TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA

POSGRADO DE MAESTRÍA EN SISTEMAS COMPUTACIONALES

"CONFIGURACIÓN DE ARQUITECTURA DE UNA RNA MEDIANTE ALGORITMOS GENÉTICOS"

TESIS

QUE PARA OBTENER EL TÍTULO DE

MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA

URIEL RUBIO ESCAMILLA

ASESOR

DR. SIMÓN PEDRO ARGUIJO HERNÁNDEZ

CO-ASESOR

M.I.A. ROBERTO ÁNGEL MELENDEZ ARMENTA

MISANTLA, VERACRUZ, MÉXICO, AGOSTO de 2018

AGRADECIMIENTOS

Uriel Rubio Escamilla agradece al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado para estudiar el posgrado.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I. GENERALIDADES

1.1 Introducción

Actualmente, el cuidado de la salud tiene una relevancia por demás importante para una gran cantidad de personas y es común que éstas incluyan actividades físicas dentro de sus actividades y cuiden su nutrición sometiéndose a dietas. A pesar de los cuidados mencionados, las enfermedades cardiovasculares (ECV, funcionamiento anormal del corazón y los vasos sanguíneos) y la diabetes están presentes en la vida cotidiana de cada vez más personas. De acuerdo a la Organización Mundial de la Salud [1], de los 56.4 millones de defunciones registradas en el mundo en 2016, más de la mitad (el 54%) fueron consecuencia de las 10 causas que se indican en la Figura 1. Como se puede apreciar en la Figura 1 destacan la cardiopatía isquémica y el accidente cerebrovascular que ocasionaron 15.2 millones de defunciones en 2015 [1], y han sido las principales causas de mortalidad durante los últimos 15 años. Así mismo, se observa en la Figura 1, que la diabetes mellitus en 2016 aparece en sexto lugar con un total 1.59 de defunciones.

Padecer ECV y diabetes está ligada a diversos factores como, por ejemplo, herencia, dietas poco saludables, sedentarismo, obesidad, entre otras. En México, según el Instituto Nacional de Estadística y Geografía (INEGI) en 2016, se registraron 685,766 defunciones [2], de las cuales 136,342 se debieron a enfermedades del corazón, 97,743 a enfermedades isquémicas del corazón y 105,572 a la diabetes mellitus. Según un artículo publicado en el periódico *El Siglo De Durango* entre las primeras siete causas de defunciones en México se encuentran la diabetes mellitus y distintos tipos de ECV [3], tal como lo muestran en la Figura 2.

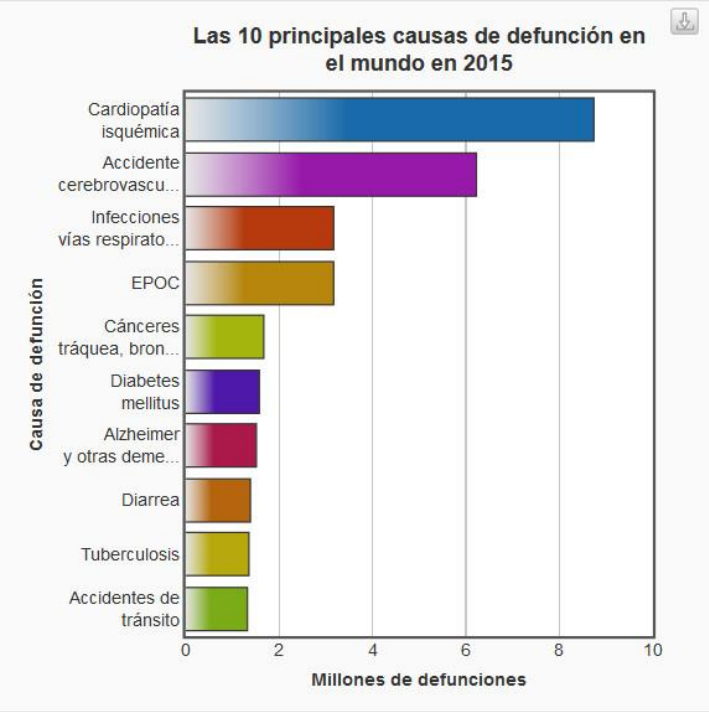


Figura 1. Las 10 principales causas de defunción en el mundo en 2015. Tomada de [1].

Figura 2. Principales causas de muertes en México en el 2016 (INEGI). Tomada de [3].

1.2 Definición de problema

Los grandes avances tecnológicos en general y de la computación en particular se ven reflejados en la solución de problemas en diversas áreas, por ejemplo, en el área biomédica, se emplean técnicas y algoritmos Inteligencia Artificial (IA). Las Redes Neuronales Artificiales (RNA) son ampliamente utilizadas en el área biomédica para la clasificación de enfermedades, utilizando como referencia registros existentes sobre la enfermedad en cuestión, baste como ejemplos [4]–[6].

Cuando se utiliza una RNA para abordar un problema, la complejidad radica en determinar la arquitectura de la RNA, a fin de obtener un mayor grado de exactitud. Preguntas como:

- ¿Cuántas capas ocultas deben utilizarse?
- ¿Cuántas neuronas debe tener cada capa oculta?
- ¿Qué tipo de función de activación debe utilizarse?
- ¿Con cuál algoritmo debe realizarse el entrenamiento?
- ¿Qué tasa de aprendizaje debe utilizarse?

Es decir, **¿Cuál es la configuración adecuada?**, se vuelven tediosas de resolver pues las respuestas son dictadas por la experiencia como experto en el tema o por la experimentación de probar cada combinación posible, sin embargo, la primera opción es muy difícil de adquirir y la segunda opción, es imposible probar todas y cada una de las combinaciones puesto que son infinitas.

1.3 Objetivos

1.3.1 Objetivo general

Utilizar algoritmos genéticos para encontrar la configuración de la arquitectura de una RNA, la cual maximice la exactitud de ésta, dentro de un espacio de búsqueda y tiempo determinado.

1.3.2 Objetivos específicos

- Buscar tres *dataset* comúnmente utilizados en el área biomédica disponibles online: arritmias, problemas del corazón y diabetes.
- Realizar el preprocesamento de los *dataset*.
- Analizar el universo de soluciones y determinar el espacio de búsqueda para las diferentes variables que conforman la arquitectura de una RNA.
- Desarrollar un prototipo que mediante Algoritmos Genéticos (AG) maximice la exactitud de una RNA en clasificación supervisada, examinando diferentes arquitecturas de la RNA.
- Experimentar el prototipo con cada uno de los 3 *dataset*, y comprobar los resultados obtenidos contra los reportados en la literatura.

1.4 Justificación

El presente proyecto, logra reducir el tiempo de trabajo tanto en recursos computacionales como humanos para encontrar la configuración correcta de una RNA que se adapte al problema a resolver. El investigador (usuario) solo debe establecer los límites del espacio de búsqueda (límites de las variables que conforman la arquitectura de la RNA) y valores de las variables pertinentes a cómputo evolutivo, y esperar los resultados de los experimentos (puede monitorear los experimentos, si así lo desea).

Mediante el cómputo evolutivo, de manera autónoma se analizan y evolucionan las mejores propuestas de solución, a fin de encontrar una arquitectura de RNA cuya exactitud que, si bien puede no ser la más alta, sí se encontrará muy cerca de la misma. Si se desea incrementar la exactitud de clasificación de la RNA, el resultado obtenido por la propuesta del presente proyecto puede ser utilizado como punto de partida para utilizar/implementar otras técnicas que permitan mejorar la exactitud.

Los tres *dataset* utilizados en el desarrollo de este trabajo fueron: arritmias, problemas del corazón y diabetes (los cuales se describen en el siguiente capítulo). Estos se eligieron por

dos razones muy importantes, la primera: son los *dataset* más utilizados en la literatura, y segunda: estas enfermedades aparecen en la lista de las 10 enfermedades que más defunciones registran en México y en el mundo, tal como se mencionó anteriormente.

Los resultados obtenidos pueden utilizarse para desarrollar o mejorar los recursos tecnológicos con los que cuenta el área médica para la toma de decisiones.

1.5 Alcances

1. En el presente proyecto solo se experimenta con arquitecturas multi-capa alimentada hacia adelante, entrenadas con el algoritmo de propagación hacia atrás.
2. Los parámetros de la arquitectura de una RNA a considerar en el cómputo evolutivo se muestran en la Tabla 1, mientras que en la Tabla 2 se muestran los parámetros ajenos a la arquitectura de una RNA, pero también considerados como variables en el cómputo evolutivo. En ambas tablas se muestran los tipos de datos y los límites[1] de los valores a tomar para cada parámetro.

   En el caso particular del parámetro "Razón de muestreo de los datos" mencionado en la Tabla 2, en cada uno de los distintos valores que puede tomar el parámetro, se menciona el porcentaje de datos que corresponden al conjunto de entrenamiento (entrenamiento y validación) y al conjunto de prueba, respectivamente.

   Solo se considera experimentar con 3 *dataset*: arritmias, problemas del corazón y diabetes, todos ellos obtenidos de [7].

[1] Los límites de cada parámetro fueron establecidos tomando en cuenta los recursos computacionales disponibles para realizar los experimentos.

Parámetro	Límites	Tipo de dato
Número de capas ocultas	[1-3]	Entero
Numero de neuronas en la primera capa oculta	[5-50]	Entero
Numero de neuronas en la segunda capa oculta	[5-50]	Entero
Numero de neuronas en la tercera capa oculta	[5-50]	Entero
Número de épocas	[10-50]	Entero
Tasa de aprendizaje	[0.2-0.5]	Flotante

Tabla 1. Límites y tipo de dato de los parámetros de la arquitectura de la RNA considerados en el cómputo evolutivo.

Parámetro	Límites	Tipo de dato
Razón de muestreo de los datos	[1-3] Dónde: 1=70%-30%; 2=80%-20%; 3=90%-10%	Entero (categórico)

Tabla 2. Límites y tipo de dato de los parámetros ajenos a la arquitectura de la RNA y considerados en el cómputo evolutivo.

1.6 Limitaciones

- Por falta de tiempo y políticas de privacidad se descarta la opción de buscar *dataset* en instituciones médicas (públicas y privadas) y se opta por obtener los datos desde [7].

CAPÍTULO II. REVISIÓN DE LA LITERATURA

2.1 Fuentes de datos

Se consideran tres *dataset* comúnmente utilizados en el área médica para realizar experimentos, los cuales se encuentran alojados en el repositorio *UCI* [7], los *dataset* contienen registros de datos de pacientes vinculados a ECV y diabetes: el primer *dataset* se utiliza para determinar a partir de los datos del paciente si existe o no algún tipo de arritmia cardíaca [8]; el segundo *dataset* se utiliza para determinar la presencia o ausencia de problemas del corazón en un paciente [9]; mientras que el tercer *dataset* se utiliza para determinar la presencia o ausencia de diabetes en el paciente [10].

2.1.1 *Dataset*: arritmia cardíaca

El *dataset* fue donado en enero de 1998 [11] y contiene información personal de 452 pacientes e información extraída mediante la práctica de ECG de 12 derivaciones a los mismos pacientes. El *dataset* está conformado por 279 atributos de los cuales los primeros 4 contienen información personal del paciente (edad, sexo, estatura y peso), los siguientes 11 atributos (del atributo 5 al atributo 15) contienen datos más comunes tomados a partir del ECG de 12 canales, el resto de los atributos contienen información de derivaciones calculadas a partir del ECG del paciente. De los 279 atributos 206 son de tipo lineal y 63 de tipo categórico. Del total de valores contenidos en el *dataset*, el 0.32% (408 valores) son valores faltantes.

De los 452 pacientes, 249 son mujeres y 203 hombres; el paciente de menor edad registra apenas 1 año, mientras que el más longevo registra 83 años. La Tabla 3 muestra los dieciséis tipos de arritmia cardíaca considerados, donde, la clase 1 representa la ausencia de arritmia cardíaca y las clases que van desde la 2 hasta la 15 representan la presencia de algún tipo de arritmia cardíaca. Cada paciente pertenece a una de las dieciséis clases, siendo la clase 1 (ausencia de arritmia cardíaca) la que más pacientes contiene con un total de 245,

mientras que los 183 pacientes restantes se encuentran distribuidos en alguno de los 15 tipos arritmia cardíaca restantes, situaciones que se representan gráficamente en la . Otra situación que salta a la vista en la es el desbalanceo existente en cuanto al número de instancias por clase.

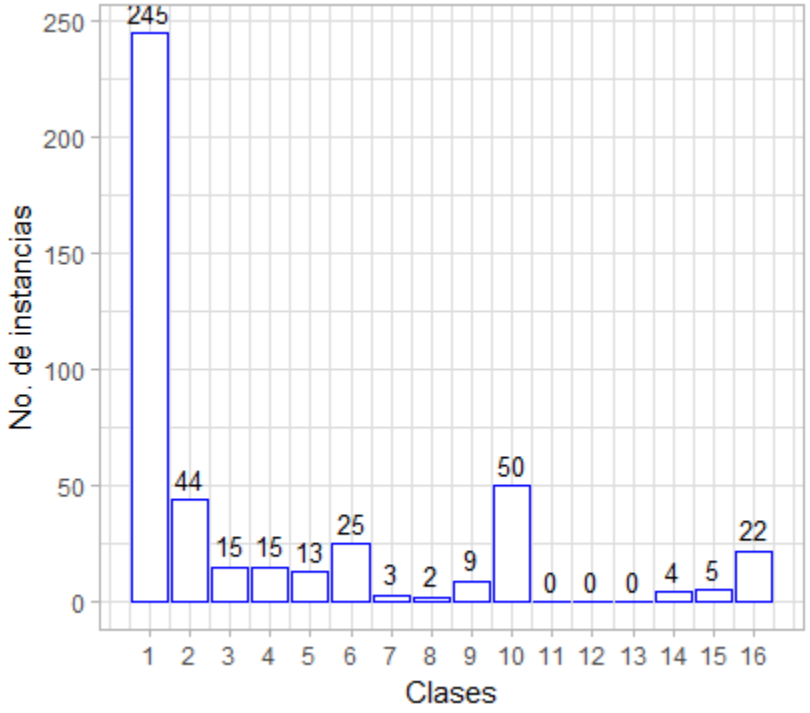


Figura 3. Distribución de las instancias sobre las clases. Dataset: arritmia cardíaca.

(Código de la clase)[2]	Clase	Número de instancias
1	Ausencia de arritmia	245
2	Cardiopatía isquémica (enfermedad coronaria o isquemia cardíaca)	44
3	Infarto de miocardio anterior	15
4	Infarto de miocardio inferior	15
5	Taquicardia sinusal	13
6	Bradicardia sinusal	25
7	Contracción ventricular prematura (CVP)	3
8	Contracción auricular prematura	2
9	Bloqueo de la rama izquierda	9
10	Bloqueo de la rama derecha	50
11	Bloque auriculoventricular (AV) de primer grado	0
12	Bloque AV de segundo grado	0
13	Bloque AV de tercer grado	0
14	Hipertrofia ventricular izquierda	4
15	Fibrilación auricular	5
16	Otras	22

Tabla 3. Distribución de las instancias sobre las clases. Dataset: arritmia cardíaca.

[2] Código bajo el cual se identifica la clase en el dataset. El código puede cambiar durante el preprocesamiento.

2.1.2 *Dataset*: problemas del corazón

El *dataset* [9] está conformado por 920 diagnósticos de problemas de corazón recolectados en cuatro lugares distintos, la base de datos *Cleveland* cuenta con el mayor número de diagnósticos. Las cuatro bases de datos comparten el mismo formato para todos los diagnósticos, almacenando 76 atributos. Todos los trabajos que analizan este *dataset* utilizan solo los 14 atributos y omiten el resto, dichos atributos son integrados por datos tanto de tipo lineal como categórico.

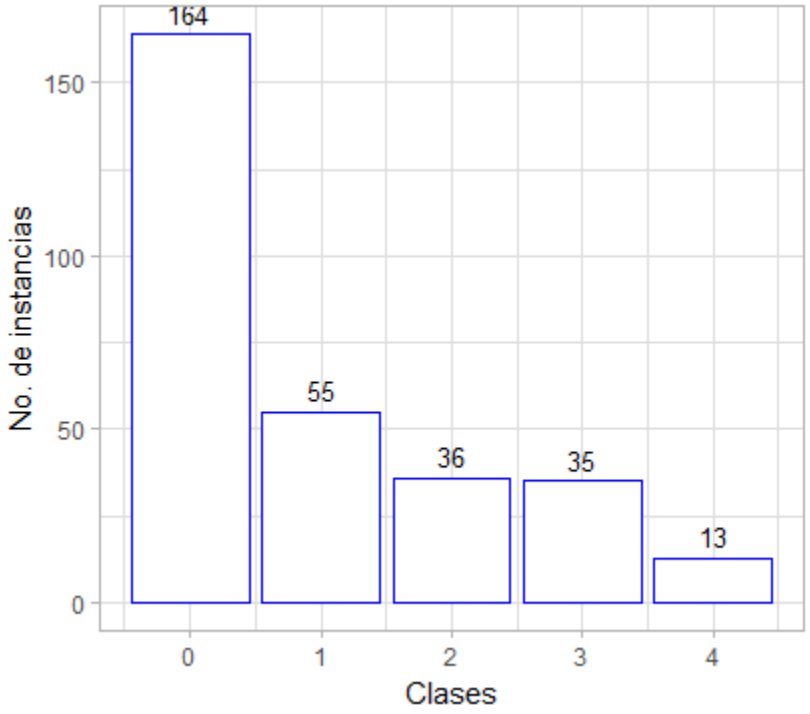


Figura 4. Distribución de las instancias sobre las clases. Dataset: problemas del corazón.

Para los experimentos del presente proyecto se utiliza la base de datos *Cleveland*, la cual está conformada por 303 diagnósticos (instancias), de los cuales 206 registros son de pacientes masculinos y 97 de pacientes femeninos, el paciente de menor edad cuenta con 29 años y el de mayor edad cuenta con 77 años. Los valores correspondientes a los catorce atributos utilizados en todos los trabajos relacionados a este *dataset* (específicamente la base de datos *Cleveland*) se encuentran alojados en [12]. Del total de los valores (4,256 valores) el 0.14% (6 valores) son valores faltantes.

El atributo "*num*" contiene valores que determinan si un paciente tiene problemas del corazón (valores del 1 al 4) o no los tiene (valor 0). Cada valor del atributo mencionado anteriormente representa una clase cuyo nombre es desconocido, la distribución de las instancias sobre cada una de las clases se muestra gráficamente en la Figura 4.

2.1.3 *Dataset*: diabetes

El *dataset* utilizado es *Pima Indians Diabetes* [13], el cual se compone de 768 registros, únicamente de pacientes femeninos mayores de 20 años de edad, y 9 atributos de tipo lineal y categórico (incluido el atributo de clasificación). El valor del noveno atributo describe si el paciente padece o no diabetes, identificando con el valor 1 a las poseedoras de diabetes (500 pacientes), y con un 0 a los que se encuentran libres de padecer la misma (268 pacientes), situación mostrada gráficamente en la Figura 5.

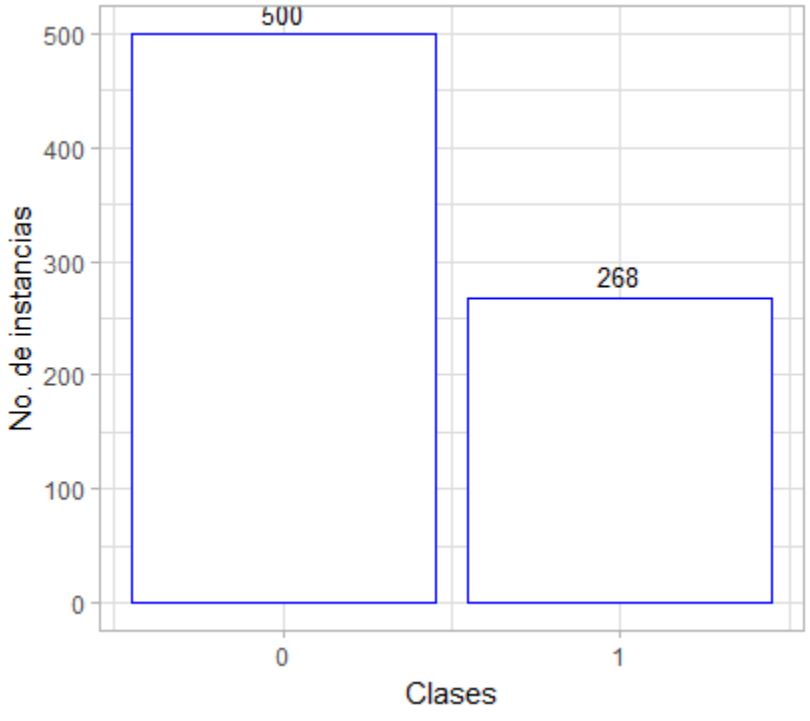


Figura 5. Distribución de las instancias sobre las clases. Dataset: Diabetes.

Los valores faltantes en el *dataset* están representados con el número 0. De 6144 valores (sin considerar el noveno atributo) 763 son valores faltantes (el 12.42%).

2.2 Trabajos relacionados

2.2.1 *Dataset*: arritmia cardíaca

El *dataset* [11] se ha utilizado con diferentes métodos en problemas de clasificación y selección de atributos, algunos resultados se mencionan a continuación :

En [14], normalizan el conjunto de datos en el rango [0, 1] y utilizan ponderación difusa para preprocesar los mismos, y posteriormente clasifican utilizando el método *Artificial Immune Recognition System (AIRS).* Para la clasificación dividen el conjunto de datos en dos subconjuntos, entrenamiento y prueba, utilizan tres razones de muestreo: 50%-50%, 70%-30% y 80%-20%, obteniendo exactitudes de 78.79%, 75% y 80.77% en la clasificación, respectivamente.

Utilizar todos los atributos del conjunto de datos no garantiza obtener la exactitud más alta en la clasificación, en [15], utilizan algoritmos genéticos para determinar los atributos que deben pertenecer al subconjunto con el fin de lograr el mejor resultado en la clasificación. Antes de utilizar los AG realizan un estudio a los datos con la finalidad de detectar datos atípicos, en cada clase, se remueve el valor que se encuentra fuera de los límites establecidos por la media más/menos tres veces la desviación estándar. La función objetivo consiste en minimizar el error en la clasificación, los cromosomas utilizan codificación binaria, el 1 indica la existencia de un atributo en el subconjunto de datos a utilizar. Una vez determinado el subconjunto de atributos, utilizan SVM para clasificar los registros. Obtienen la misma exactitud (80%) al clasificar utilizando tanto 279 atributos como solo 7 atributos, sin embargo, al utilizar 30 logran mayor exactitud (90%).

En [16] proponen utilizar k-vecinos más cercanos con una variación a su *kernel*, lo llaman *Kernel difference-weighted k-nearest Neighbor.* Utilizando muestreo aleatorio y validación

cruzada con 10-carpetas, realizan 10 experimentos y promedian los resultados, obteniendo así una exactitud de 70.66% en la clasificación, mejorando los resultados obtenidos utilizando el *kernel* tradicional.

En [17], proponen los métodos *mRMR* por sus siglas en inglés *mínimum Redundancy-Maximum Relevance* y C*mRMR* por sus siglas en inglés *Covariate mínimum Redundancy-Maximum Relevance,* cuya idea central es evitar la correlación entre atributos al durante el proceso de selección de los mismo. Antes de comenzar los experimentos, los datos son preprocesados, tratando datos faltantes y atípicos. Durante el proceso de clasificación se emplea *SVM* utilizando validación cruzada de 10-carpetas, por un lado, con el método *mRMR* la mayor exactitud (71.90%) de clasificación se obtienen al utilizar 40 atributos, por el otro, la más alta es de 72.12% al utilizar ya sea 25 o 30 atributos.

En [18], proponen utilizar redes neuronales modulares (*MNN* por sus siglas en inglés). Construyen modelos con distintos números de capas ocultas que van desde uno hasta tres, utilizan el algoritmo propagación hacía atrás para su entrenamiento. Los datos faltantes son reemplazados utilizando el valor de la columna más cercana de su misma clase. Al experimentar utilizan cinco diferentes razones de muestreo, pero la mayor exactitud de clasificación 82.22% la obtienen utilizando un muestreo de 90% de los datos para entrenamiento y el resto para prueba, utilizando una *MNN* con 2 capas ocultas. Es importante mencionar que se trata de clasificación binaria, solo determinan si el paciente sufre o no arritmia cardíaca.

En [19], utilizan *correlation-based feature CBF* con selección lineal hacia adelante para seleccionar el subconjunto de atributos. Para la clasificación utilizan redes neuronales con propagación incremental hacia atrás (*IBPLN* por sus siglas en inglés), logran obtener arriba de 87.71% de exactitud.

En [20], proponen un método para la construcción de clasificadores, a partir de probar combinaciones entre distintos subconjuntos de atributos y diferentes métodos de clasificación (Naïve Bayes, SVM, árboles de decisión y redes bayesianas). Proponen evaluar

cuatro subconjuntos de atributos: C_{All} que contiene todos los atributos, C_1 contiene veinticuatro, C_2 que contiene veintisiete y C_3 que contiene doce atributos. Obtienen la exactitud más alta (81.75%) a partir de combinar árboles de decisión con el subconjunto de atributos C_1.

En [21], al igual que en el trabajo pasado, dividen la metodología en dos partes, primero, utilizan *Fisher score* para reducir las dimensiones del conjunto de datos seleccionando un subconjunto de atributos. Para la segunda parte utilizan el clasificador SVM con mínimos cuadrados. Utilizando 65 atributos obtienen la más alta exactitud en la clasificación 82.09%.

Utilizando la herramienta de minería de datos *WEKA* (descrita en [22]). En [23], realizan combinaciones entre diferentes clasificadores con doble propósito: incrementar la exactitud en la clasificación y reducir el tiempo de cómputo de los experimentos. Utilizan los clasificadores *FT tree*, *J48 tree*, *BF*, Rep tree, *Random Forest* y *Simple Cart*, con una validación cruzada de 10-carpetas realizando Bagging y Adaboost, La más alta exactitud (86.60%) es lograda mediante *FT Bagging*.

En [24], proponen la implementación de un algoritmo para la detección de ECV en la nube, mediante una arquitectura que permite la aplicación de dicha tecnología a los pacientes con arritmias cardíacas. Apoyados en la herramienta de minería de datos *WEKA* (descrita en [22]) probaron cinco algoritmos, entre ellos, árboles de decisión (*J48),* clasificador basado en reglas (*Jrip*), clasificador basado en probabilidades (*Naïve Bayes*), *ANN* y *SVM*. *Naïve Bayes* arrojo la exactitud de clasificación más baja, 75.06%, mientras que la más alta se obtuvo con *J48,* 98.29%.

2.2.2 *Dataset*: problemas del corazón

A continuación, se describen brevemente trabajos relacionados al presente *dataset*.

Una red neuronal perceptron multi-capa se utiliza en [25] con una capa oculta, cuyo número de neuronas en esta capa se determina mediante el proceso de aprendizaje en cascada. La

ANN se entrena mediante el algoritmo de propagación hacia atrás. Los datos faltantes se reconstruyeron por la media de los datos pertenecientes a la misma clase. La exactitud alcanzada por el la ANN en la clasificación va desde 63.60% hasta 82.90%.

En [26], utilizan varias técnicas de minería de datos para realizar una clasificación binaria donde, determinan la presencia o ausencia de problemas del corazón. Entre las técnicas usadas están: *part C4.5*, *Naïve Bayes*, tabla de decisiones, *ANN*, *voted perceptron*, *SMO*, *RBF Gaussian, Repetead Inc Pruning* y *Kernel density*, en todas se utiliza validación cruzada con 10-carpetas. Es mediante la última técnica como se obtiene la exactitud más alta en la clasificación, 84.44%.

En [27], obtienen una exactitud de 87% preprocesando los datos mediante el método *k*-vecinos más cercanos. Un sistema de reconocimiento inmunológico artificial es utilizado como clasificador.

En [28], presentan un sistema híbrido que incluye ANN y FNN (*Fuzzy Neural Network*), ambas entrenadas mediante el algoritmo de propagación hacia atrás. Durante el entrenamiento utilizan validación cruzada con 10-carpetas. La exactitud de clasificación obtenida es 86.80%.

Apoyados en la herramienta de aprendizaje automático *Tanagra* en [29], implementan tres algoritmos: *Naïve Bayes*, listas de decisión y *k*-vecinos más cercanos. Utilizando validación cruzada con 10-carpetas y un muestreo de datos de 70% para entrenamiento y 30% para prueba obtienen exactitudes de 52.33%, 52% y 45,67%, con cada algoritmo respectivamente.

En [30], implementan un sistema de soporte en la toma de decisiones que utilizando el algoritmo *Naïve Bayes* analizando datos específicos de un paciente determina si este último sufre alguna enfermedad del corazón, la exactitud de los resultados va desde un 50% hasta 70%.

En [4], utilizan las herramientas *Tanagra* y *Weka* para experimentar con distintos algoritmos y distintos subconjuntos de atributos del conjunto de datos. La exactitud más alta se obtiene al combinar *ANN* con quince atributos, 100%, mientras que al utilizar árboles de decisión con los mismos atributos se obtiene 99.62%, sin embargo, al combinar arboles de decisión con algoritmos genéticos y seis atributos el resultado es 99.2%.

En [31], diseñan un sistema que indica el riesgo en un paciente de sufrir problemas del corazón. El sistema funciona en dos fases: primero se toman factores críticos (selección de atributos) y la información es dividida en 80% para entrenamiento y 20% para pruebas, en la siguiente fase, mediante ANN entrenada mediante propagación hacia atrás y reglas difusas determinan si el riesgo del paciente a padecer la enfermedad es bajo, medio o crítico. Para experimentar utilizan la herramienta *MatLab* descrita en [32].

En [33], realizan una comparación entre árboles de decisión (*C4.5 tree* y *CART tree*). Integran los cuatro dataset mencionados anteriormente en la sección 2.1.2 en diferentes combinaciones, a fin de reducir el desbalanceo de instancias en las clases, como, por ejemplo, en una integración agregan dos veces los datos del subconjunto *Cleveland*, dos veces los de *Hungarian*, cinco veces los de *Switzerland* y tres veces los de *VALong*. En cada uno de los experimentos el algoritmo *C4.5 tree* es mejor que *CART tree.*

Apoyados en la herramienta de minería de datos *WEKA*, en [23], realizan combinaciones entre diferentes clasificadores con el objetivo de incrementar la exactitud en la clasificación y reducir el tiempo de cómputo de los experimentos. Utilizan los clasificadores *FT tree*, *J48 tree*, *BF*, Rep tree, *Random Forest* y *Simple Cart*, con una validación cruzada de 10-carpetas realizando Bagging y Adaboost, La más alta exactitud (83.49%) es lograda mediante *Rep tree Bagging*.

2.2.3 *Dataset*: diabetes

A continuación, se describen brevemente trabajos relacionados al presente *dataset*.

En [28], presentan un sistema híbrido que incluye ANN y FNN (*Fuzzy Neural Network*), ambas entrenadas mediante el algoritmo de propagación hacia atrás. Durante el entrenamiento utilizan validación cruzada con 10-carpetas. La exactitud de clasificación obtenida es 84.20%.

En [34], revisan y modifican la metodología para la evolución de clasificación difusa. La arquitectura *eClass* es utilizada para actualizar de manera recursiva la base de reglas difusas, sin embargo, el resultado depende en gran medida del orden en los datos de entrada. La modificación antes mencionada consiste en aplicar una estrategia de optimización simple que, encuentre el orden en los datos de entrada que mejore los resultados que se obtienen. Después de la aplicación de dicha estrategia, se obtuvo una exactitud de clasificación de 79.33%.

En [35], realizan una mejora al sistema de reconocimiento inmune artificial (*AIRS* por sus siglas en inglés) y la llaman $AIRS_2$, posteriormente realizan una mejora a la última versión, donde, reemplazan el método del *k*-vecinos más cercanos por el método difuso *k*-vecinos más cercanos y lo llaman $MAIRS_2$. Utilizando validación cruzada con 10-carpetas obtienen las exactitudes de clasificación 82.69% y 89.10%, respectivamente.

En [36], proponen una mejora al algoritmo *k*-vecinos más cercanos (*KNN*), la cual se especializa en la reconstrucción de datos faltantes, puesto que consideran que el preprocesamiento de la información repercute de manera directa hacia los resultados que se obtienen. Aplicando validación cruzada con 10-carpetas obtienen 71.84% de exactitud en la clasificación utilizando el algoritmo *KNN* tradicional, mientras que utilizando el algoritmo propuesto obtienen 78.16%.

En [37], utilizando SVM como clasificador, validación cruzada con 10-carpetas obtienen una exactitud de 78% en la clasificación.

Utilizando la herramienta de minería de datos *WEKA* (descrita en [22]). En [23], realizan combinaciones entre diferentes clasificadores con doble propósito: incrementar la exactitud en la clasificación y reducir el tiempo de cómputo de los experimentos. Utilizan los clasificadores *FT tree*, *J48 tree*, *BF*, Rep tree, *Random Forest* y *Simple Cart*, con una validación cruzada de 10-carpetas realizando Bagging y Adaboost, La más alta exactitud (75.78%) es lograda mediante *FT Bagging*.

En [38], proponen un sistema hibrido para determinar si un paciente sufre o no diabetes, consta de tres partes. La primera parte es el preprocesamiento, donde remueven la inconsistencia de información, los datos faltantes lo reemplazan con la media del atributo, remueven el ruido mediante la técnica clusterización jerárquica *K-Means* (*K*-Means*) y *K-Means* en su forma tradicional. La segunda parte es la reducción de la dimensionalidad ejecutada mediante algoritmos genéticos (*GA* por sus siglas en inglés). La tercera parte es la clasificación, donde emplean SVM. Las combinaciones realizadas y la exactitud en clasificación obtenida se enlistan a continuación:

- *K-Means* + *GA* + *SVM,* 95.94%.
- *K*-Means* + *GA* + *SVM,* 97.59%.

El trabajo presentado en [39] es similar al anterior. Durante el preprocesamiento reemplazan los datos faltantes con la media del atributo. Apoyados en la herramienta de minería de datos *Weka* mediante el método *K-Means*, reducen las dimensiones del conjunto de datos. Para la clasificación utilizan nuevamente *Weka* para construir el árbol de decisión *J48* utilizando validación cruzada con 10-carpetas, logrando clasificar con una exactitud de 90.04%

CAPITULO III. MARCO TEÓRICO

3.1 Aprendizaje automático

Según [40], aprendizaje automático se trata de explorar y desarrollar modelos matemáticos y algoritmos para aprender de la información existente; además menciona que, la clasificación también se denomina como aprendizaje supervisado, el cual necesita de un conjunto de datos etiquetados que se dividen en tres subconjuntos (de entrenamiento; de validación; de prueba) los cuales son utilizados en el proceso del aprendizaje automático, en sus siguientes fases:

- Entrenamiento: se encuentran los parámetros óptimos del modelo, a partir del subconjunto de datos etiquetados de entrenamiento. Se trata de un algoritmo para entrenar al modelo, en otras palabras, estima, aproxima y optimiza los parámetros del modelo de aprendizaje automático, a partir del conjunto de datos etiquetados, el dominio de características y las características de las clases [40].
- Validación: evita el sobre-entrenamiento del modelo, mediante el uso del subconjunto de datos etiquetados de validación. Se trata de un algoritmo para validar la efectividad del modelo, usualmente se realiza mediante el entropía y error mínimo cuadrado.

  La técnica que más se utiliza es validación cruzada con n-carpetas [41], [42]. Para el enfoque de validación cruzada con 10-carpetas, el subconjunto de datos de entrenamiento se divide en diez carpetas[40], las cuales se denominan: *fold1*, *fold2*, ..., *fold10*, tal como se muestra en la Figura 6. Para el primer paso se toman los primeros nueve *folds* para entrenar al modelo, y el *fold10* se usa para evaluar al modelo; en el segundo paso se usa el *fold9* para evaluar al modelo y el resto para realizar el entrenamiento del mismo; en el tercer paso se usa el *fold8* para evaluar al modelo y el resto para entrenar al mismo; el proceso continua sucesivamente hasta realizar 10 iteraciones, como se muestra en la Figura 6. El error mínimo cuadrado es una forma de medir la efectividad del modelo.

- Prueba: determina la exactitud del modelo, utilizando el subconjunto de datos etiquetados de prueba. Se trata de un algoritmo que verifica si los algoritmos de entrenamiento y validación cruzada funcionan con el conjunto de datos no utilizado en las fases anteriores, y realiza una comparación entre los resultados arrojados por el modelo final (modelo obtenido de las fases entrenamiento y validación) y la etiqueta (clase) de los datos, obteniendo los siguientes parámetros de medición: exactitud, sensibilidad, especificidad y precisión [40].

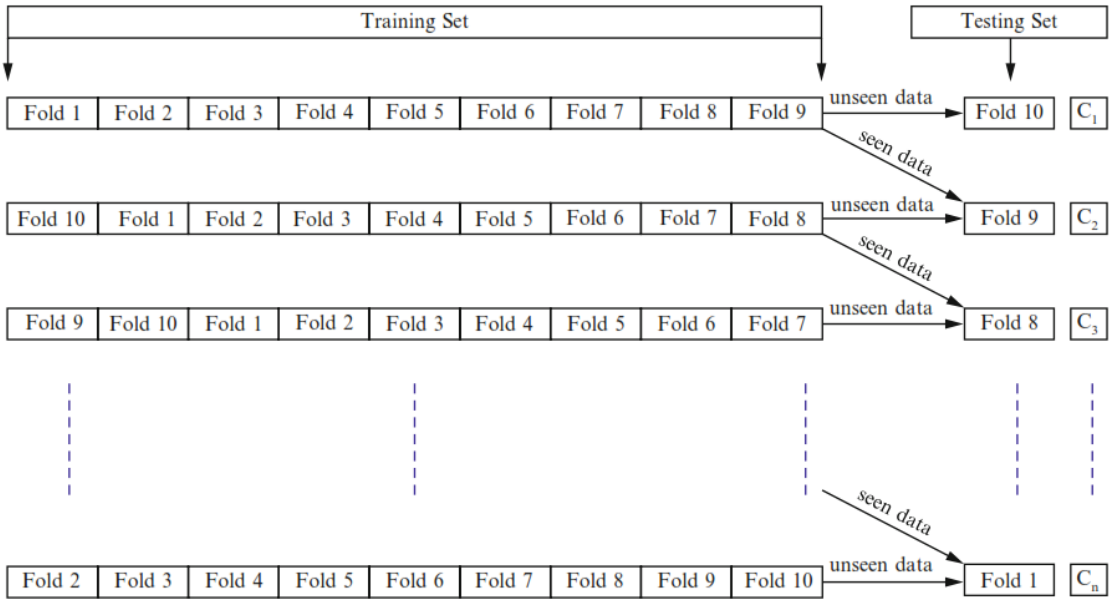


Figura 6. Uso común de la validación cruzada con 10-carpetas. Tomada de [40].

Las aplicaciones del aprendizaje automático han aumentado en los últimos años debido a que resuelve problemas de distintas áreas. Algunos ejemplos de aplicación del aprendizaje automático son [43]:

- Traducción automática de documentos.
- Control de acceso mediante reconocimiento facial.
- Reconocimiento de voz.
- Clasificación de correos electrónicos como *spam*.

3.1.1 Aprendizaje supervisado

Las clases (etiquetas) juegan un papel importante al momento de diferenciar el aprendizaje supervisado del no supervisado. En el primero, la información se encuentra previamente etiquetada [40], dicho de otra manera, cada registro pertenece a una clase como se muestra en la Figura 7, y los límites de las clases se encuentran definidos como se muestra en la Figura 8.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	class
2	63	1	1	145	233	1	2	150	0	2.3	3	0	6	0
3	67	1	4	160	286	0	2	108	1	1.5	2	3	3	2
4	67	1	4	120	229	0	2	129	1	2.6	2	2	7	1
5	37	1	3	130	250	0	0	187	0	3.5	3	0	3	0
6	41	0	2	130	204	0	2	172	0	1.4	1	0	3	0
7	56	1	2	120	236	0	0	178	0	0.8	1	0	3	0
8	62	0	4	140	268	0	2	160	0	3.6	3	2	3	3
9	57	0	4	120	354	0	0	163	1	0.6	1	0	3	0
10	63	1	4	130	254	0	2	147	0	1.4	2	1	7	2
11	53	1	4	140	203	1	2	155	1	3.1	3	0	7	1
12	57	1	4	140	192	0	0	148	0	0.4	2	0	6	0
13	56	0	2	140	294	0	2	153	0	1.3	2	0	3	0
14	56	1	3	130	256	1	2	142	1	0.6	2	1	6	2
15	44	1	2	120	263	0	0	173	0	0	1	0	7	0
16	52	1	3	172	199	1	0	162	0	0.5	1	0	7	0
17	57	1	3	150	168	0	0	174	0	1.6	1	0	3	0
18	48	1	2	110	229	0	0	168	0	1	3	0	7	1
19	54	1	4	140	239	0	0	160	0	1.2	1	0	3	0
20	48	0	3	130	275	0	0	139	0	0.2	1	0	3	0

Figura 7. Conjunto de datos etiquetados, donde cada registro pertenece a una clase.

Cuando el conjunto de clases C contiene solo dos elementos, entonces se trata de un problema de clasificación binaria (ver la Figura 8a) clasificación binaria): $C = \{amarillo, rojo\}$; en cambio, cuando el conjunto de clases C contiene tres o más elementos, entonces se trata de un problema de clasificación multi-clase (ver la Figura 8b) clasificación multiclase): $C = \{amarillo, rojo, azul\}$.

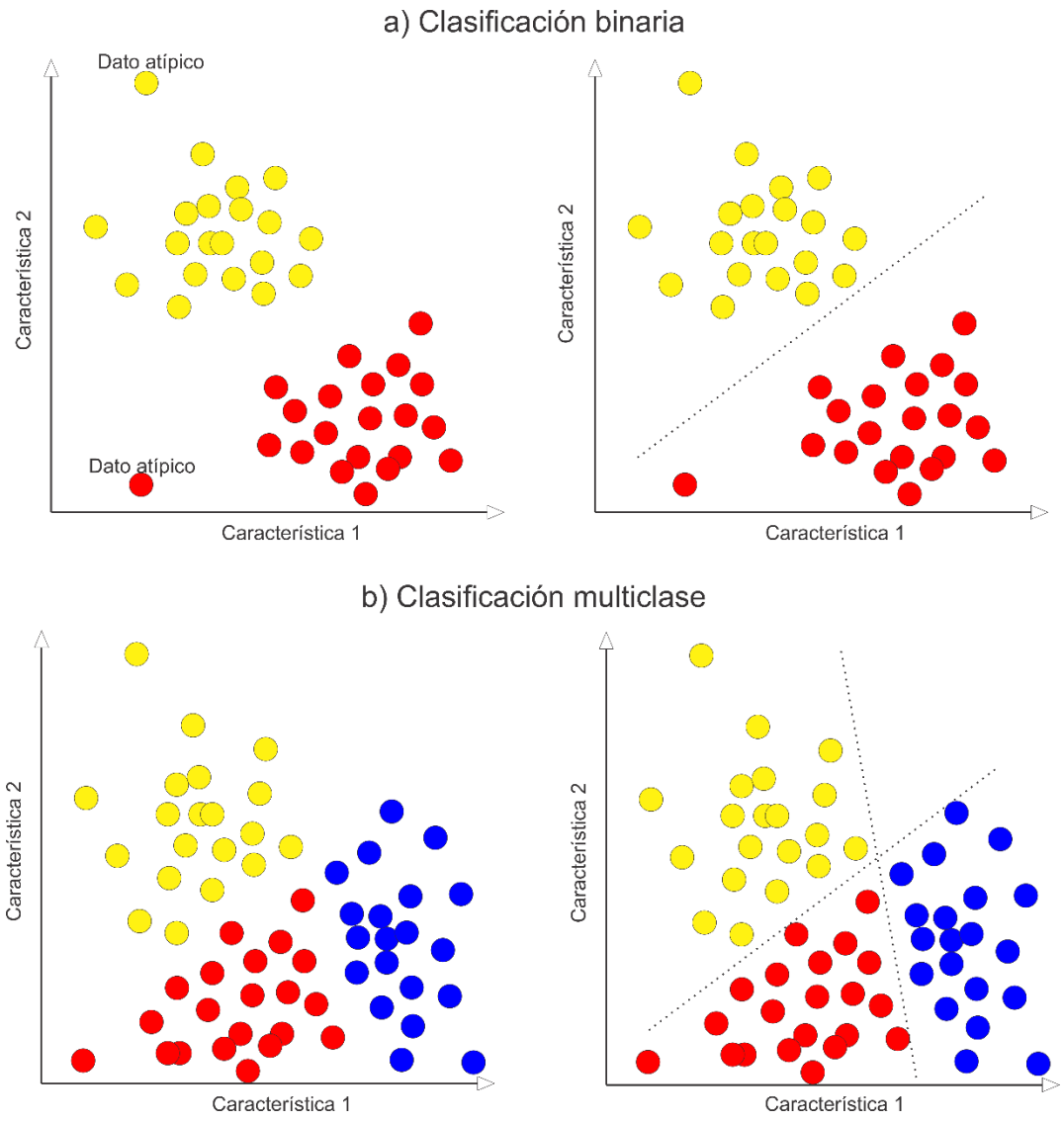


Figura 8. Tipos de clasificación.

3.1.2 Evaluación del modelo

Según [44], existen diferentes maneras de medir el desempeño de un modelo de clasificación, las más populares son: matriz de confusión, log-loss, AUC y exactitud, la cual se expresa en (1).

$$exactitud = \frac{\#\ de\ predicciones\ correctas}{\#\ total\ de\ instancias} \quad (1)$$

La exactitud es una buena métrica de evaluación, sin embargo, no realiza mediciones entre clases, mediciones que también son importantes, por ejemplo: cuando un doctor diagnostica a un paciente un problema del corazón y el paciente no lo padece (esto se conoce como falso positivo).

Según [44], [45], partiendo de una matriz de confusión (también conocida como tabla de confusión) podemos calcular diferentes métricas de medición detallada para cada clase, como, por ejemplo: exactitud, sensibilidad, especificidad, valor predictivo positivo, valor negativo predictivo. La Tabla 4 muestra la manera en que se integra una matriz de confusión, las columnas representan a las etiquetas reales de las instancias, mientras que, por otro lado, las filas representan los resultados arrojados por el modelo en evaluación.

		Referencia	
		P	N
Clasificación	Y	**T**rue **P**ositives	**F**alse **P**ositives
	N	**F**alse **N**egatives	**T**rue **N**egatives
Total, por columna:		**P**	**N**

Tabla 4. Matriz de confusión. Tomada de [45].

Cada registro clasificado se contabiliza dentro de alguna de las celdas de la matriz de confusión (ver la Tabla 4) de acuerdo a lo siguiente:

- Celda **T**rue **P**ositives (TP): registros clasificados como positivos y que realmente son positivos, es decir, cuando al paciente le diagnostican problema del corazón y realmente lo padece.

- Celda **T**rue **N**egatives (TN): registros clasificados como negativos y que realmente son negativos, es decir, cuando al paciente le diagnostican ausencia de problema del corazón y realmente no la padece.
- Celda **F**alse **P**ositives (FP): registros clasificados como negativos y que en realidad son positivos, es decir, cuando al paciente le diagnostican sano, pero padece problema del corazón.
- Celda **F**alse **N**egatives (FN): registros clasificados como positivo y que en realidad es negativo, es decir, cuando al paciente le diagnostican problema del corazón y en realidad es sano.

A partir de la matriz de confusión se pueden calcular las siguientes métricas de evaluación:

- $Exactitud$, es la probabilidad de clasificar correctamente un registro:

$$Exactitud = \frac{TP + TN}{TP + FN + FP + TN} \quad (2)$$

- $Sensibilidad$, también se conoce como *recall*. Es la probabilidad de que la prueba clasifique un registro en una clase específica, dado que realmente pertenece a ella, por ejemplo, que el modelo clasifique a un paciente como enfermo, dado que realmente padece la enfermedad:

$$Sensibilidad = \frac{TP}{TP + FN} \quad (3)$$

- $Especificidad$, es la probabilidad de que, dado que el paciente no padece la enfermedad, el modelo lo clasifique como tal:

$$Especificidad = \frac{TN}{FP + TN} \quad (4)$$

- $Valor\ positivo\ predictivo$, también conocido como $Precision$, es la probabilidad de que, si el modelo ha clasificado al paciente como enfermo, el paciente realmente lo esté:

$$\text{Valor positivo predictivo} = \frac{TP}{TP + FP} \quad (5)$$

- $Valor\ negativo\ predictivo$, es la probabilidad de que, si el modelo ha clasificado al paciente como sano, el paciente realmente lo esté:

$$\text{Valor negativo predictivo} = \frac{TN}{FN + TN} \quad (6)$$

En resumen, los registros de las celdas ***T**rue* **N**egatives y ***T**rue **P**ositives* contienen los registros que han sido clasificados correctamente, caso apuesto de las celdas **F**alse ***N**egatives* y ***F**alse **P**ositives* que contienen los registros clasificados incorrectamente.

3.1.3 Preprocesamiento de datos

La preparación y limpieza de la información es un paso sumamente importante previo al aprendizaje automático donde, el objetivo es obtener al final un *dataset* (conjunto de datos) con información consistente que permita realizar un aprendizaje automático sin problemas. Comienza con el proceso *ETL* (Extracción-Transformación-Carga) y continua con el preprocesamiento de los datos [46], [47]. Algunos de los procesos del preprocesamiento se mencionan en esta sección.

3.1.3.1 Balanceo de las clases

La (ver página 9) es un claro ejemplo de desbalanceo en el número de instancias por clase, por un lado, la clase 1 contiene 245 instancias, por otro, la clase 8 solo contiene 2 instancias, ¡existe una gran diferencia! La clasificación del modelo en uso puede incurrir en clasificaciones sesgadas hacia la clase con mayor número de instancias, en consecuencia, es necesario implementar medidas que brinden de alguna manera un mejor balanceo al dataset, tanto en la distribución de las instancias sobre las clases como en las dimensiones (cantidad de instancias y atributos) del dataset, por ejemplo:

- Utilizar un subconjunto de atributos.

- Omitir clases con pocas/muchas instancias.
- Generalizar clases, dicho de otra manera, agrupar instancias en una clase más general.

3.1.3.2 Tratamiento de datos atípicos

Podemos considerar a un dato como atípico, siempre y cuando se encuentre muy distante del resto de los demás datos, comúnmente son generados por errores como: mezcla de datos, confusión entre medidas (por ejemplo, kilómetros y metros) o lectura errónea por parte de un sensor, no obstante, no se trata de un dato inválido; los datos atípicos deben ser identificados e investigados [48]. La Figura 9 muestra dos datos atípicos.

Por lo que se refiere a la identificación de los datos atípicos, [49] mencionan varios datos estadísticos que nos ayudan a comprender el comportamiento de los datos como, por ejemplo:

- Estadísticos de centralización (media, mediana y moda): que proporcionan información sobre los valores centrales de los datos.
- Estadísticos de dispersión (rango, varianza muestral, desviación estándar, cuartiles y los rangos intercuartílicos): que proporcionan información sobre la variabilidad del conjunto de datos. Este tipo de estadísticos pueden ser representados de manera gráfica mediante el diagrama de caja y bigotes (*boxplot*), histogramas, diagrama de barras, diagrama de violín, entre otros.

Algunas métricas para el tratamiento de los datos atípicos son:

- La mediana: se considera mejor que la media, debido a que la mediana no se ve influenciada por datos atípicos.
- Puntos de cortes: cortar por el ejemplo, el 10% (de los valores mayores y menores) provee protección contra la aparición de datos atípicos, sin embargo, esta métrica reduce la cantidad de instancias del dataset.

La presencia de un dato atípico puede estar asociada a ruido o a datos informativos, por consiguiente, entender el comportamiento de los datos es esencial para dar la correcta interpretación y tratamiento de estos.

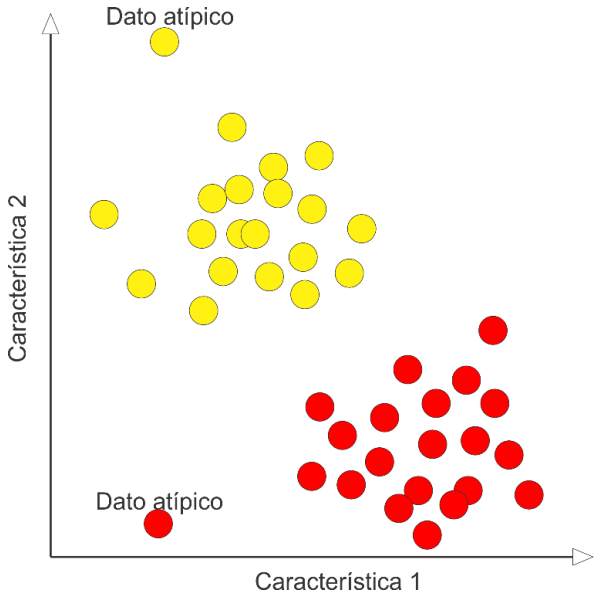


Figura 9. Datos atípicos.

3.1.3.3 Tratamiento de datos perdidos (faltantes)

Los datos faltantes son una de las inconsistencias más comunes [46], también son conocidos como *NA* (*Not Available*). La Figura 10, muestra varios datos faltantes los cuales pueden ser tratados en diferentes maneras, por ejemplo:

- Eliminar la instancia que lo contiene.
- La imputación, que consiste en remplazar el dato faltante por un sustituto calculado mediante algún modelo de predicción (por ejemplo: regresión), valores observados en registros con características similares (por ejemplo: vecinos cercanos), algunas medidas estadísticas (por ejemplo: mediana), o bien puede ser construido por un experto en el área [46], [50].

P	QRST	J	Heart rate	Channel D	Channel D
64	-2		63	0	52
-17	31		53	0	48
70	66	23	75	0	40
-5	20		71	0	72
61	3		?	0	48
52	88		84	0	36
75	65		70	0	44
8	51		67	0	44
78	66	84	64	0	40
70	71		63	0	44
68	72		70	20	36
49	?		72	0	40
41	-13		73	0	72
60	63		56	0	92
45	40		72	0	80
68	40		76	0	48
77	75		67	0	48
-2	54		70	0	48

Figura 10. Datos faltantes (perdidos).

3.1.3.4 Normalización de datos

Un *dataset* puede estar compuesto por información de distintos formatos, medida en diferentes rangos, dicha información puede ser cuantitativa o cualitativa, la idea central es que, la información del *dataset* puede ser muy variada.

La Figura 11 y Figura 12, muestran la edad de diez personas y el salario de las mismas personas, respectivamente. Por un lado, los valores de edad van desde 21 hasta 60, por otro, el salario tiene valores desde $10{,}000\ hasta\ 48{,}000$. Al introducir los valores de los atributos (variables) edad y salario en crudo, el modelo clasificador puede ser influenciado en mayor proporción por los valores tan altos de los salarios, para evitar dicha situación, es necesario preparar e integrar la información del *dataset* [47]. La solución es transformar la información dentro de un formato/esquema común, es decir normalizar o estandarizar la información de cada atributo (variable) a un intervalo común, por ejemplo $[-1{,}1]$ o $[0{,}1]$.

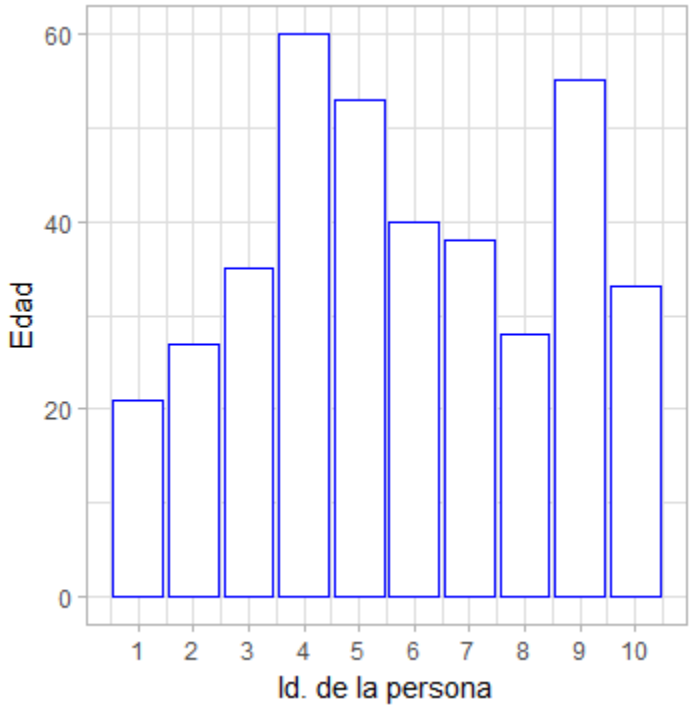


Figura 11. Edad de 10 personas.

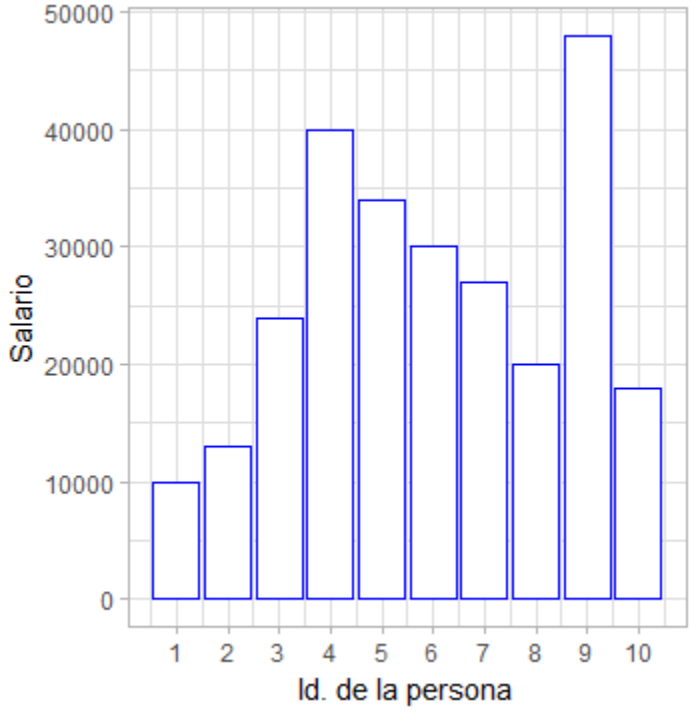


Figura 12. Salario de 10 personas.

3.1.4 Selección de atributos

Son tres objetivos de la selección de variables (atributos):

1. Mejorar la predicción/clasificación del modelo.
2. Proporcionar modelos de predicción/clasificación más rápidos y más rentables.
3. Facilitar y mejorar el entendimiento de los datos.

Incluir todos los atributos al modelo no garantiza obtener los mejores resultados. Existen atributos que aportan información redundante (información que tiene el mismo significado), atributos que aportan poca o mucha información, en consecuencia, se debe determinar los atributos a utilizar, de tal manera que se preserve de manera sustancial la mayor parte de la información. En [51] recopilan información acerca de los enfoques para la selección de atributos y proponen el cuestionario para determinar el enfoque a usar, algunos enfoques mencionados son:

1. Filtros: el *ranking* de los atributos es el principal elemento para seleccionar los mismos. Algunos métodos utilizados para construir el *rank* son: *T-test*, criterios *Fisher*, R^2, en regresión, por ejemplo.
2. *Wrapper*: utilizan aprendizaje automático dentro de una caja negra para calificar subconjuntos de variables acorde a su poder de predicción. Es un problema NP-duro por las combinaciones posibles, usan fuerza bruta (selecciona hacia adelante, eliminación hacia atrás)
3. Métodos embebidos: seleccionan subconjuntos de variables mediante un proceso de entrenamiento y usualmente son específicos para el algoritmo de aprendizaje seleccionado. Se puede dividir la información en entrenamiento y prueba y usar validación.
4. Métodos anidados: tienen una función objetivo a buscar en un espacio de soluciones, se utiliza por ejemplo SVM.
5. Construir subconjuntos de atributos más compactos: sin omitir atributo alguno, compactan la información. *PCA*, *SVD*, y *LDA*, son algunos ejemplos.

3.2 Red Neuronal Artificial (RNA)

El cerebro humano puede ser visto como una computadora sumamente compleja, no lineal y paralela (un sistema de procesamiento de información), que puede organizar sus neuronas de tal manera que mejore el reconocimiento de patrones, la percepción, y el control del cuerpo, además, en muchas ocasiones, puede procesar información más rápido que la computadora digital [52]. En definitiva, el cerebro humano es un órgano asombroso y muy complejo que puede reconocer patrones, clasificar información e intuir situaciones, a gran velocidad, todo ello a partir de la información.

La RNA es una técnica de la inteligencia artificial inspirada en un proceso de la naturaleza, que trata de emular matemática y computacionalmente el funcionamiento del cerebro humano.

Una RNA es un computador paralelo distribuido masivamente, compuesto de unidades simples de procesamiento (neuronas), el cual de manera natural almacena conocimiento experimental y lo mantiene disponible para usarlo. Se parece al cerebro humano en dos aspectos:

1. Adquiere el conocimiento a partir del ambiente mediante un proceso de aprendizaje.
2. Las fortalezas de conexión interneurona, conocidas como pesos sinápticos, se usan para almacenar conocimiento.

El procedimiento para realizar el proceso de aprendizaje se conoce como algoritmo de aprendizaje [52]. La unidad más básica del cerebro humano es la neurona, mientras que en una RNA la unidad más básica se conoce como perceptrón.

3.2.1 Perceptrón

El perceptrón es la forma más simple (básica) de la RNA usado para la clasificación de patrones. Básicamente, consiste en una sola neurona con pesos sinápticos ajustables (mediante un algoritmo de aprendizaje) y el *bias* [52].

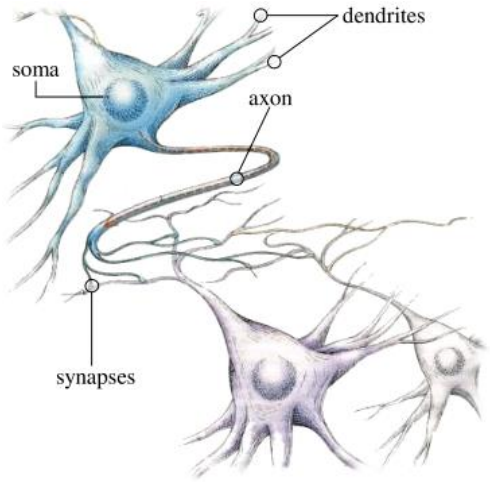


Figura 13. Anatomía de una neurona, estructura y conexión. Tomada de [53].

La Figura 13 muestra las partes de una neurona, [52], [54] exponen que, las dendritas permiten la entrada de señales (información) hacia la parte del cuerpo (núcleo o *soma*, multiplicando cada señal de acuerdo a la importancia de cada señal (sinapsis). En el cuerpo, se integran las señales, y dependiendo, si las señales integradas exceden cierto límite, una nueva señal se transmite (o no) a través del axón.

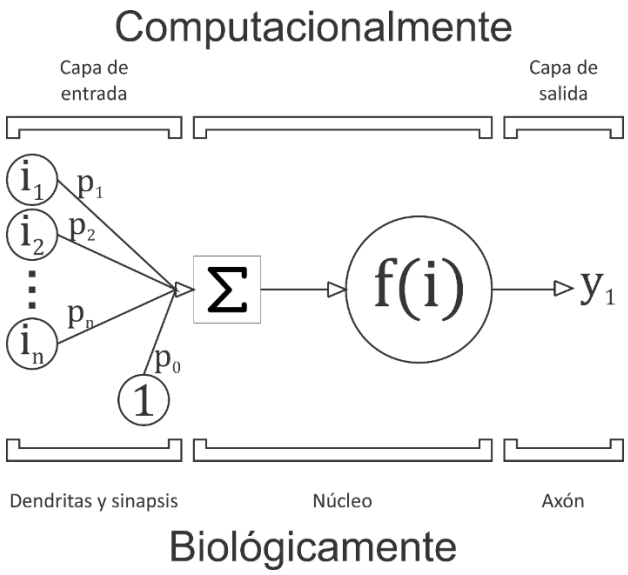


Figura 14. Representación gráfica del perceptrón. Comparación entre neurona biológica y perceptrón.

De la Figura 14, podemos destacar dos puntos importantes:

1. Representa gráficamente los elementos del perceptron: capa de entrada, pesos, sumatoria, función de activación y capa de salida. La representación matemática se describe en (7) y en (8) de manera matricial.
2. Relaciona las partes del perceptrón con las partes de la anatomía de la neurona mostradas en la Figura 14.

$$y_1 = f\{(\sum_{j=1}^{n} i_j p_j) + p_0\} \tag{7}$$

Donde:

- y_1: es el valor resultante (salida) del perceptrón;
- n: es el número de elementos contenidos en $\bar{\imath}$;
- i_j: es el valor de entrada j de $\bar{\imath}$;
- p_j: es el peso correspondiente al valor de entrada j;
- p_0: es el peso correspondiente al *bias*;
- $f\{\}$: representa la función de activación;

$$y_1 = f\{\bar{\imath}^T \bar{p} + p_0\}$$

Donde: (8)

- y_1: es el valor resultante (salida) del perceptrón;
- $\bar{p}$: es vector de pesos;
- $\bar{\imath}$: es el vector de entradas;
- p_0: es el peso correspondiente al *bias*;
- $f\{\}$: representa la función de activación;

La función de activación determina el valor final que arroja una neurona, es seleccionada de acuerdo al tipo de problema enfrentado. Las más comunes se muestran en la Figura 15.

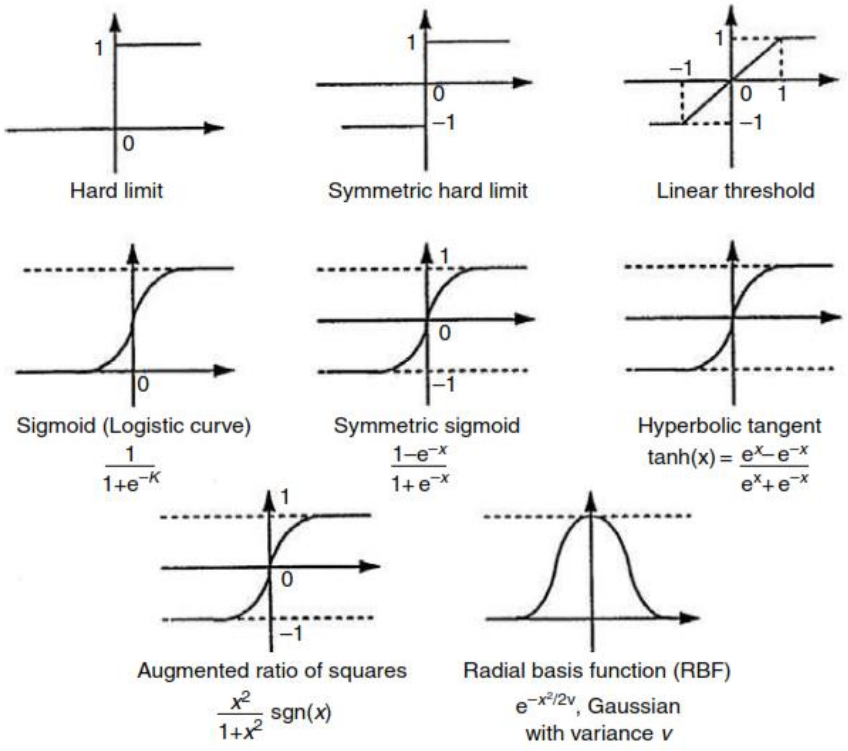


Figura 15. Funciones de activación comúnmente utilizadas en RNA. Tomada de [54].

3.2.2 Arquitectura de una RNA

A la unión de dos o más perceptrones (neuronas) para abordar un problema se le conoce como RNA. La manera de estructurarse y conectarse entre las neuronas, y el algoritmo de aprendizaje utilizado por la RNA se conoce como arquitectura de la RNA, existen tres tipos de arquitecturas [52]:

1. Red de capa-única alimentada hacia adelante: todas las neuronas de la red son estructuradas en una única capa de salida, y no cuenta con capas ocultas. Ver la .

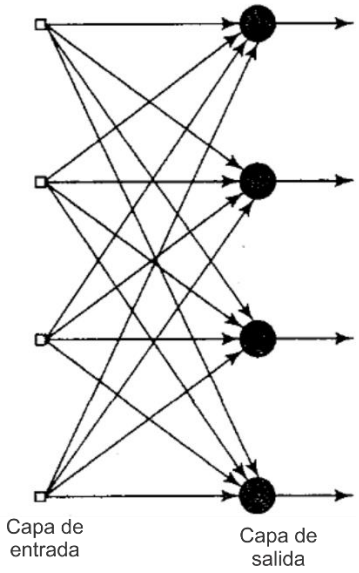


Figura 16. RNA de capa-única alimentada hacia adelante, totalmente conectada. Tomada de [52].

2. Red multi-capa alimentada hacia adelante: este tipo de arquitectura se distingue por contener una o más capas ocultas con la presencia de neuronas ocultas en cada una de ellas (ver Figura 17). Las capas ocultas intervienen entre la capa de entrada y la capa de salida y son útiles en distintas maneras, por ejemplo: (Churchland y Sejnowski, 1992) sostienen que, gracias a las capas ocultas, la red adquiere una perspectiva global en un sentido bastante amplio.

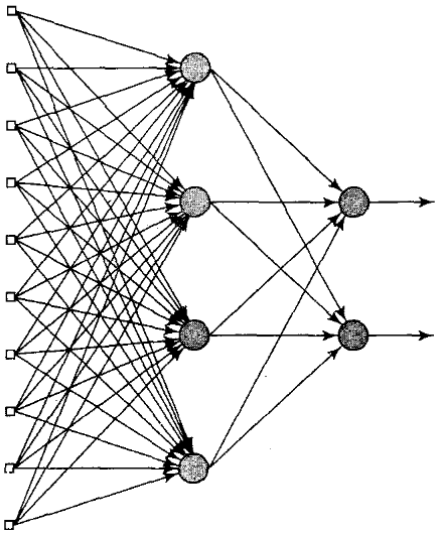


Figura 17. RNA multi-capa alimentada hacia adelante, totalmente conectada. Contiene una capa oculta con cuatro neuronas ocultas. Tomada de [52].

1. Red recurrente: este tipo de redes se caracterizan por tener un ciclo de alimentación hacia atrás, tal como se muestra en la Figura 18.

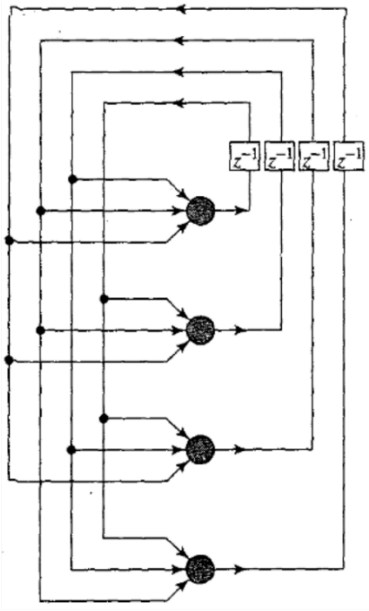


Figura 18. RNA recurrente. Tomada de [52].

3.2.3 Aprendizaje

(Fischler and Firschein, 1987) definen de manera genérica "conocimiento" como [52]:

> *Conocimiento se refiere a almacenar información o modelos usados por una persona o máquina para interpretar, predecir, y responder apropiadamente al mundo exterior.*

Aprender (generar conocimiento) es el paso más importante de una RNA, consiste en ajustar los pesos de cada neurona (incluidos los pesos del *bias*) de la red durante un número definido o indefinido de iteraciones, denominadas épocas. (Mendel y McClaren, 1970), definen aprendizaje en el contexto de RNA como [52]:

> *Aprender es un proceso mediante el cual los parámetros libres de una red neuronal son adaptados durante un proceso de simulación por el entorno en el que se integra la red. El tipo de aprendizaje es determinado por la manera en que los cambios en los parámetros toman lugar.*

Esta definición implicada tres secuencias de eventos:

1. La RNA es simulada por el entorno (problema).
2. La RNA sufre cambios en sus parámetros como resultado de dicha simulación.
3. La RNA responde en una manera nueva al entorno debido a los cambios ocurridos en su estructura interna.

El proceso de aprendizaje se realiza mediante los siguientes pasos:

1. Alimentación hacia adelante (*FeedForward*): consiste calcular el resultado de cada neurona, aplicando la ecuación (7), comenzando por las neuronas de la capa de entrada, continuando por las neuronas de cada capa oculta en orden ascendente y terminando con las capas de la capa de salida.
2. Aprendizaje: consiste en calcular el error en la salida de cada neurona de la capa de salida, para después corregir cada peso de acuerdo con el algoritmo de aprendizaje

elegido. En el presente proyecto utilizamos el algoritmo “propagación hacia atrás (BackPropagation)”.

3. Se repiten los pasos 1 y 2, hasta cumplir el número de épocas predefinido por el usuario o hasta que la RNA converge, lo que suceda primero. Se asume que la RNA converge cuando error total es menor al umbral δ, comúnmente establecido por el usuario. La suma del error de cada una de las neuronas de la capa de salida se calcula como:

$$E_{total} = \sum_{k=1}^{m} E_k \quad (9)$$

Donde:

- m: es el número de neuronas en la capa de salida;
- E_k: es el error calculado en la neurona k;

3.2.3.1 Propagación hacia atrás (BackPropagation)

El algoritmo de aprendizaje utilizado en el presente proyecto es el algoritmo de propagación hacia atrás, mismo que fue cual fue creado por (Rumelhart, Geoffrey y Ronald, 1986) y se describe en [55].

Este algoritmo actualiza los pesos de las conexiones entre neuronas calculando la derivada del error total (ver ecuación (9)) respecto del peso a actualizar, tal como se muestra en la ecuación (10). Para poder calcular la parte $\frac{\partial E_{total}}{\partial p_{jk}}$ de la ecuación (10), se utiliza la regla de la cadena.

$$p_{jk} = p_{jk} - \alpha * \frac{\partial E_{total}}{\partial p_{jk}} \quad (10)$$

Donde:

- p_{jk}: es el peso j de la capa k a actualizar;
- α: es la tasa de aprendizaje;

- ∂E_{total}: es la suma de los errores de cada neurona en la capa oculta, ver ecuación (9);

Cada peso unidad oculta, ya sea peso o neurona, es afectada por las unidades de la siguiente capa con las que se relaciona. Para más detalles del algoritmo propagación hacia atrás consulte [52], [56], [57], un ejemplo explicado paso a paso lo encuentra en [58].

Una vez terminado el proceso de aprendizaje, se realizan evaluaciones al modelo con el fin de medir el grado de conocimiento que el modelo posee sobre el entorno. Una técnica utilizada para realizar dichas mediciones es la matriz de confusión (ver 3.1.2). Dichas mediciones se realizan evaluando cada registro contenido en el subconjunto de prueba.

3.3 Algoritmos genéticos (AG)

Los algoritmos genéticos son una técnica adaptativa de la IA que puede usarse para resolver problemas No Polinomiales (NP) del mundo real, es decir, problemas donde el costo computacional requerido para su solución crece exponencialmente conforme aumenta la complejidad del problema, específicamente en problemas de combinatoria, búsqueda y optimización [59]. Los principios básicos de AG fueron establecidos por Holland en 1975 [59].

AG es una heurística de exploración del universo de soluciones (espacio de búsqueda) en busca de la optimización. *AG* se encuentran dentro de la computación evolutiva y son una rama de la IA, esta herramienta está inspirada en la teoría de la evolución de Charles Darwin, es decir, en el proceso de la evolución natural de los seres vivos (el más adaptado el medio, sobrevive) [59]. El elemento principal de los AG es el individuo (también conocido como cromosoma) compuesto por genes que contienen un valor codificado por el usuario, de acuerdo con las necesidades del problema abordado (algunos ejemplos de tipos de codificación son: binaria, flotante, entera. Ver la Figura 20), cada gen puede tomar solo un valor del alfabeto que rige de igual manera a cada uno de los individuos, tal como se

muestra en la Figura 19 y Figura 20; cada individuo propone una posible solución al problema abordado; los individuos forman una población que es iterada en el tiempo para dar paso a mejores individuos a través de las generaciones, forzando de esta manera la evolución en cuanto a optimización de los individuos como lo explica [60].

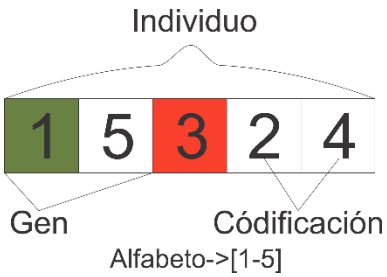


Figura 19. Partes de un individuo (cromosoma).

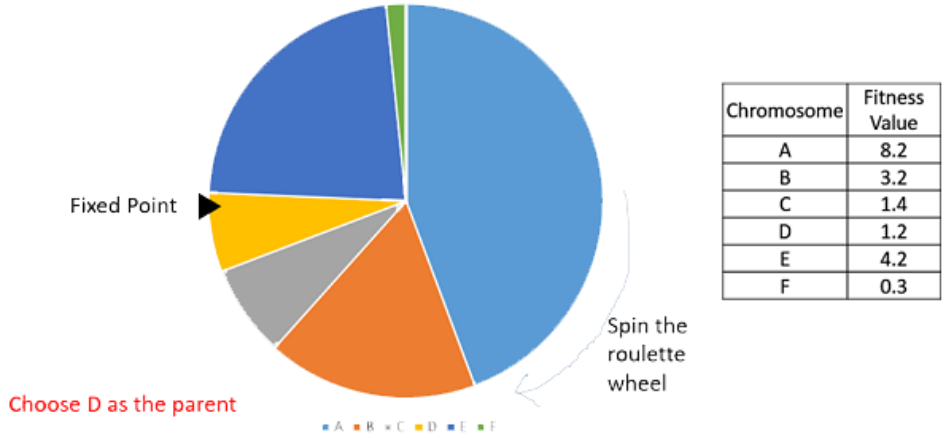
Figura 20. Ejemplos de diferentes codificaciones de un individuo (cromosoma).

La forma de trabajar de *AG* consiste en generar una población inicial de *n* individuos a la cual se le aplican los *operadores genéticos* (*evaluación, selección, cruza y mutación*).

- *Evaluación:* también conocida como función de evaluación; consiste en calcular la aptitud de un individuo, es decir, asignar una calificación (número real) a un individuo [28] y [29]. Esta función siempre se programa por el usuario y debe ser adaptada de acuerdo con el problema abordado, hay que tener en cuenta que se debe aplicar el mismo criterio de evaluación para cada individuo. Una manera de hacer valer las restricciones del problema (en caso de haber) consiste en penalizar

al individuo que las incumple. De esta forma se preparan a todos los individuos de la generación en curso para aplicar el operador *selección*.

- *Selección*: "Supervivencia del más apto" (Charles Darwin). Como se menciona en [59], [60], este operador consiste en seleccionar a los padres (progenitores) para realizar la cruza de los mismos y obtener a los individuos (hijos) que poblarán la siguiente generación. Existen diferentes métodos para seleccionar a los individuos padres:
  - *Rueda de la ruleta:* es el método más usado. La suma de las aptitudes es el 100% de la ruleta y a cada individuo le toca una porción proporcional a su aptitud, dicho de otra manera, a mayor aptitud de un individuo, mayor proporción en la ruleta, y a mayor proporción en la ruleta mayor probabilidad de ser elegido, sin embargo, este método no implica que el individuo con mayor probabilidad sea seleccionado, puesto que se trata de una selección aleatoria [59], [60]. En la Figura 21 se muestra de manera gráfica que el individuo A posee mayor probabilidad de ser elegido, sin embargo, al hacer girar la ruleta, el individuo D fue el seleccionado.



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Figura 21. Método de selección por rueda de la ruleta.

  - *Por rango:* primeramente, se asigna un rango numérico en base a la aptitud de cada individuo, posteriormente, se ordena descendentemente a los individuos por rango, por último, se toma a los

mejores individuos [59], [60]. En la Figura 22 se muestra de manera gráfica que los individuos se ordenan descendentemente por rango (de acuerdo con su aptitud) y se seleccionan los cuatro mejores (los cromosomas b, f, d y g).

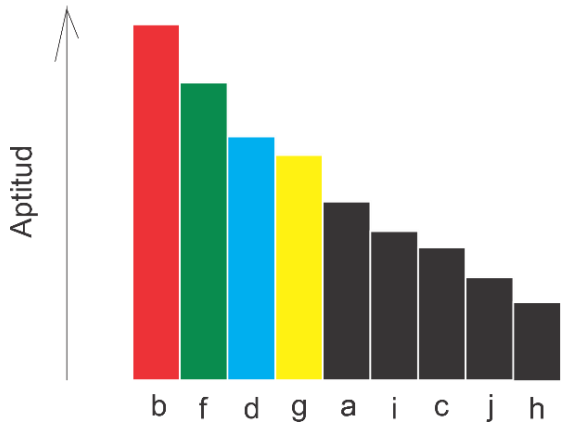


Figura 22. Método de selección por rango.

- *Elitista:* toma a los mejores individuos de la generación actual y además los conserva en la siguiente generación [59], [60]. En la Figura 23 se observa como los cromosomas (individuos) rojo, verde, azul y amarillo son seleccionados como progenitores y además son conservados en la siguiente generación.

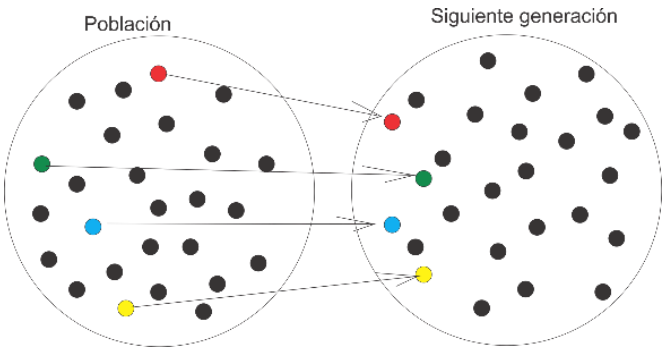


Figura 23. Método de selección elitista.

- *Por torneo:* toma individuos aleatoriamente, posteriormente, compiten entre ellos para ser seleccionados [59], [60]. En la Figura 24 se muestra

de manera gráfica que los individuos: rojo, verde, azul y amarillo fueron tomados aleatoriamente para después competir para ser seleccionados.

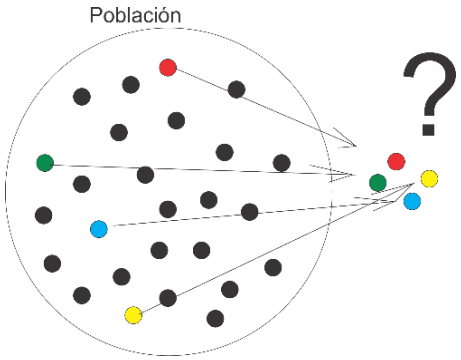


Figura 24. Método de selección por torneo.

En la mayoría de los métodos de selección, los individuos son tomados en base a su aptitud, sin embargo, en algunos métodos, tener la mejor aptitud no garantiza que el individuo se seleccionado, puesto que dependen de una selección aleatoria, baste como muestra el método de selección por rueda de la ruleta.

- *Cruza*: los individuos padres mezclan entre sí su configuración para dar origen a nuevos individuos hasta repoblar la siguiente generación; el hijo contendrá genes de todos los padres, tal como se muestra en la Figura 25 donde se observa una cruza a partir de un punto de cruza.
- *Mutación*: consiste en alterar genes de los individuos de la nueva generación basándose en probabilidades para decidir qué individuos y que genes de estos individuos deben cambiar (mutar) su valor. Este operador se aplica con la intención de expandir el espacio de soluciones y evitar que la búsqueda se concentre en un máximo local. En la Figura 25, los genes $amarillo$ y $azul$ son valores que no pertenecen a los genes de los padres, por lo tanto, son mutaciones.

El proceso se repite hasta evolucionar las generaciones marcadas por el usuario, el individuo mejor calificado en todas las generaciones representa la solución óptima al problema [59], [60].

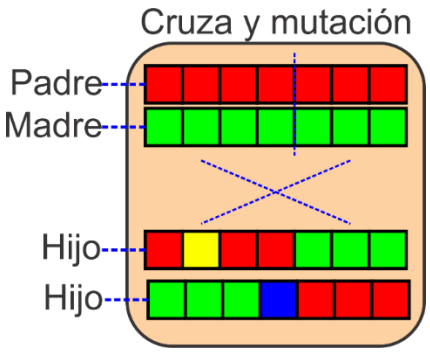


Figura 25. Cruza y mutación

CAPITULO IV. METODOLOGÍA

4.1 Herramientas

4.1.1 R Project

Para el desarrollo de la investigación se utilizó el software *R Project* [61]. R Project es un lenguaje y ambiente para computación gráfica y estadística similar al lenguaje S, fue desarrollado en Bell Laboratories por John Chambers y colaboradores; R provee una amplia gama de técnicas estadísticas (modelos lineales y no lineales, pruebas clásicas de estadística análisis de series de tiempo, clasificación, clusterización, etc.,) y gráficas [61], [62].

R Studio es una interfaz gráfica que proporciona diferentes herramientas (interfaz de línea de comandos, editor de texto y más) para interactuar de manera más sencilla con *R Project* [63]. La versión utilizada de *R Studio* es: 1.0.153.

4.1.2 Recursos computacionales

El computador utilizado durante los experimentos fue un ordenador portátil Samsung Ultrabook serie 5, S.O. *Windows* 8 *Enterprise 64* bits, procesador *Intel(R) Core(TM)*i3-3217U CPU @ 1.80GHz y 12GB de memoria *RAM*.

4.2 Configuración de Arquitectura de una RNA mediante Algoritmos Genéticos

Como se mencionó, la manera de estructurarse y conectarse entre las neuronas, y el algoritmo de aprendizaje utilizado por la RNA se conoce como arquitectura de la RNA.

4.2.1 Parámetros por configurar de la arquitectura de la RNA y otros parámetros

En esta sección se mencionan los parámetros que forman parte del espacio de búsqueda del algoritmo genético, en otras palabras, se mencionan las variables que definen el universo de soluciones en cada una de la dimensión aportada por cada variable. Cada variable es plasmada en un único gen del cromosoma de forma abstracta.

Los parámetros propios de la arquitectura de la RNA a considerar dentro del algoritmo genético son:

- Número de capas ocultas.
- Número de neuronas en cada capa oculta.
- Número de épocas (iteraciones).
- Tasa de aprendizaje.

Parámetros que no forman parte de la arquitectura de la RNA, pero son considerados en el algoritmo genético:

- Proporción (razón) de muestreo del dataset.
  Para conformar los subconjuntos de entrenamiento y prueba.

Los parámetros propios de la arquitectura de la RNA que no están considerados dentro del algoritmo genético son:

- Algoritmo de aprendizaje.
  Todas las arquitecturas utilizan el algoritmo de aprendizaje propagación hacia atrás, ver la sección 3.2.3.1.
- Función de activación.
  Todas las neuronas en todas las arquitecturas utilizan la función de activación logística expresada como:

$$\frac{1}{1 + e^{-((\sum_{j=1}^{n} i_j p_j) + p_0)}} \tag{11}$$

Donde:

- n: es el número de elementos contenidos en $\bar{\imath}$;
- i_j: es el valor de entrada j de $\bar{\imath}$;
- p_j: es el peso correspondiente al valor de entrada j;
- p_0: es el peso correspondiente al *bias*;

4.2.2 Codificación del cromosoma y limitación del espacio de búsqueda

Considerando que el universo de soluciones (espacio de búsqueda) tiende a ser infinito, es necesario limitar cada una de las dimensiones (variables) que integran el mismo. La Tabla 5 muestra los límites para cada parámetro, por ejemplo, el número de neuronas en la primera, segunda y tercera capa oculta (gen 3, 4 y 5), debe ser entre 5 y 50 neuronas, y el valor debe ser de tipo entero, caso similar al número de capas ocultas (gen 2) y numero de épocas (gen 6) donde los límites son de 1 a 3 y de 10 a 50, respectivamente. En el caso de la razón (porcentaje) de muestreo a utilizar (gen 1), los límites son desde 1 a 3, y el valor debe ser entero, donde cada valor representa dos porcentajes, el primero es el porcentaje de registros del *dataset* que conformarán el subconjunto de entrenamiento, el segundo porcentaje es de registros (el resto de los mismo) conformarán el subconjunto de prueba. Con respecto a la tasa de aprendizaje (gen 7) el valor debe ser de tipo flotante y estar dentro de los límites $[0.2 - 0.5]$.

En relación con la codificación del cromosoma (individuo), la Figura 26 muestra un ejemplo de un individuo codificado con valores tomados del alfabeto $Alfabeto \rightarrow [0 - 1]$ de tipo flotante.

Núm. de gen	Parámetro que determina el gen y sus límites	Tipo de dato
1	Razón de muestreo a utilizar [1-3] (1=60%-40%; 2=70%-30%; 3=80%-20%)	Entero (nominal)
2	Número de capas ocultas [1-3]	Entero
3	Número de neuronas en la primera capa oculta [5-50]	Entero
4	Número de neuronas en la segunda capa oculta [5-50]	Entero
5	Número de neuronas en la tercera capa oculta [5-50]	Entero
6	Número de épocas [10-50]	Entero
7	Tasa de aprendizaje [0.2-0.5]	Flotante

Tabla 5. Parámetros determinados por cada gen del cromosoma y su tipo de dato.

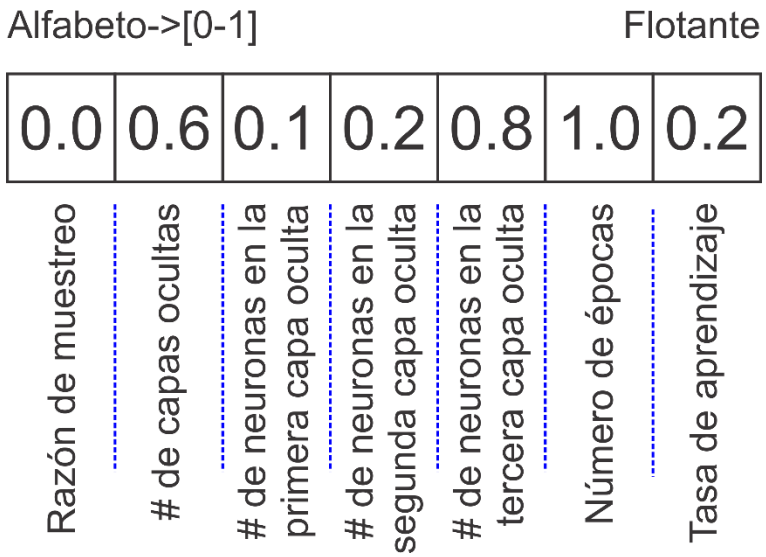


Figura 26. Ejemplo de un cromosoma (individuo) codificado.

4.2.3 Decodificación del cromosoma

Como se mencionó anteriormente (sección 3.3), el alfabeto rige por igual a cada gen de cada cromosoma, dicho de otra manera, cada gen del individuo puede tomar valores flotantes de 0 a 1, sin excepción alguna, sin embargo, el valor $\#\ de\ capas\ ocultas = 0.9$

del gen 2 del cromosoma ejemplo de la Figura 26 no es el valor deseado que se especifica en la Tabla 5, donde se menciona necesitar un valor entero dentro del rango $[1-3]$, por lo tanto es necesario realizar una decodificación del valor de dicho gen y de cada uno de ellos.

Núm. de gen	Ecuación utilizada para la decodificación	Decodificación del cromosoma de la Figura 26
1	(13)	$valGenDecod = integer(0.0 * ((3+1)-1)+1 = 1$
2	(13)	$valGenDecod = integer(0.6 * ((3+0.9)-1)+1 = 2$
3	(13)	$valGenDecod = integer(0.1 * ((50+0.9)-5) + 5 = 9$
4	(13)	$valGenDecod = integer(0.2 * ((50+0.9)-5) + 5 = 14$
5	(13)	$valGenDecod = integer(0.8 * ((50+0.9)-5) + 5 = 41$
6	(13)	$valGenDecod = integer(1.0 * ((50+0.9)-10) + 10 = 50$
7	(12)	$valGenDecod = 0.7 * (0.5-0.2)+0.2 = 0.26$

Tabla 6. Proceso de decodificación, con ejemplo.

$$valGenDecod = valGenCod * (LS - LI) + LI \quad (12)$$

Donde:

- $valGenDecod$: es el valor decodificado del gen;
- $valGenCod$: es el valor codificado del gen;
- LS: es el límite superior del rango deseado;
- LI: es el límite inferior del rango deseado;

$$valGenDecod = integer(valGenCod * ((LS + 0.9) - LI) + LI \qquad (13)$$

Donde:

- $valGenDecod$: es el valor decodificad del gen;
- $valGenCod$: es el valor del codificado del gen;
- $integer$: indica que sola la parte entera será tomada;
- LS: es el límite superior del rango deseado;
- LI: es el límite inferior del rango deseado;

En la Tabla 6 se especifica la ecuación utilizada para decodificar cada gen del cromosoma y además se muestra como ejemplo la decodificación de los genes del cromosoma mostrado en la Figura 26, como resultado se muestra en la Figura 27 el mismo cromosoma decodificado, el cual puede ser interpretado (leído) como:

> RNA con dos capas ocultas, la primera capa oculta contiene nueve neuronas y la segunda capa contiene 14 neuronas. La RNA aprende durante cincuenta épocas con una tasa de aprendizaje de 0.4. La RNA utiliza el 60% de los registros del *dataset* para entrenar y el resto para ser evaluada. Misma interpretación se muestra de manera gráfica en la Figura 28.

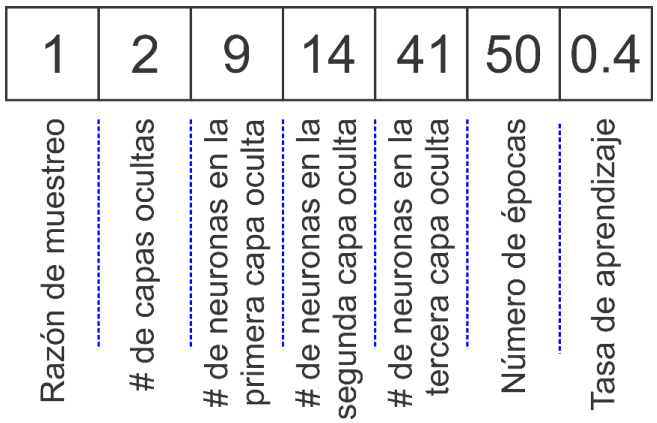


Figura 27. Ejemplo de un cromosoma (individuo) decodificado.

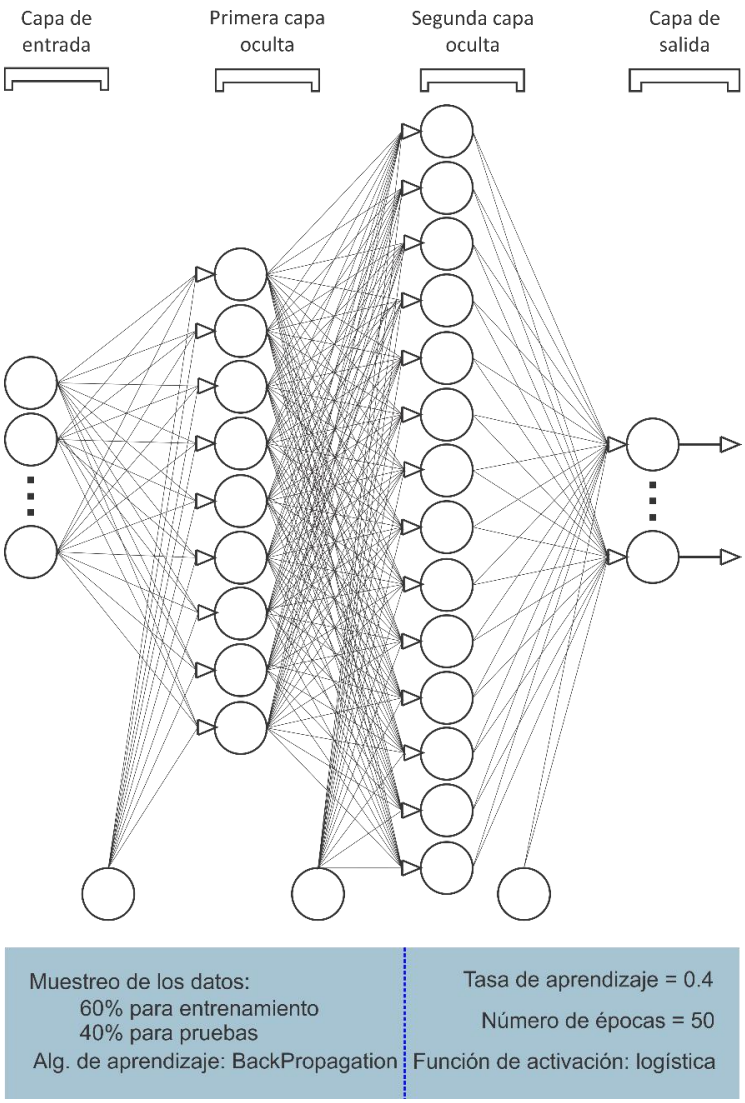


Figura 28. RNA con dos capas ocultas.

4.2.4 Ciclo de configuración de la arquitectura

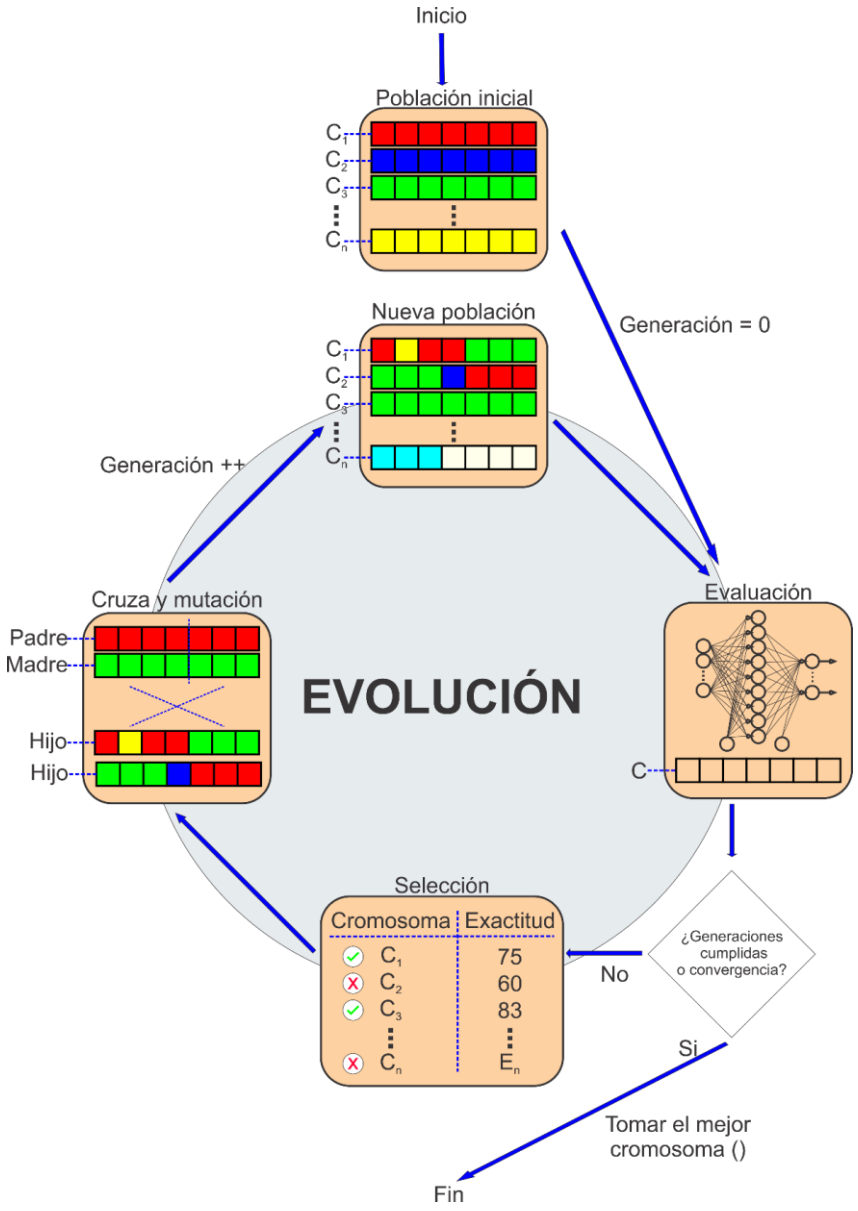


Figura 29. Ciclo de evolución del AG para determinar la arquitectura óptima.

La muestra el proceso de evolución del AG para determinar la arquitectura óptima de la RNA, mismo que se detalla en los siguientes pasos:

1. Población inicial: para empezar, se genera una población inicial compuesta por n cromosomas (individuos) $C = \{C_1, C_2, C_3, \dots, C_n\}, C \in P_0$ con 7 genes cada uno, los individuos son similares al que se muestra en la Figura 26 y los valores de los genes

se generan de manera aleatoria dentro de los límites mencionados anteriormente en la sección 4.2.2 Codificación del cromosoma y limitación del espacio de búsqueda. La población inicial se conoce como la generación cero $P_0 = G_0 \ \therefore C \in G_0$.

2. Evaluación: cada uno de los cromosomas de la población (generación) en curso $\forall C \in G_k$ es evaluado bajo los siguientes pasos:
    a. Decodificación del cromosoma.
    b. Construcción de la arquitectura de la RNA: a partir de la decodificación del cromosoma se construya la arquitectura de la RNA, por ejemplo, decodificando el cromosoma mostrado en la Figura 26, se obtiene el cromosoma mostrado en la Figura 27, con el cual se construye la arquitectura mostrada en la Figura 28.
    c. Proceso de aprendizaje de la RNA: el proceso de aprendizaje de la RNA se realiza mediante el algoritmo de aprendizaje propagación hacia atrás BackPropagation, utilizando validación cruzada con 10-carpetas.
    d. Prueba de la RNA: mediante el cálculo de la matriz de confusión se calcula la exactitud, valor que se asigna al cromosoma como calificación o aptitud.

    Al final obtenemos $E = \{E_1, E_2, E_3, \dots, E_n\}, \ \forall C \ \exists \ E$.
3. Condición de parada: se consideran dos condiciones de parada en el proceso de evolución:
    a. Cuando se cumplen las k generaciones establecidas por el usuario.
    b. Cuando los cromosomas C de la población k convergen. Se dice que la población converge cuando al menos el 95% de los cromosomas C son iguales.

    Si se cumple la condición de parada, se toma el mejor individuo de la población actual y finaliza el proceso, de lo contrario el proceso continúa con el paso 4.
4. Selección: la selección de los individuos para la cruza se realiza por el método rueda de la ruleta. Además, los mejores cromosomas (el 20%) son conservados en la

siguiente generación G_{k+1}, método conocido como elitismo. Un mismo cromosoma puede salir seleccionado en ambos métodos, es decir, participar en la cruza y sobrevivir en la siguiente generación G_{k+1}.

5. Cruza y mutación: la población de la generación G_{k+1} se compone en su 20% por los mejores cromosomas de la población de la generación actual G_k, el resto de la población se forma a partir de la cruza de los cromosomas seleccionados por el método de la rueda de la ruleta. La cruza se realiza basada en un punto de cruza. Una vez concebido el cromosoma hijo, se realiza la mutación de los genes de acuerdo con la probabilidad de mutación. La Figura 25 y muestran gráficamente la realización de la cruza y mutación.
6. Nueva población: una vez realizada la cruza y mutación, el número de generación incrementa $k = k + 1$, ahora la nueva población se conoce como $P_k = G_k \ \therefore C \in G_k$. El proceso continúa con el paso 2.

4.3 Preprocesamiento de datos

En relación con el balanceo de las clases, se omitirán las clases con menos registros (cuando sea necesario), los datos atípicos son corregidos utilizando la mediana de los valores del mismo atributo pertenecientes a la misma clase de cada valor atípico [48]. Se eligió la mediana y no la media debido a que la media está influenciada por datos atípicos y la mediana no. Para detectar los datos atípicos se utiliza el diagrama de cajas y bigotes (*boxplot*) [64], [65].

Con respecto a los valores faltantes en los datos, estos son reconstruidos utilizando el método “el vecino más cercano”, considerando evaluar solo los vecinos pertenecientes a la misma clase del valor faltante [21].

La normalización de los datos se realiza aplicando la ecuación (14) a los datos de cada atributo (columna) del *dataset*, obteniendo así una transformación de los datos al intervalo

[0,1]. Para la selección de atributos, en cada *dataset* se consideraron subconjuntos de atributos mencionados en la literatura.

$$\hat{x}_{ij} = \frac{x_{ij} - Min(\bar{x}_j)}{Max(\bar{x}_j) - Min(\bar{x}_j)} \quad (14)$$

Donde*:*

- n: número de registros (instancias) en el *dataset*;
- m: número de atributos (variables) en el *dataset*;
- $i = \{1, 2, \dots, n\}$;
- $j = \{1, 2, \dots, m\}$;
- $\hat{x}_{ij}$: valor transformado i del atributo $\bar{x}_j$;
- x_{ij}: valor i del atributo $\bar{x}_j$. Valor por transformar;
- $Min(\bar{x}_j)$: valor mínimo del atributo $\bar{x}_j$;
- $Max(\bar{x}_j)$: valor máximo del atributo $\bar{x}_j$

CAPITULO V. EXPERIMENTOS Y ANÁLISIS DE RESULTADOS

5.1 *Dataset*: arritmia cardíaca

5.1.1 Preparación del *dataset*

Dado que la mayoría de las instancias corresponden a la clase 1 existen clases con muy pocas instancias e inclusive sin instancias (ver la), se eliminaron todas las instancias de las clases cuyos números de instancias era menor a 10 [15], eliminando 23 registros, en consecuencia el *dataset* se conforma por 429 registros distribuidos sobre 8 clases $C = \{1, 2, 3, 4, 5, 6, 10\ y\ 16\}$ como se muestra en la Figura 30, sin embargo, el rango de las clases sigue siendo amplio [1,16]. Dado que los clasificadores funcionan mejor cuando el rango de clasificación es menor, el rango se redujo $[0,7]$, reasignando un nuevo código a cada clase prevaleciente, tal como se muestra en la Tabla 7, así que, la información de la Tabla 3 y la se actualiza a la información mostrada en la Tabla 7 y la Figura 31, respectivamente.

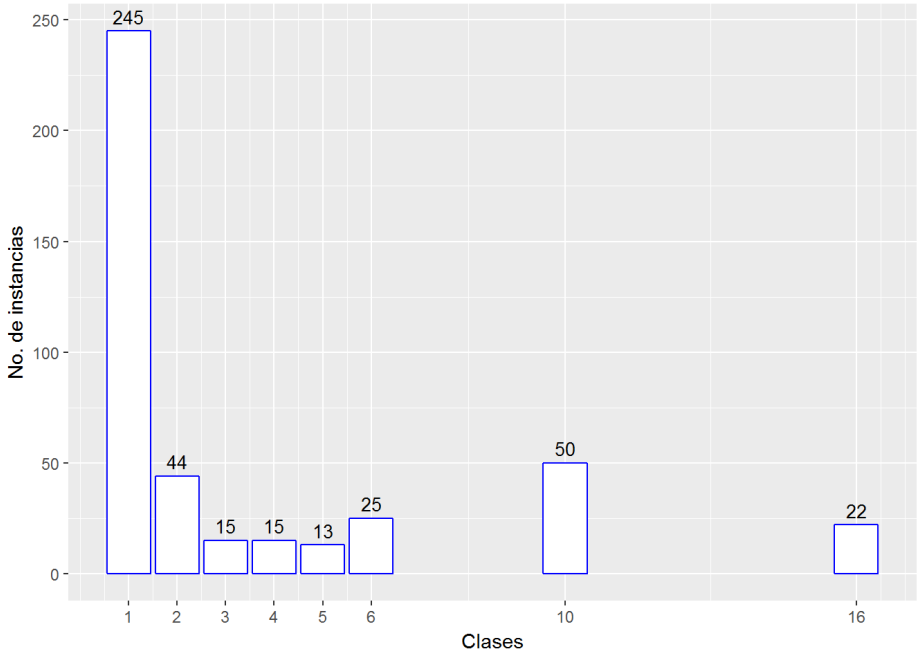


Figura 30. Distribución de las instancias sobre las clases que contienen más de 10 instancias. Dataset: arritmia cardíaca.

(Código anterior de la clase)[3]	(Código nuevo de la clase)[4]	Clase	Número de instancias
1	0	Ausencia de arritmia	245
2	1	Cardiopatía isquémica (enfermedad coronaria o isquemia cardíaca)	44
3	2	Infarto de miocardio anterior	15
4	3	Infarto de miocardio inferior	15
5	4	Taquicardia sinusal	13
6	5	Bradicardia sinusal	25
10	6	Bloqueo de la rama derecha	50
16	7	Otras	22

Tabla 7. Clases prevalecientes después de eliminar las clases cuyo número de instancia era menor a 10 instancias. Dataset: arritmia cardíaca.

279 atributos son muchos para tan solo 452 registros [66], por lo tanto el subconjunto de atributos utilizado en los experimentos es el que utilizan en [20], consta de 24 atributos[5] cuyas posiciones en el *dataset* original son: A={5, 7, 8, 11, 15, 40, 76, 90, 93, 100, 103, 112, 114, 190, 197, 211, 217, 224, 228, 247, 248, 267, 277, 279}.

La Figura 32 muestra en qué atributo se concentran los datos faltantes, destacando el atributo *J al* contener el 85.55% de sus datos en estado de faltantes. El atributo *T* pertenece al subconjunto de atributos A mencionado anteriormente, puesto que su posición en el dataset original [8] es la posición 11, en la Figura 32 se observa que dicho atributo contiene 8 de sus 429 valores en calidad de faltantes, mismos que han sido reemplazados utilizando el método *k*-vecinos más cercanos.

[3] Código bajo el cual se identificaba la clase en el dataset.
[4] Código bajo el cual se identifica actualmente la clase en el dataset.
[5] Los nombres de los atributos seleccionados pueden ser consultados en [8].

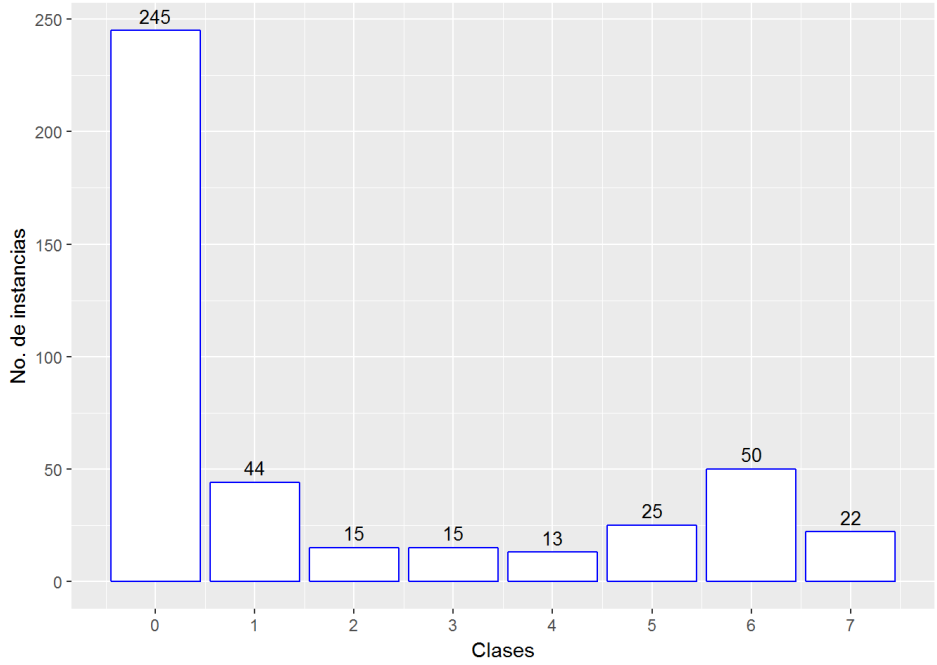


Figura 31. Reasignación de código a las clases que contienen más de 10 instancias. Dataset: arritmia cardíaca.

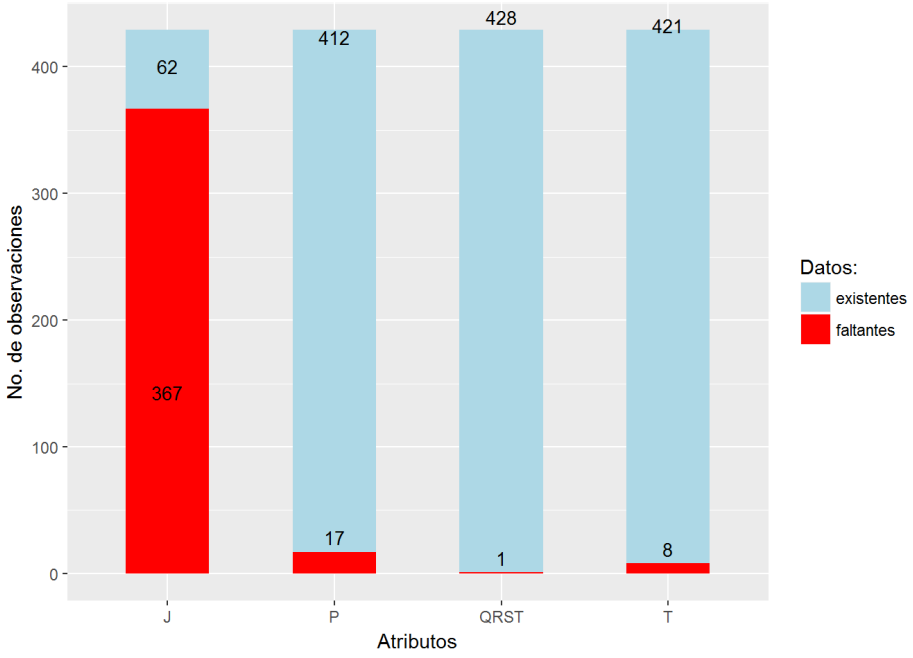


Figura 32. Datos existentes y faltantes por atributo[7]. Dataset: arritmia cardíaca.

[7] Solo se muestran los atributos que contienen datos faltantes (perdidos).

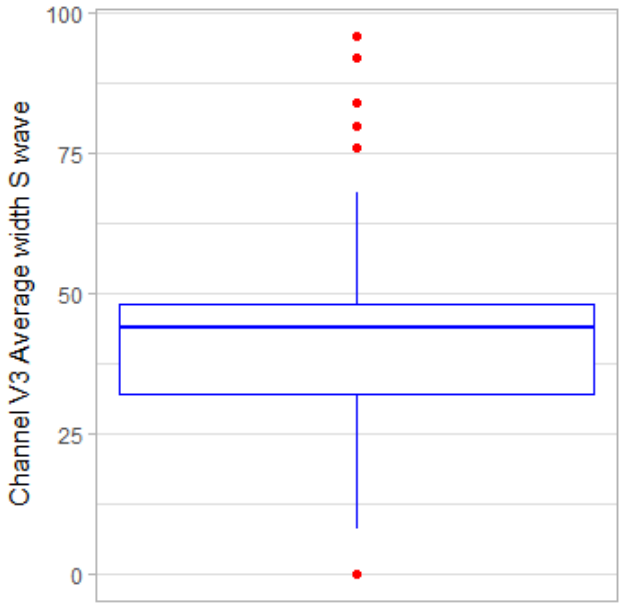


Figura 33. Gráfico de caja y bigotes del atributo 114 del dataset: arritmia cardíaca.

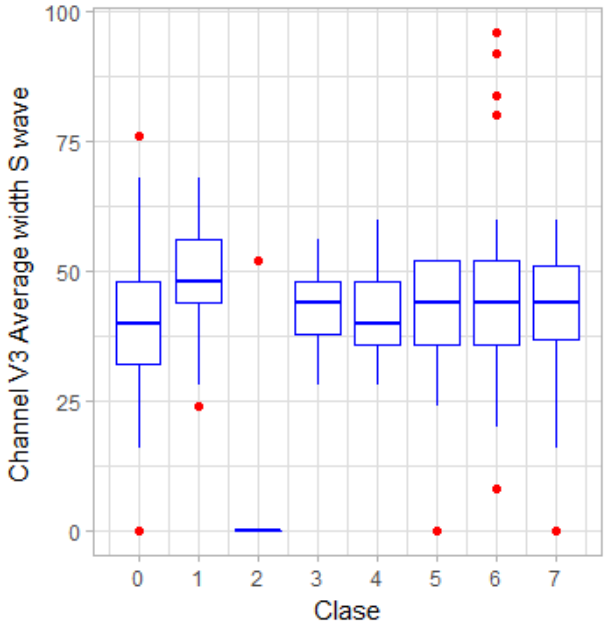


Figura 34. Gráfico de caja y bigotes por clase del atributo 114 del dataset: arritmia cardíaca.

Los valores atípicos fueron tratados utilizando la mediana. La Figura 33 muestra los datos atípicos del atributo 114 del dataset en uso [8], mientras que la Figura 34 muestra los datos

atípicos del mismo atributo por cada clase. En cada una de las figuras, los datos atípicos se muestran en color rojo. La normalización de los datos se realizó utilizando la ecuación (14).

5.1.2 Experimentos y resultados

Durante veinte generaciones se evaluaron 2,835 cromosomas (propuestas de solución o arquitecturas de RNA), utilizando el ciclo de evolución que se muestra en la , bajo los parámetros mencionados en la Tabla 8, obteniendo así, los resultados mostrados en La Figura 35.

Población	135
Generaciones	20
Probabilidad de mutación del cromosoma	1%
Método de selección	Por rueda de la ruleta
Elitismo	20%
Tiempo de cómputo	17:54:08 hh:mm:ss

Tabla 8. Parámetros de algoritmo genético. Experimento: arritmia.

Es en la generación 17 cuando se obtiene el cromosoma que define la arquitectura de la RNA con mejor exactitud $E = 81.21\%$ en la clasificación, dicho cromosoma se muestra decodificado en la Figura 36 conforme a lo indicado en la sección 4.2.3 Decodificación del cromosoma, y representa la siguiente arquitectura:

> RNA con tres capas ocultas que contiene veinticuatro entradas (accesos para los datos) en la capa de entrada, diez neuronas ocultas tanto en la primera como en la segunda capa oculta, mientras que en la tercera capa oculta contiene treinta y tres neuronas ocultas y en la capa de salida contiene ocho neuronas. Dicha arquitectura es de tipo multi-capa alimentada hacia adelante (*multi-layer perceptron FeedForward*) totalmente conexa, entrenada mediante el algoritmo de aprendizaje

BackPropagation utilizando validación cruzada con 10-carpetas con una tasa de aprendizaje de 0.24, cada neurona utiliza función de activación logística. Utiliza el setenta por ciento de los datos del *dataset* para aprender y el resto para ser evaluada.

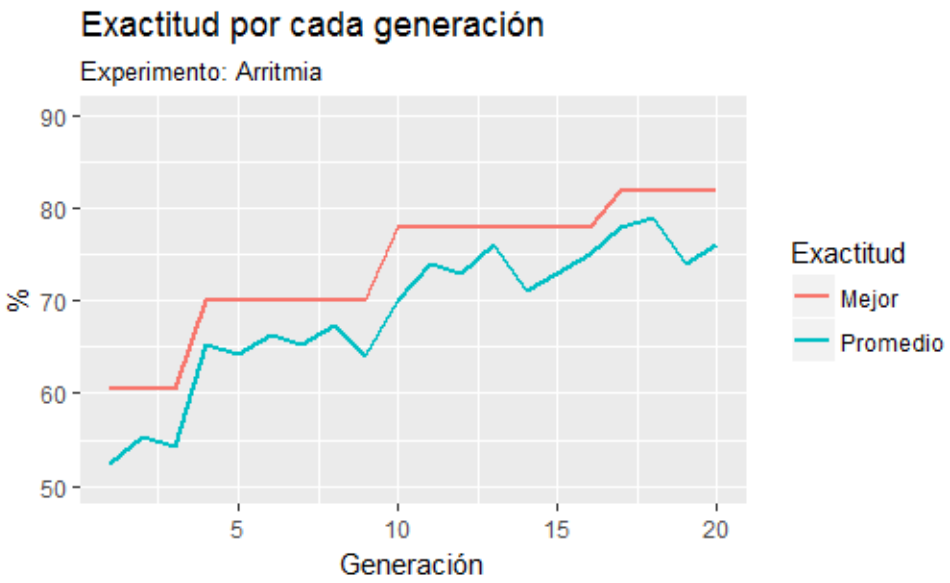


Figura 35. Evolución del algoritmo genético. Experimento: arritmia.

La variabilidad presentada por la gráfica de la exactitud promedio en cada generación, indica que, durante la evolución, el AG evalúa cromosomas de diferentes partes del espacio de soluciones y evita el problema del máximo local. El experimento tarde diecisiete horas con cincuenta y cuatro minutos y ocho segundos en realizar el proceso evolutivo.

Es importante mencionar los siguientes datos: el *dataset* preprocesado contiene 429 registros y el mejor cromosoma utiliza la tasa de muestreo $70\% - 30\%$, que indica que el setenta por ciento de los registros se utilizan para entrenar la RNA y el treinta por ciento restantes se utilizan para evaluar a la misma, por lo tanto, de los 429 registros, 297 integran el subconjunto de entrenamiento y 132 el subconjunto de prueba (evaluación). La tasa de muestreo se aplica a cada clase tal como se muestra en la Tabla 9.

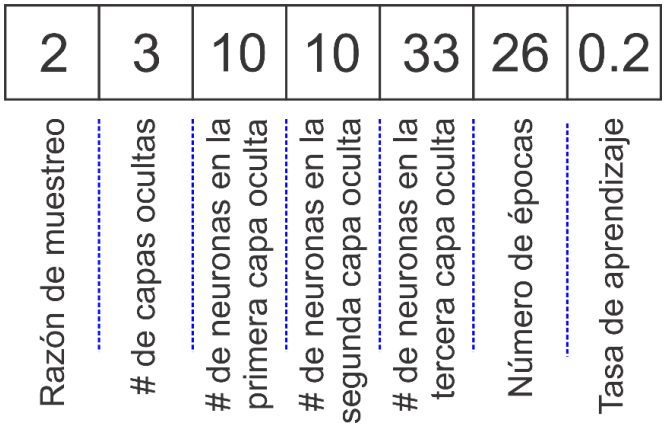


Figura 36. Mejor cromosoma decodificado. Experimento: arritmia.

	Subconjunto de entrenamiento	Subconjunto de prueba	**Número total de registros por clase:**
Clase: 0	171	74	**245**
Clase: 1	30	14	**44**
Clase: 2	10	5	**15**
Clase: 3	10	5	**15**
Clase: 4	9	4	**13**
Clase: 5	17	8	**25**
Clase: 6	35	15	**50**
Clase: 7	15	7	**22**
Número total de registros por subconjunto:	**297**	**132**	**429**

Tabla 9. Razón de muestreo 70% subconjunto de entrenamiento – 30% subconjunto de prueba, aplicada por clase al dataset arritmia preprocesado.

La Tabla 10 y la Tabla 11 muestran la matriz de confusión y tres métricas de evaluación, respectivamente, ambas tablas corresponden a la evaluación de la arquitectura óptima mencionada anteriormente con el treinta por ciento de los registros del *dataset* preprocesado (ver la Tabla 9). A partir de dichas tablas podemos concluir los siguiente:

- La RNA óptima clasifica con una exactitud $E = 81.21\%$.

- De las clases 2 y 3 todos sus registros se clasifican correctamente, en otras palabras, estas dos clases están perfectamente separadas de las demás y entre ellas mismas, situación que confirma el valor de $Sensibilidad = 100\%$ en ambas clases.
- En contraste con el punto anterior, todos los registros de la clase 7 son clasificados erróneamente y tiende a confundirse fuertemente con la clase 0 y débilmente con las clases 2 y 6.
- La clase 0 posee alta $Sensibilidad = 98.64\%$, al clasificar erróneamente solo un registro de los 74 existentes en esta misma clase, sin embargo, como la clase 0 contiene el mayor número de registros (ver la Figura 31), la RNA muestra un sesgo hacia esta clase, razón por la cual la mayoría de las clasificaciones erróneas recaen hacia esta clase. El porcentaje de $Especificidad = 72.41$ confirma el sesgo, puesto que la clase 0 reporta la especificidad más baja y muy distante del resto de las clases.

		Referencia								
Clasificación		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	
	0	**73**	4	0	0	2	2	3	5	**89**
	1	0	**6**	0	0	1	0	0	0	**7**
	2	0	1	**5**	0	0	0	0	1	**7**
	3	0	3	0	**5**	0	0	0	0	**8**
	4	0	0	0	0	**1**	0	0	0	**1**
	5	0	0	0	0	0	**6**	0	0	**6**
	6	1	0	0	0	0	0	**12**	1	**14**
	7	0	0	0	0	0	0	0	**0**	**0**
		74	**14**	**5**	**5**	**4**	**8**	**15**	**7**	**132**

Tabla 10. Matriz de confusión que evalúa la RNA cuya arquitectura está determinada por el mejor cromosoma. Experimento: arritmia.

- Aunado al sesgo mostrado por la RNA hacia la clase 0 como se explica en el punto anterior, la clase 4 contiene el menor número de registros (ver la Figura 31) causando una $Sensibilidad = 25\%$ muy baja, de modo que al clasificar

erróneamente uno solo de sus registros reduce drásticamente su sensibilidad, lo cual se interpreta como, poca probabilidad de clasificar correctamente. Esta clase en particular es la que más reciente el efecto del sesgo hacia las clases más con mayor número de registros.

	Exactitud balanceada %	Sensibilidad %	Especificidad %
Clase: 0	85.53	98.64	72.41
Clase: 1	71	42.85	99.15
Clase: 2	99.21	100	98.42
Clase: 3	98.81	100	97.63
Clase: 4	62.5	25	100
Clase: 5	87.5	75	100
Clase: 6	89.14	80	98.29
Clase: 7	50	0	100

Tabla 11. Exactitud balanceada, sensibilidad y especificidad por clase. Dataset: arritmia.

5.2 *Dataset*: problemas del corazón

5.2.1 Preparación del *dataset*

La Figura 4 evidencian que existe un desbalanceo en la distribución de las instancias sobre las clases consideradas en el dataset. La clase 4 contiene el menor número de registros, por consiguiente, esta clase ha sido omitida. Después de eliminar la clase 4 con 13 instancias, el dataset solo contiene un total de 290 registros distribuidos sobre 4 clases $C = \{0, 1, 2, 3\}$ como se muestra en la Figura 30.

En los experimentos del presente proyecto se han utilizado solo los 5 atributos[8] A={3, 10, 12, 32, 58} (incluido el atributo que especifica la clase de enfermedad), tal como lo sugieren

[8] Los nombres de los atributos seleccionados pueden ser consultados en [12].

[31]. Hay que hacer notar que existen todos los datos de dichos atributos, fueron normalizados utilizando la ecuación (14) y los valores atípicos fueron tratados utilizando la mediana, tal como se mencionó anteriormente.

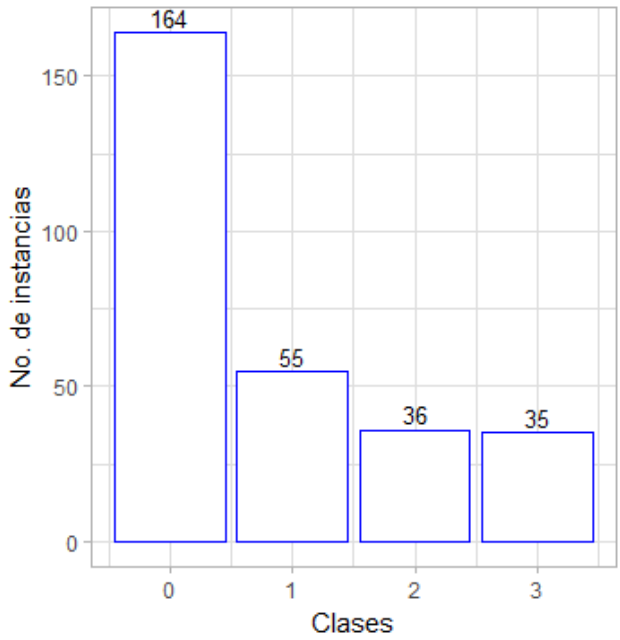


Figura 37. Distribución de las instancias sobre las clases omitiendo la clase 4. Dataset: problemas del corazón.

La Figura 38 muestra en color rojo los datos atípicos por cada clase del atributo 10 del dataset en uso [12].

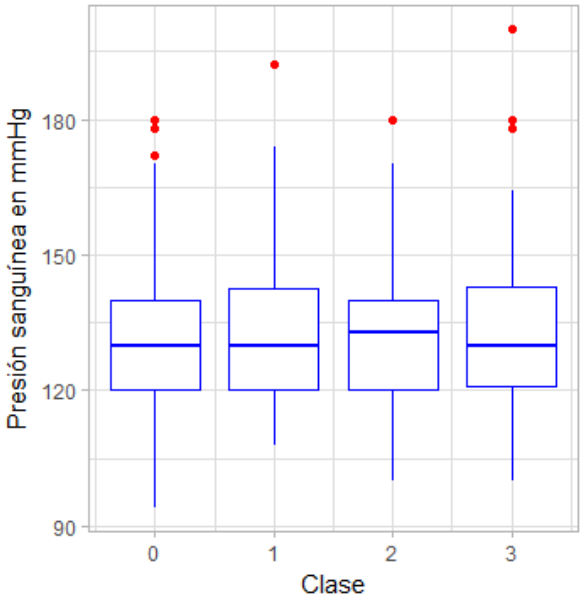


Figura 38. Gráfico de caja y bigotes por clase del atributo 10 presión sanguínea durante descanso (en mmHg al ingresar al hospital) del dataset: problemas del corazón.

5.2.2 Experimentos y resultados

En la Tabla 12 se mencionan los parámetros utilizados para realizar el ciclo de evolución mostrado en la durante veinte generaciones, evaluando así, un total de $2{,}835$ cromosomas, obteniendo el mejor cromosoma con una exactitud de clasificación $E = 72.82\%$ durante la treceava generación (ver Figura 39). El cromosoma se muestran decodificado en la Figura 40 y su arquitectura se describe a continuación:

> RNA con dos capas ocultas que contiene cuatro entradas (accesos para los datos) en la capa de entrada, cinco neuronas ocultas en la primera y veinticinco neuronas ocultas en la segunda capa oculta, y en la capa de salida contiene cuatro neuronas. Dicha arquitectura es de tipo multi-capa alimentada hacia adelante (*multi-layer perceptron Feedforward*) totalmente conexa, entrenada mediante el algoritmo de aprendizaje *BackPropagation* utilizando validación cruzada con 10-carpetas con una tasa de aprendizaje de 0.26, cada neurona utiliza función de activación logística.

Utiliza el ochenta por ciento de los datos del *dataset* para aprender y el resto para ser evaluada.

El experimento tardó catorce horas con cincuenta y seis minutos en realizar el proceso evolutivo.

Población	135
Generaciones	20
Probabilidad de mutación del cromosoma	1%
Método de selección	Por rueda de la ruleta
Elitismo	20%
Tiempo de cómputo	14:56:00 hh:mm:ss

Tabla 12. Parámetros de algoritmo genético. Experimento: problemas del corazón.

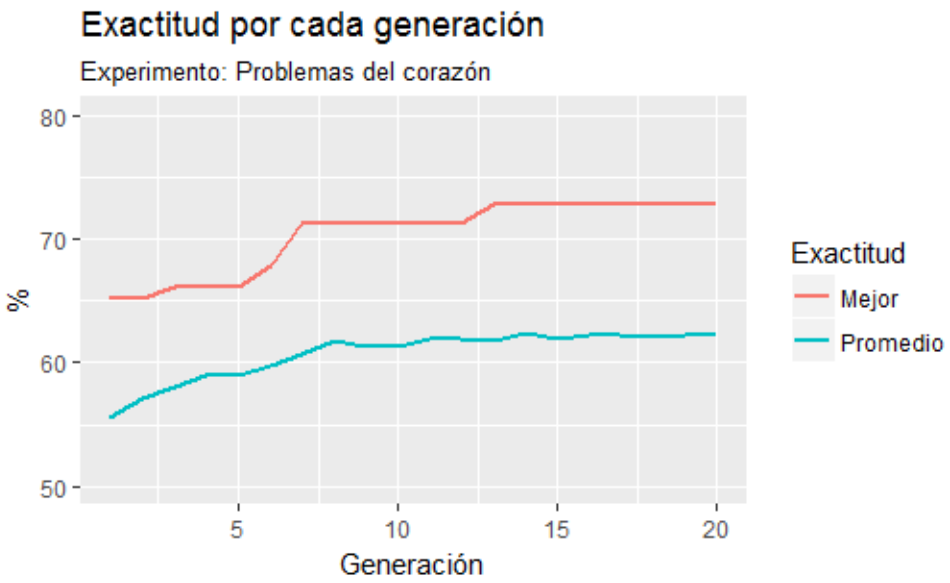


Figura 39. Evolución del algoritmo genético. Experimento: problemas del corazón.

El *dataset* preprocesado contiene 290 registros y existen dos cromosomas óptimos, ambos utilizan la tasa de muestreo $80\% - 20\%$, que indica que el ochenta por ciento de los registros se utilizan para entrenar la RNA y el veinte por ciento restantes es usado para

evaluar a la misma, por lo tanto, de los 290 registros, 231 integran el subconjunto de entrenamiento y 59 el subconjunto de prueba (evaluación). Es importante mencionar que la tasa de muestreo se aplica a cada clase tal como se muestra en la Tabla 13, por ejemplo, la clase 0 que contiene 164 registros (ver Figura 37), el ochenta por ciento es igual a 131.2, por lo tanto, 131 registros de la clase 0 forman parte del subconjunto de entrenamiento y el resto (33) de los registros pertenecientes a la misma clase forman parte del subconjunto de prueba.

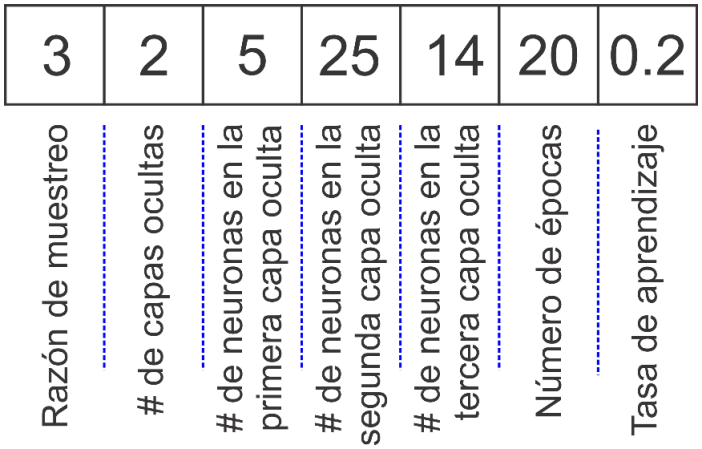


Figura 40. Mejor cromosoma decodificado. Experimento: problemas del corazón.

	Subconjunto de entrenamiento	Subconjunto de prueba	**Número total de registros por clase:**
Clase: 0	131	33	**164**
Clase: 1	44	11	**55**
Clase: 2	28	8	**36**
Clase: 3	28	7	**35**
Número total de registros por subconjunto:	**231**	**59**	**290**

Tabla 13. Razón de muestreo 80% subconjunto de entrenamiento – 20% subconjunto de prueba, aplicada por clase al dataset problemas del corazón preprocesado.

La Tabla 14 muestra la matriz de confusión del cromosoma óptimo (ver Figura 40) mientras que, la Tabla 15 muestra los resultados en tres métricas de evaluación calculadas a partir

de dicha matriz. Estas tablas corresponden a la evaluación del mejor cromosoma con el veinte por ciento de los registros del *dataset* preprocesado (ver la Tabla 13). A partir de dichas tablas podemos concluir los siguiente:

- La RNA óptima clasifica con una exactitud $E = 72.82\%$.

		Referencia			
Clasificación	**0**	**1**	**2**	**3**	
0	**32**	6	0	3	**41**
1	0	**4**	0	1	**5**
2	0	1	**4**	0	**5**
3	1	0	4	**3**	**8**
	33	**11**	**8**	**7**	**59**

Tabla 14. Matriz de confusión que evalúa la RNA cuya arquitectura está determinada por el primer mejor cromosoma. Experimento: problemas del corazón.

	Exactitud balanceada %	Sensibilidad %	Especificidad %
Clase: 0	81.17	96.96	65.38
Clase: 1	67.14	36.36	97.91
Clase: 2	74.01	50	98.03
Clase: 3	66.62	42.85	90.38

Tabla 15. Exactitud balanceada, sensibilidad y especificidad por clase. Dataset: problemas del corazón. Primer mejor cromosoma.

- La clase 0 reporta la mejor sensibilidad, clasificando correctamente el 96.96% de los registros pertenecientes a ella misma, no óbstate, es la clase con el mayor número de registros (ver la Figura 37), situación que genera sesgo en el modelo

hacia esta clase y se confirma con el porcentaje de $Especificidad = 65.38\%$ puesto que la clase 0 reporta la especificidad más baja y muy distante del resto de las clases.

- El menor acierto de clasificación se presenta en la clase 1, la mayoría de las clasificaciones incorrectas recaen en la clase 0, esto debido al sesgo antes mencionado.

5.3 *Dataset*: diabetes

5.3.1 Preparación del *dataset*

Como se describió anteriormente, el *dataset* contiene un total de 768 registros distribuidos sobre 2 clases $C = \{0, 1\}$ como se muestra en la Figura 5. El subconjunto de atributos utilizado en los experimentos consta de 9 atributos[10] (el noveno determinada a qué clase pertenece el registro) cuyas posiciones en el *dataset* original son: A={1, 2, 3, 4, 5, 6, 7, 8, 9}.

La Figura 41 muestra los atributos contenedores de datos faltantes, destacando el atributo *4 al* contener el 29.55% y el atributo *5 al* contener el 48.69% de sus datos en calidad de faltantes. La Figura 42 detalla a qué clase pertenecen los datos faltantes del atributo 5, mismos que han sido reemplazados mediante el método del vecino más cercano tal como se mencionó anteriormente.

Los valores atípicos fueron tratados como se menciona en la sección 4.3. La Figura 43 muestra en color rojo los datos atípicos por cada clase en el atributo 5. La normalización de los datos se realizó de acuerdo con lo detallado en 4.3.

[10] Los nombres de los atributos seleccionados pueden ser consultados en [13].

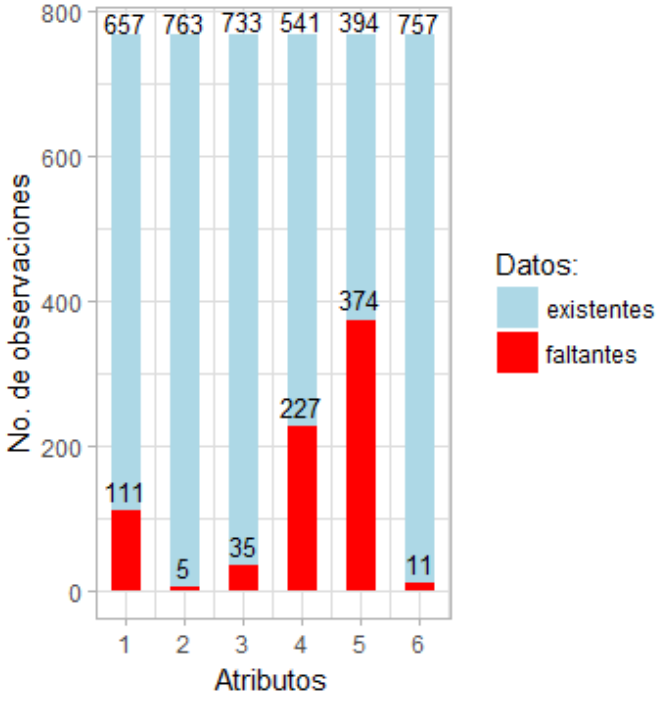


Figura 41. Datos existentes y faltantes por atributo[11]. Dataset: Diabetes.

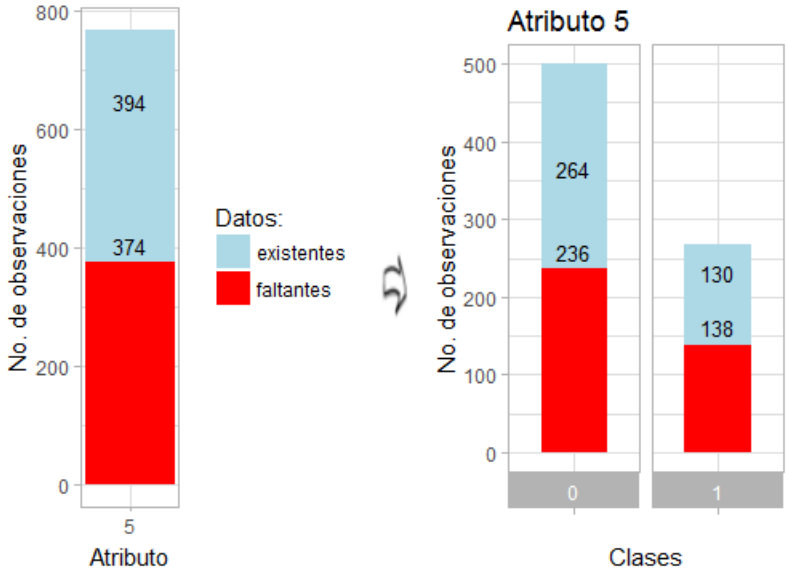


Figura 42. Datos existentes y faltantes por cada clase en el atributo 5. Dataset: diabetes.

[11] Solo se muestran los atributos que contienen datos faltantes (perdidos), sus respectivos nombres pueden ser consultados en la o en [13].

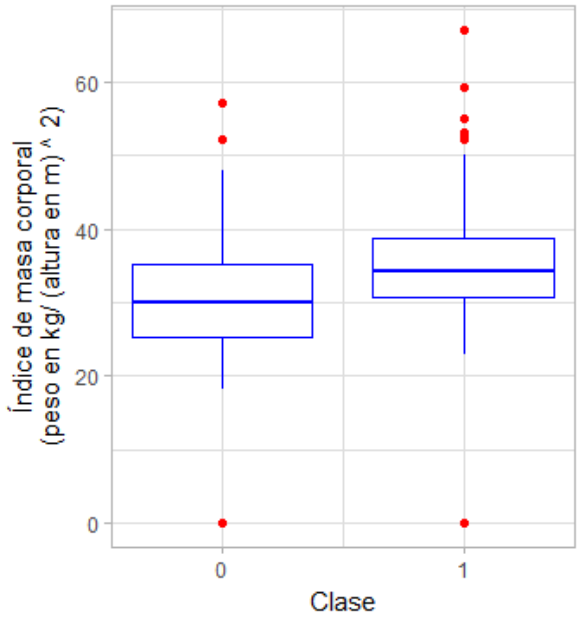


Figura 43. Gráfico de caja y bigotes por clase del atributo 6 índice de masa corporal (peso en $\frac{kg}{altura\ en\ m^2}$ del dataset: Diabetes.

5.3.2 Experimentos y resultados

Realizando el proceso de evolución mostrado en la Figura 29, bajo los parámetros mencionados en la Tabla 16, se evaluaron $2{,}835$ cromosomas (propuestas de solución o arquitecturas de RNA) durante veinte generaciones, conservando el 20% de los mejores individuos en la siguiente generación, seleccionando a los progenitores mediante el método de la rueda de la ruleta y realizando mutación en los cromosomas descendientes bajo una probabilidad de 1%.

La Figura 44 muestra la evolución de los cromosomas en dos aspectos: el mejor cromosoma y el promedio de cada generación. Al inicio del experimento, la exactitud de clasificación de la mejor RNA (cromosoma) es $E \approx 80\%$, mientras que el promedio de exactitud de la población es $PE \approx 76\%$. En la siguiente generación la exactitud aumenta a $E \approx 81\%$. Después en la cuarta generación, surge una RNA con la exactitud más alta $E = 81.17\%$ y se mantiene por seis generaciones más. Al final, en la generación diecisiete aparece la RNA

cuya arquitectura le permite clasificar con una exactitud de $E = 81.21\%$ y se mantiene durante las siguientes dieciséis generaciones. La poca variabilidad presentada por la gráfica de la exactitud promedio en cada generación, indica que, durante la evolución, el AG enfrenta el problema del máximo local y requiere una probabilidad de mutación del cromosoma más alta. El experimento tarde dieciocho horas con cuarenta y tres minutos y cincuenta y tres segundos en realizar el proceso evolutivo.

Población	135
Generaciones	20
Probabilidad de mutación del cromosoma	1%
Método de selección	Por rueda de la ruleta
Elitismo	20%
Tiempo de cómputo	18:43:53 hh:mm:ss

Tabla 16. Parámetros de algoritmo genético. Experimento: diabetes.

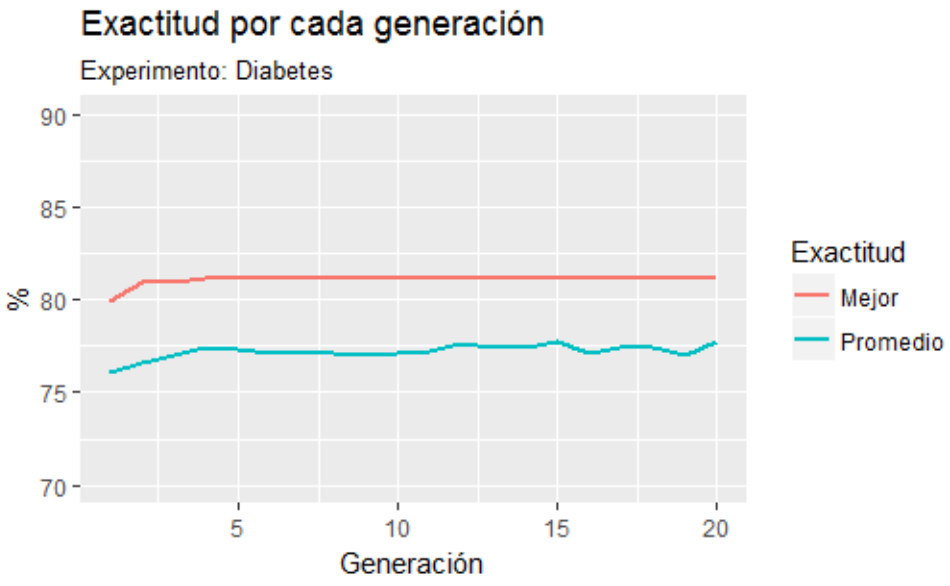


Figura 44. Evolución del algoritmo genético. Experimento: diabetes.

Como se mencionó anteriormente, en la cuarta generación se obtiene el cromosoma óptimo con una exactitud $E = 81.17\%$, dicho cromosoma se muestra en la Figura 45

decodificado conforme a lo indicado en la sección 4.2.3, y representa la siguiente arquitectura:

RNA con dos capas ocultas que contiene ocho entradas (accesos para los datos) en la capa de entrada, veinticuatro neuronas ocultas en la primera ocultan y cuarenta y siete neuronas ocultas en la segunda capa oculta, en la capa de salida contiene dos neuronas. Dicha arquitectura es de tipo multi-capa alimentada hacia adelante (*multi-layer perceptron Feedforward*) totalmente conexa, entrenada mediante el algoritmo de aprendizaje *BackPropagation* utilizando validación cruzada con 10-carpetas con una tasa de aprendizaje de 0.32, cada neurona utiliza función de activación logística. Utiliza el ochenta por ciento de los datos del *dataset* para aprender y el resto para ser evaluada.

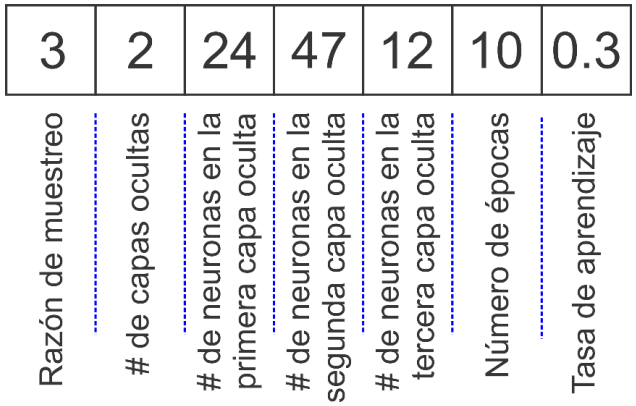


Figura 45. Mejor cromosoma decodificado. Experimento: diabetes.

Retomando datos anteriores, el *dataset* preprocesado contiene 768 registros y el mejor cromosoma utilizan la tasa de muestreo $80\% - 20\%$, que indica que el ochenta por ciento de los registros se utilizan para entrenar la RNA y el veinte por ciento restantes es usado para evaluar a la misma, por lo tanto, de los 768 registros, 614 integran el subconjunto de entrenamiento y 154 el subconjunto de prueba (evaluación). Es importante mencionar que la tasa de muestreo se aplica a cada clase tal como se muestra en la Tabla 17, por ejemplo, la clase 0 que contiene 500 registros (ver la Figura 5), el ochenta por ciento es igual a 614.4, por lo tanto, 614 registros de la clase 0 forman parte del subconjunto de entrenamiento y

el resto (100) de los registros pertenecientes a la misma clase forman parte del subconjunto de prueba.

	Subconjunto de entrenamiento	Subconjunto de prueba	**Número total de registros por clase:**
Clase: 0	400	100	**500**
Clase: 1	214	54	**268**
Número total de registros por subconjunto:	**614**	**154**	**768**

Tabla 17. Razón de muestreo 80% subconjunto de entrenamiento – 20% subconjunto de prueba, aplicada por clase al dataset problemas del corazón preprocesado.

la matriz de confusión se muestra en la Tabla 10, y en la Tabla 11 se muestra los resultados en tres métricas de evaluación calculadas a partir de dicha matriz. Ambas tablas corresponden a la evaluación de la arquitectura óptima mencionada anteriormente con el treinta por ciento de los registros del *dataset* preprocesado (ver la Tabla 9). A partir de dichas tablas podemos concluir los siguiente:

- La RNA óptima clasifica con una exactitud $E = 81.17\%$.

	Referencia		
Clasificación	**0**	**1**	
0	**92**	21	**113**
1	8	**33**	**41**
	100	**54**	**154**

Tabla 18. Matriz de confusión que evalúa la RNA cuya arquitectura está determinada por el mejor cromosoma. Experimento: diabetes.

- La clase 0 tiene $Sensibilidad = 92\%$, mientras que la clase 1 tiene $Sensibilidad = 61.11\%$, estos valores indican que la clase 0 realizo más porcentaje de clasificaciones correctamente que la clase 1.

- Existe la posibilidad de que exista un sesgo en el modelo hacia la clase 0, debido a que esta posee más cantidad de registros que la clase 1, tal como se muestra en la Figura 5.

CAPITULO VI. CONCLUSIONES Y TRABAJO FUTURO

Se implementó un AG capaz de determinar y optimizar diferentes combinaciones de los parámetros de la arquitectura de una RNA para maximizar a través de las generaciones (iteraciones) la exactitud de la RNA al realizar clasificación de distintos tipos de clases en tres dataset comúnmente utilizados en el área médica: arritmia, problemas del corazón y diabetes.

La combinación realizada en esta investigación permite descubrir distintas combinaciones óptimas a la arquitectura de una RNA, las cuales pueden ser punto de partida de otras técnicas con el fin de mejorar los resultados (por ejemplo, la exactitud de clasificación).

Cuando se tiene poca o nula experiencia al clasificar información mediante RNA, la metodología utilizada en el presente proyecto es buena opción.

Los resultados obtenidos $E = 81.21\%$, $E = 72.82\%$ y $E = 81.21\%$ en los experimentos con tres *dataset* comúnmente utilizados en el área biomédica (arritmia, problemas del corazón y diabetes, respectivamente) son aceptables dado que, por un lado, tanto en el caso la exactitud lograda para con el *dataset* de arritmias como en *dataset* diabetes se encuentran por arriba del promedio de exactitud reportado en la literatura , por el otro lado el resultado obtenido para con el *dataset* problemas del corazón no se encuentra muy distante del valor promedio reportado en la literatura.

Dataset: arritmia [8]	*Dataset*: problemas del corazón [9]	*Dataset*: diabetes [10]
73.14%	75.93%	73.83%

Tabla 19. Exactitud promedio reportada en la literatura revisada por cada dataset.

Como trabajo futuro se proponen los siguientes puntos:

- Experimentar en computadoras más dotadas en cuestión de recursos.
- Incluir el método de selección, la función de activación y algoritmo de entrenamiento como opciones modificables por AG.

- Ampliar los espacios de búsqueda en cada una de las dimensiones (parámetro) que conforman el espacio de búsqueda.
- Experimentar con más *dataset* del área médica u otras áreas.
- Paralelizar las tareas tanto del AG como de la RNA a fin y efecto de lograr una exploración de un espacio de búsqueda más amplio en menor tiempo.

REFERENCIAS

[1] "OMS | Las 10 principales causas de defunción", *2017-01-01*, 01-ene-2017. [En línea]. Disponible en: http://www.who.int/mediacentre/factsheets/fs310/es/. [Consultado: 08-sep-2017].
[2] "INEGI | Mortalidad", *Registros administrativos. Vitales. Natalidad. Matrimonios*. [En línea]. Disponible en: http://www.beta.inegi.org.mx/temas/mortalidad/. [Consultado: 31-oct-2017].
[3] "Crea INEGI ranking de causas de muerte", *El Siglo De Durango*, Durango, Mex., p. 6, 31-oct-2017.
[4] N. Bhatla y K. Jyoti, "An Analysis of Heart Disease Prediction using Different Data Mining Techniques", *Int. J. Eng. Res. Technol. IJERT*, vol. 1, núm. 8, pp. 1–4, oct. 2012.
[5] C. S. Dangare y S. S. Apte, "A Data Mining Approach for Prediction of Heart Disease Using Neural Networks", Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2175569, nov. 2012.
[6] N. Kishore y S. Singh, "Cardiac Analysis and Classification of ECG Signal using GA and NN", *Int. J. Comput. Appl.*, vol. 127, núm. 12, pp. 23–27, 2015.
[7] "UCI Machine Learning Repository", 01-nov-2017. [En línea]. Disponible en: http://archive.ics.uci.edu/ml/index.php. [Consultado: 08-may-2017].
[8] "UCI Machine Learning Repository: Arrhythmia Data Set", *UCI Machine Learning Repository*, 30-sep-2017. [En línea]. Disponible en: http://archive.ics.uci.edu/ml/datasets/Arrhythmia. [Consultado: 30-sep-2017].
[9] "UCI Machine Learning Repository: Heart Disease Data Set", 08-nov-2017. [En línea]. Disponible en: http://archive.ics.uci.edu/ml/datasets/heart+Disease. [Consultado: 08-nov-2017].
[10] "UCI Machine Learning Repository: Pima Indians Diabetes Data Set", 08-nov-2017. [En línea]. Disponible en: https://archive.ics.uci.edu/ml/datasets/pima+indians+diabetes. [Consultado: 08-nov-2017].
[11] H. A. Guvenir, B. Acar, y H. Muderrisoglu, "Arrhythmia Data Set". UCI Machine Learning Repository, 01-ene-1998.
[12] R. Detrano, W. Steinbrunn, M. Pfisterer, y R. Detrano, "Cleveland Dataset", 22-jul-1988. [En línea]. Disponible en: http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data. [Consultado: 08-nov-2017].
[13] V. Sigillito, "Pima Indians Diabetes Dataset", 09-may-1990. [En línea]. Disponible en: https://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.data. [Consultado: 08-nov-2017].
[14] K. Polat, S. Sahan, y S. Gunes, "A new method to medical diagnosis: artificial immune recognition system (AIRS) with fuzzy weighted pre-processing and application to ECG arrhythmia", *Expert Syst. Appl.*, vol. 31, núm. 2, pp. 264–269, 2006.
[15] S. Yeniterzi, R. Yeniterzi, A. Kücükural, y U. Sezerman, "Feature selection with genetic algorithms on cardiac arrhythmia database", *2nd Int. Symp. Health Inform. Bioinforma. HIBIT*, 2007.

[16] W. M. Zuo, W. G. Lu, K. Q. Wang, y H. Zhang, "Diagnosis of cardiac arrhythmia using kernel difference weighted KNN classifier", en *2008 Computers in Cardiology*, 2008, pp. 253–256.

[17] O. Kurşun, C. O. Şakar, O. Favorov, N. Aydin, y S. F. Gürgen, "Using covariates for improving the minimum redundancy maximum relevance feature selection method", *Turk. J. Electr. Eng. Comput. Sci.*, vol. 18, núm. 6, pp. 975–989, dic. 2010.

[18] S. M. Jadhav, S. L. Nalbalwar, y A. A. Ghatol, "Modular Neural Network Based Arrhythmia Classification System Using ECG Signal Data", *Int. J. Inf. Technol. Knowl. Manag.*, vol. 4, núm. 1, pp. 205–209, jun. 2011.

[19] M. Mitra y R. K. Samanta, "Cardiac Arrhythmia Classification Using Neural Networks with Selected Features", *Procedia Technol.*, vol. 10, pp. 76–84, ene. 2013.

[20] E. Namsrai, T. Munkhdalai, M. Li, J.-H. Shin, O.-E. Namsrai, y K. H. Ryu, "A Feature Selection-based Ensemble Method for Arrhythmia Classification", *J. Inf. Process. Syst.*, vol. 9, núm. 1, pp. 31–40, mar. 2013.

[21] E. Yılmaz, "An Expert System Based on Fisher Score and LS-SVM for Cardiac Arrhythmia Diagnosis", *Comput. Math. Methods Med.*, vol. 2013, jun. 2013.

[22] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java". [En línea]. Disponible en: https://www.cs.waikato.ac.nz/ml/weka/. [Consultado: 16-mar-2017].

[23] P. Dhakate, K. Rajeswari, y D. Abin, "Analysis of Different Classifiers for Medical Dataset using Various Measures", *Int. J. Comput. Appl.*, vol. 111, pp. 20–24, feb. 2015.

[24] H. Alshraideh, M. Otoom, A. Al-Araida, H. Bawaneh, y J. Bravo, "A Web Based Cardiovascular Disease Detection System", *J. Med. Syst.*, vol. 39, núm. 10, p. 122, oct. 2015.

[25] H. Yan, J. Zheng, Y. Jiang, C. Peng, y Q. Li, "Development of a decision support system for heart disease diagnosis using multilayer perceptron", en *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS '03*, 2003, vol. 5, pp. V-709-V–712 vol.5.

[26] P. Andreeva, "Data modelling and specific rule generation via data mining techniques", presentado en International Conference on Computer Systems and Technologies-CompSysTech, 2006.

[27] K. Polat, S. Şahan, y S. Güneş, "Automatic detection of heart disease using an artificial immune recognition system (AIRS) with fuzzy resource allocation mechanism and k-nn (nearest neighbour) based weighting preprocessing", *Expert Syst. Appl.*, vol. 32, núm. 2, pp. 625–631, feb. 2007.

[28] H. Kahramanli y N. Allahverdi, "Design of a hybrid system for the diabetes and heart diseases", *Expert Syst. Appl.*, vol. 35, núm. 1, pp. 82–89, jul. 2008.

[29] M. phi. (Computer S. Rajkumar y G. S. (HOD of B. Reena, "Diagnosis of Heart Disease using Datamining Algorithm", *Glob. J. Comput. Sci. Technol.*, sep. 2010.

[30] S. A. Pattekari y A. Parveen, "PREDICTION SYSTEM FOR HEART DISEASE USING NAIVE BAYES", *Int. J. Adv. Comput. Math. Sci.*, vol. 3, núm. 3, pp. 290–294, 2012.

[31] A. K. Sen, S. B. Patel, y D. P. Shukla, "A Data Mining Technique for Prediction of Coronary Heart Disease Using Neuro-Fuzzy Integrated Approach Two Level", *Int. J. Eng. Comput. Sci.*, vol. 2, núm. 9, pp. 2663–2671, sep. 2013.

[32] "MATLAB - MathWorks". [En línea]. Disponible en: https://www.mathworks.com/products/matlab.html. [Consultado: 17-feb-2016].
[33] H. Sug, "Performance Comparison of Decision Tree Algorithms for Medical Data Sets", *Int. J. Math. Comput. Simul.*, vol. 8, pp. 107–115, 2014.
[34] S. Lekkas y L. Mikhailov, "Evolving fuzzy medical diagnosis of Pima Indians diabetes and of dermatological diseases", *Artif. Intell. Med.*, vol. 50, núm. 2, pp. 117–126, oct. 2010.
[35] M. A. Chikh, M. Saidi, y N. Settouti, "Diagnosis of Diabetes Diseases Using an Artificial Immune Recognition System2 (AIRS2) with Fuzzy K-nearest Neighbor", *J. Med. Syst.*, vol. 36, núm. 5, pp. 2721–2729, oct. 2012.
[36] Y. A. Christobel y Sivaprakasam, "A New Classwise k Nearest Neighbor (CKNN) method for the classification of diabetes dataset", *Int. J. Eng. Adv. Technol.*, vol. 2, núm. 3, pp. 396–200, 2013.
[37] V. A. Kumari y R. Chitra, "Classification Of Diabetes Disease Using Support Vector Machine", *V Anuja Kumari RChitra Int. J. Eng. Res. Appl. IJERA*, vol. 3, núm. 2, pp. 1797–1801, abr. 2013.
[38] K. Bhatia y R. Syal, "Predictive analysis using hybrid clustering in diabetes diagnosis", en *2017 Recent Developments in Control, Automation Power Engineering (RDCAPE)*, 2017, pp. 447–452.
[39] W. Chen, S. Chen, H. Zhang, y T. Wu, "A hybrid prediction model for type 2 diabetes using K-means and decision tree", en *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017, pp. 386–390.
[40] S. Suthaharan, *Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning*. Springer, 2015.
[41] H. Suominen, T. Pahikkala, y T. Salakoski, "Critical points in assessing learning performance via cross-validation", presentado en In Proceedings of the 2nd International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning, 2008, pp. 9–22.
[42] P. Refaeilzadeh, L. Tang, y H. Liu, "Cross-Validation", en *Encyclopedia of Database Systems*, L. LIU y M. T. ÖZSU, Eds. Springer US, 2009, pp. 532–538.
[43] A. Smola y S. V. N. Vishwanathan, *INTRODUCTION TO MACHINE LEARNING*. Cambridge University Press, 2010.
[44] A. Zheng, *Evaluating Machine Learning Models - A Beginner's Guide to Key Concepts and Pitfalls*. United States of America: O'Reilly Media, Inc, 2015.
[45] T. Fawcett, "An introduction to ROC analysis", *Pattern Recognit. Lett.*, vol. 27, núm. 8, pp. 861–874, jun. 2006.
[46] F. Castanedo, *Data Preparation in the Big Data Era - Best Practices for Data Integration*. O'Reilly Media, Inc, 2015.
[47] F. Castanedo, *Advancing Procurement Analytics - Capturing the Long Tail with Simplified Data Preparation*. O'Reilly Media, Inc, 2016.
[48] P. Bruce y A. Bruce, *Practical Statistics for Data Scientists: 50 Essential Concepts*. O'Reilly Media, Inc., 2017.
[49] J. Pérez, Á. Alejandro, de la F. Rodríguez, M. Blanca, y A. Vila, *Estadística*, Primera edición. Av. Tibidabo, 39-43, 08035 Barcelona: Eureca Media, SL, 2011.

[50] B. N. I. Eskelson, H. Temesgen, V. Lemay, T. M. Barrett, N. L. Crookston, y A. T. Hudak, "The roles of nearest neighbor methods in imputing missing data in forest inventory and monitoring databases", *Scand. J. For. Res.*, vol. 24, núm. 3, pp. 235–246, jun. 2009.
[51] I. Guyon y A. Elisseeff, "An Introduction to Variable and Feature Selection", *J. Mach. Learn. Res.*, vol. 3, núm. Mar, pp. 1157–1182, 2003.
[52] S. Haykin, *NEURAL NETWORKS - A Comprehensive Foundation*, Second Edition. Pearson Pretince Hall, 2001.
[53] J. Torres Sospedra, "Ensembles of Artificial Neural Networks: Analysis and Development of Design Methods", Ph.D. Thesis, Universitat Jaume I, 2011.
[54] J. Sarangapani, *Neural network control of nonlinear discrete-time systems*, vol. 21. CRC Press, 2006.
[55] D. E. Rumelhart, E. H. Geoffrey, y J. W. Ronald, "Learning representations by backpropagating errors.", *Cogn. Model. 53*, 1988.
[56] N. Buduma y N. Locascio, *Fundamentals of Deep Learning - DESIGNING NEXT-GENERATION MACHINE INTELLIGENCE ALGORITHMS*, First Edition. United States of America.: O'Reilly Media, Inc., 2017.
[57] R. Hecht-Nielsen, "Theory of the backpropagation neural network", en *International 1989 Joint Conference on Neural Networks*, 1989, pp. 593–605 vol.1.
[58] "ANN BackPropagation", 17-mar-2015. [En línea]. Disponible en: https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/. [Consultado: 15-abr-2017].
[59] S. N. Sivanandam y S. N. Deepa, *Introduction to Genetic Algorithms*. Springer Science & Business Media, 2007.
[60] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT Press, 1998.
[61] "R: The R Project for Statistical Computing", 12-oct-2017. [En línea]. Disponible en: https://www.r-project.org/. [Consultado: 12-oct-2017].
[62] "R: What is R?", 04-dic-2017. [En línea]. Disponible en: https://www.r-project.org/about.html. [Consultado: 04-dic-2017].
[63] "RStudio", 04-dic-2017. [En línea]. Disponible en: https://www.rstudio.com. [Consultado: 04-dic-2017].
[64] Á. A. Juan, B. De La Fuente, y A. Villa, *Estadística*, Primera Edición. Eureca Media, SL, 2011.
[65] J. W. Turkey, *Exploratory Data Analysis*, vol. Vol 2. 1977.
[66] V. Gupta, S. Srinivasan, y S. S. Kudli, "Prediction and Classification of Cardiac Arrhythmia".