Tan *et al.*, 2008] Tan, F., Fu, X., Zhang, Y., y Bourgeois, A. G. (2008). A genetic algorithm-based method for feature subset selection. *Soft Computing*, 12:111–120.
- [Tharwat, 2020] Tharwat, A. (2020). Classification assessment methods. *Applied computing and informatics*, 17(1):168–192.
- [Tsoumakas *et al.*, 2006] Tsoumakas, G., Katakis, I., y Vlahavas, I. (2006). A review of multi-label classification methods. En *Proceedings of the 2nd ADBIS workshop on data mining and knowledge discovery (ADMKD 2006)*, pp. 99–109.
- [Verleysen y François, 2005] Verleysen, M. y François, D. (2005). The curse of dimensionality in data mining and time series prediction. En *International work-conference on artificial neural networks*, pp. 758–770. Springer.
- [William *et al.*, 1995] William, W., Olvi, M., Nick, S., y W., S. (1995). Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5DW2B>.
- [Zebari *et al.*, 2020] Zebari, R. R., Abdulazeez, A. M., Zeebaree, D. Q., Zebari, D. A., y Saeed, J. N. (2020). A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction. *Journal of applied science and technology trends (JASTT)*, 1:56–70.
- [Zeebaree *et al.*, 2017] Zeebaree, D. Q., Haron, H., Abdulazeez, A. M., y Zeebaree, S. (2017). Combination of k-means clustering with genetic algorithm: A review. *International Journal of Applied Engineering Research*, 12(24):14238–14245.
- [Zhang y Zhou, 2007] Zhang, M.-L. y Zhou, Z.-H. (2007). MI-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048.
- [Zhang y Zhou, 2013] Zhang, M.-L. y Zhou, Z.-H. (2013). A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837.
- [Zhu *et al.*, 2005] Zhu, S., Ji, X., Xu, W., y Gong, Y. (2005). Multi-labelled classification using maximum entropy method. En *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 274–281.