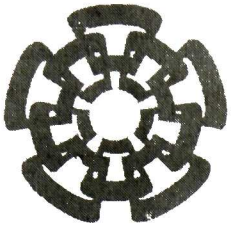


Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

Selección de Atributos para Clasificación Usando Aprendizaje Bayesiano y Computación Evolutiva

Tesis que presenta:

Ana Isabel Rodríguez Barragán

para obtener el grado de:

Maestro en Ciencias

en la especialidad de:

Ingeniería Eléctrica

Directores de Tesis

Dr. Mario Angel Siller González Pico
Dr. Ricardo Vilalta López

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Marzo de 2010.

Selección de Atributos para Clasificación Usando Aprendizaje Bayesiano y Computación Evolutiva

Tesis de Maestría en Ciencias
Ingeniería Eléctrica

Por:
Ana Isabel Rodríguez Barragán
Ingeniero en Sistemas Computacionales
Instituto Tecnológico de Colima 2002-2007

Becario de CONACYT, expediente no. 212756

Directores de Tesis
Dr. Mario Angel Siller González Pico
Dr. Ricardo Vilalta López

CINVESTAV del IPN Unidad Guadalajara, Marzo de 2010.

Tesis de Maestría en Ciencias en Ingeniería Eléctrica

Presentada por:
Ana Isabel Rodríguez Barragán

Para obtener el grado de:
Maestro en Ciencias

Con especialidad en:
Ingeniería Eléctrica

Dr. Mario Ángel Siller González Pico Dr. Ricardo Vilalta López Director(es) de Tesis	Prof. Dr. Eduardo José Bayro Corrochano Sinodal

Dr. Juan Humberto Sossa Azuela
Sinodal Externo

Guadalajara, Jal., 8 de marzo de 2010.

Agradecimientos.

Quisiera agradecer al Prof. Dr. Eduardo Bayro Corrochano por haber sido mi guía a lo largo de este trabajo de tesis, por haberme brindado su tiempo y sus conocimientos además de alentarme a seguir trabajando y no permitir que me rindiera ante la adversidad.

También quiero agradecer al Dr. Mario Siller González Pico por aceptarme como su alumna y por haberme apoyado en la revisión y corrección de este trabajo, así como también al Dr. Ricardo Vilalta por haber creído en mí al inicio del proyecto de tesis.

A mi compañero y amigo el M.C. Eduardo Vázquez Santacruz, por todo su tiempo, sus conocimientos, su paciencia y su amistad.

Al Dr. Humberto Sossa por haberme apoyado con la revisión de este documento a pesar de la premura.

Un especial agradecimiento a mis padres y a mis hermanos Brenda, Isabel y Elías por su apoyo, su cariño, confianza y amor que hicieron posible que yo emprendiera y concluyera este proyecto.

Le doy gracias a Dios por darme licencia de un paso más de evolución material.

Al CONACYT por el apoyo económico recibido.

RESUMEN

Con el rápido avance de las tecnologías en computación se han creado nuevas aplicaciones que producen bases de datos con cientos o incluso miles de atributos o variables. Algunas de estas variables son irrelevantes para las tareas de aprendizaje, causando que los algoritmos de predicción tomen más tiempo en procesar la información o incluso degradando la exactitud en la predicción usando nuevos datos. La selección de atributos tiene como objetivo obtener un subconjunto óptimo de variables que sean necesarias y suficientes para la discriminación entre clases.

En este trabajo analizamos el comportamiento del algoritmo de selección de atributos *Automatic Relevance Determination* (*ARD)* basado en su capacidad para asignar un nivel de relevancia a atributos que presentan correlación entre ellos. Tal algoritmo presenta problemas para determinar la relevancia de atributos correlacionados que pertenecen al mismo nivel de importancia, además de que no es capaz de determinar la longitud mínima del subconjunto.

El objetivo de este trabajo fue demostrar que la combinación de métodos basados en modelos probabilísticos, usando inferencia Bayesiana y algoritmos genéticos y por el principio fundamental de sinergia y cooperación, se puede obtener un método mejor que subsane las deficiencias que presentan los métodos usados individualmente.

ABSTRACT

Fast changes and improvement in information technologies has allowed new applications to be developed that produces databases with hundreds or even thousands of attributes or variables. Some of these variables are irrelevant to learning tasks, causing the prediction algorithms to take a longer time to process information or even degrading the accuracy in prediction when using new data. The selection of attributes is designed to obtain optimal subset of variables which are necessary and sufficient for discrimination between classes.

In this paper we analyze the behavior of the attribute selection algorithm *Automatic Relevance Determination (ARD)*, based on its ability to assign a level of relevance to attributes that have correlation between them. This algorithm has problems to determine the relevance of correlated attributes that belong to the same level of importance, besides is not capable of determining the minimum size of the subset.

The aim of this research was to demonstrate that the combination of methods based on probabilistic modeling using Bayesian inference and genetic algorithms for the fundamental principle of synergy and cooperativity, can be achieved a better method to reduce the deficiencies in the methods used individually.

Índice General

Capítulo I

Introducción **1**

1.1 Antecedentes 1

1.2 Planteamiento del Problema 3

1.3 Propuesta de tesis 4

1.4 Objetivos 5

1.5 Organización de la tesis 5

Capítulo II

Selección de Atributos **7**

2.1 Definición 7

2.1.1 Caracterización de los Algoritmos de Selección de Atributos (ASA) 8

2.1.2 Función de Evaluación (Métricas) 9

2.1.3 Métodos de Búsqueda 10

2.1.4 Categorías de Algoritmos de Selección de Atributos 12

2.2 Métodos Bayesianos y análisis de datos. 13

2.2.1 Redes neuronales como modelos probabilísticos 15

2.2.2 Automatic Relevance Determination (ARD) 17

2.2.3 Algoritmos Evolutivos. 20

2.2.4 Algoritmo Genético 21

Capítulo III

Propuesta de Tesis **26**

3.1 Modelo Híbrido para la Selección de Atributos 26

3.1.1 Trabajo Experimental 27

3.2 Esquema híbrido ARD + Algoritmos Genéticos 34

3.2.1 Esquema Algoritmo Genético-RANDOM 34

3.2.2 Esquema ARD+AG 35

Capítulo IV

Análisis de Resultados **39**

4.1 Trabajo Experimental 39

4.2 Experimentos con Esquema RANDOM 39

4.3 Experimentos con esquema ARD+AG 41

Capítulo V

Conclusiones y Trabajo Futuro **47**

5.1 Conclusiones 47

5.2 Trabajo Futuro 49

Referencias **50**

Índice de Figuras

Figura 2.1. Proceso de selección de atributos según [28]..... 8
Figura 2.2. Arquitectura de una red neuronal.....16
Figura 2.3. Modelado de la red neuronal con múltiples hiperparámetros, uno para cada clase. 19
Figura 2.4. Pseudocódigo de la estructura de un algoritmo evolutivo [45].....21
Figura 2.5. Ejemplo de Selección por Ruleta 23
Fig. 2.6. Ejemplo de operación de cruza o crossover.....24
Figura 3.1. Diagrama de entradas y salidas de la red neuronal. 28
Figura 3.2. Resultado del ranking dado por ARD para la función Seno + Ruido.....28
Figura 3.3. Resultado del ranking dado por ARD para LIC1 con *in6* = dummy.....30
Figura 3.4. Resultado del ranking dado por ARD para LIC1 con *in6* = (*x1-x2*). 30
Figura 3.5. Resultado del ranking dado por ARD con ensamble de MLP para la base de datos LIC1, con *in6* = (*x1-x2*). 32
Figura 3.6. Resultado del ranking dado por ARD con un único MLP para la base de datos LIC4, con in8 = (x1*y2) 33
Figura 3.7. Frecuencia de los atributos de Waveform+noise.....36
Figura 3.8. Esquema Poisson-Uniforme para selección de atributos de ARD.....37
Figura 3.9. Esquema propuesto basado en la distribución Poisson-Uniforme y la Matriz de Correlación MC.....38
Figura 4.1. Rendimiento del AG para cada set de 15 iteraciones del AG con el esquema RANDOM40
Figura 4.2. Resultados del rendimiento del esquema RANDOM para diversos tamaños *n* de individuo.....40
Figura 4.3. Rendimiento del AG para cada set de 15 iteraciones del AG con el esquema ARD+AG .42
Figura 4.4. Resultados del rendimiento del esquema ARD+GA para diversos tamaños *n* de individuo.....42
Figura 4.5. Resultados del rendimiento del esquema ARD+GA para diversos tamaños *n* de individuo.....43
Figura 4.6. Resultado de 30 iteraciones de MLP para los 40 atributos de la base de datos Waveform+noise 44
Figura 4.7. Subconjunto de los primero 27 atributos del ranking de relevancia de ARD. 44
Figura 4.8. Resultado de 30 iteraciones de MLP para los primeros 27 atributos del ranking dado por ARD. 45
Figura 4.9. Subconjunto de atributos que presentaron mejor rendimiento usando el algoritmo genético, con tamaño *n*=27.....45
Figura 4.10. Resultado del MLP para el mejor subconjunto de Poisson-Uniforme+MC con *n*=27...46

Capítulo I

Introducción

La selección de atributos (también llamada selección de variables, selección de sub-espacios o reducción de dimensionalidad) [2] es un procedimiento para seleccionar un subconjunto del conjunto original de atributos o características que describen de manera abstracta una base de datos, de tal manera que los elementos redundantes o no informativos para la búsqueda de patrones sean eliminados, reduciendo a su vez el costo computacional.

En esta tesis analizaremos el comportamiento de un método de selección de atributos el cual se ejecuta con datos cuyos atributos se encuentran altamente correlacionados entre sí. En el análisis se demuestra que dicha correlación afecta el desempeño del método y se propone una mejora usando computación evolutiva para corregir el subconjunto de atributos elegido y obtener un mayor rendimiento al aplicar un algoritmo clasificador.

1.1 Antecedentes

El aprendizaje de máquina es generalmente aplicado a datos almacenados con el objetivo de encontrar en ellos conocimiento no evidente desde el punto de vista humano. Estos datos son presentados en forma de *ejemplos*, los cuales son descritos por un vector de *atributos* que en conjunto se relacionan con una *etiqueta* que sirve para referenciar la *clase* o categoría a la que pertenecen.

Los métodos de reconocimiento de patrones intentan descubrir regularidades o relaciones entre atributos y clases en una fase de entrenamiento para posteriormente usar esa información para la fase de clasificación, etapa en la cual aplican el modelo aprendido a nuevos ejemplos no analizados. Estos ejemplos tienen docenas, cientos o incluso miles de atributos que los describen, los cuales no siempre son necesarios o ideales para el éxito del clasificador. Si hay mucha información irrelevante o redundante, el proceso de entrenamiento se vuelve complejo e ineficiente. Para atacar este problema se ejecutan algoritmos de selección de atributos, cuyo objetivo es identificar y remover toda la información irrelevante o redundante posible y por consiguiente reducir la dimensionalidad o número de atributos que contienen los datos, lo cual a su vez disminuye el tiempo de procesamiento.

La selección de atributos no es una tarea trivial debido a que cada problema presenta características y objetivos distintos, por lo cual se han desarrollado propuestas para aplicar a diferentes disciplinas de las ciencias. Existen varios trabajos que intentan describir las generalidades de la selección de atributos, recopilando diversas definiciones y esquematizando sus características [2], [3], [4]. Otros trabajos se enfocan en explicar el fundamento de algunos algoritmos, estudiando sus ventajas y desventajas con diversos enfoques [5], [6] e incluso intentando unificarlos [7].

Existen estudios más profundos acerca del comportamiento de los algoritmos de selección de atributos en presencia de factores como dependencia y correlación, como en [8] donde se busca explotar las dependencias de los atributos, o bien como [9] y [10] que analizan algoritmos capaces de trabajar sólo con pares de atributos correlacionados. Isabelle Guyon & Elisseeff [11] hace un análisis acerca de cómo la correlación impacta la redundancia de los atributos, concluyendo que las variables perfectamente correlacionadas son redundantes, pero sólo un alto grado de correlación no significa que las variables no se complementen entre ellas y aporten información valiosa al clasificador.

Respecto a métodos especializados de selección de atributos que se ven afectados por la presencia de elementos correlacionados está *Automatic Relevance Determination (ARD)* [12], el cual es el caso de estudio de este trabajo. Yu Fu & Antonie Browne [13] analizan el método determinando que la presencia de correlación afecta el desempeño en la evaluación, concluyendo que el método puede hacer sólo un *ranking* global de relevancia pero falla al predecir el grado de importancia para atributos individuales.

En [14] muestran que tener un gran número de atributos de entrada para ARD ocasiona el sobre-entrenamiento de la red neuronal sobre la cual funciona el algoritmo. Un esfuerzo por incorporar a ARD un enfoque evolutivo se encuentra en [15], donde los

autores usan algoritmos genéticos para diseñar y entrenar un ensamble de redes neuronales con la finalidad de obtener un mejor subconjunto de atributos. Otros trabajos que buscan mejorar ARD desde otras perspectivas, sin tomar en cuenta la correlación, pueden ser encontrados en [16] [17].

Existen también artículos donde se usan algoritmos genéticos para mejorar la selección explotando la correlación existente entre los atributos. En [18] usan una etapa de filtro que cuantifica la correlación de los atributos con la clase y posteriormente selecciona los más correlacionados para inicializar el algoritmo genético. Otro ejemplo es [19], donde proponen un esquema de dos algoritmos genéticos anidados, donde la entrada para el primer algoritmo son cúmulos de datos altamente correlacionados.

En todos estos trabajos se han encontrado debilidades tanto en el algoritmo ARD como en los métodos evolutivos usados para selección de atributos, pero hasta el momento ningún autor ha propuesto la combinación de ambos métodos para subsanar las debilidades individuales.

1.2 Planteamiento del Problema

El tiempo que toma a un algoritmo de aprendizaje de máquina en procesar un conjunto de datos está directamente relacionado con el tamaño de tal conjunto, mientras más datos irrelevantes se deban analizar el tiempo se incrementa y el rendimiento del algoritmo clasificador decrece. La selección de atributos es un método usado para reducir la dimensionalidad de un conjunto de datos y así disminuir el tiempo de procesamiento, sin degradar el rendimiento del algoritmo clasificador. La selección se basa en un análisis de la relación atributo-clase, como una medida de importancia individual para cada atributo. Sin embargo este criterio suele no contemplar que gran parte de la información a analizar presenta fuerte correlación entre atributos, es decir, atributos que contienen básicamente la misma información y no ayudan al proceso de predicción del clasificador, pero que igualmente están relacionados con la clase. En estas situaciones el algoritmo de selección no puede discriminar estos atributos y como resultado aumenta el tiempo de procesamiento y disminuye el rendimiento del algoritmo clasificador. Este problema hace necesario no solo medir la correlación atributo-clase, sino también analizar la relevancia entre atributos teniendo como referencia la *correlación* presente entre los mismos. Si un atributo no está correlacionado ni con otros atributos ni con la clase, es considerado como ruido o información inútil.

Los algoritmos de selección de atributos se ven afectados por la presencia de elementos correlacionados que no permiten obtener un subconjunto de atributos óptimo que mejore la tarea de clasificación. Este problema nos permite plantear las siguientes preguntas de investigación:

a) ¿Es posible probar de manera empírica que la correlación entre atributos afecta el desempeño de los algoritmos de selección de atributos?

b) ¿Es posible usar un algoritmo de selección de atributos como caso de estudio para realizar un análisis de su comportamiento con elementos correlacionados?

c) ¿Existe algún método que permita mejorar los resultados obtenidos por el algoritmo de selección de atributos en términos de tamaño mínimo de subconjunto y elementos a seleccionar cuando existe correlación en sus datos, sin degradar el rendimiento en la fase de clasificación?

Analizar el comportamiento de los algoritmos de selección de atributos con relación a las características intrínsecas de los datos, nos permite encontrar deficiencias y trabajar en métodos para mejorar su rendimiento.

1.3 Propuesta de tesis

En esta tesis se analiza un caso particular del algoritmo de selección de atributos llamado "*Automatic Relevance Determination (ARD)*", el cual es capaz de separar de manera global el conjunto de atributos relevantes de los irrelevantes, pero falla al discriminar elementos con el mismo nivel de relevancia [12]. Esto significa que el resultado de ARD consistente en una lista o "*ranking*" de atributos ordenados por su nivel de relevancia contiene elementos mal posicionados, lo cual hace difícil decidir cuál es el subconjunto óptimo basándose solo en la lista resultante.

Cuando dos o más atributos se encuentran altamente correlacionados es necesaria la búsqueda de la mejor combinación de elementos que ayuden a decidir entre discriminar uno u otro atributo. Para poder encontrar una solución al conflicto de posiciones en el resultado preliminar que arroja el algoritmo ARD se propone usar Computación Evolutiva (Algoritmos Genéticos) [20]. Los Algoritmos Genéticos son una clase de algoritmos de búsqueda estocástica capaces de explorar y explotar todo el espacio de soluciones, evitando estancarse en un mínimo local, lo que le da su potencia y su capacidad para

encontrar soluciones globales. Debido a su potencial son ampliamente usados para muchos problemas de selección de atributos [21][22][23].

Lo anterior nos permite plantear la siguiente

Hipótesis

H_0. Es posible mejorar el rendimiento del algoritmo de selección de atributos ARD en función de la obtención de un subconjunto óptimo y con el tamaño mínimo, manteniendo el rendimiento del algoritmo clasificador.

1.4 Objetivos

El objetivo de este trabajo es presentar un esquema de selección de atributos para clasificación donde los atributos contengan un alto grado de correlación, del cual se obtenga como resultado la minimización del tamaño del subconjunto de atributos elegido y la elección de sólo los elementos realmente relevantes, manteniendo o incluso mejorando el rendimiento del algoritmo clasificador al usar tal subconjunto. Los objetivos de la propuesta de tesis son:

a) Definir una categorización o jerarquía de atributos usando el algoritmo ARD como primera aproximación.
b) Aplicar un algoritmo genético especialmente adaptado para selección de atributos como método de búsqueda que refine la jerarquía proporcionada por la aplicación del algoritmo ARD sobre atributos altamente correlacionados.
c) Obtener el tamaño mínimo de subconjunto para cada problema.

El esquema propuesto presenta la unión de dos métodos distintos: provee una aproximación probabilística al usar el Teorema de Bayes que trabaja sobre la evidencia obtenida de cada atributo y refina sus resultados usando un método evolutivo capaz de encontrar la mejor combinación de atributos y descartar los irrelevantes.

1.5 Organización de la tesis

Este trabajo está organizado de la siguiente manera: en el capítulo 2 se describe el estado del arte de los algoritmos de selección de atributos, sus métricas actuales y tipos, así como el fundamento del algoritmo ARD. También se da una introducción a los algoritmos genéticos. En el capítulo 3 se describe el análisis experimental realizado para

analizar el comportamiento de ARD con datos correlacionados. Se describe también la propuesta de refinamiento del algoritmo ARD usando algoritmos genéticos y explotando la correlación en los datos. En el capítulo 4 se describen los resultados obtenidos de la experimentación. Con el capítulo 5 se concluye este trabajo resaltando nuestros principales logros y sugiriendo nuevas direcciones de investigación que surgieron de esta propuesta.

Capítulo II

Selección de Atributos

2.1 Definición

Los métodos de Selección de Atributos [2] contribuyen a que los algoritmos de predicción trabajen sólo con los atributos más representativos para predecir la clase objetivo. Un gran número de atributos como entrada para los algoritmos de inducción (clasificación) puede aumentar exponencialmente el consumo de tiempo de procesamiento y volverlos ineficientes en el uso de memoria. Desafortunadamente, muchos de los atributos de entrada son tanto parcial o totalmente irrelevantes, o bien redundantes a la clase. Un atributo irrelevante no mejora el resultado de la clasificación, mientras que un atributo redundante no sólo no aporta información nueva para la tarea, sino que puede confundir a los algoritmos de aprendizaje haciéndolos llegar a falsas conclusiones y empeorando su desempeño [24].

Un Algoritmo de Selección de Atributos (ASA) utiliza algo muy cercano a la definición de relevancia[1] al usar heurísticas que miden qué tan predictivo es un subconjunto para la clasificación. Dependiendo el problema sobre el cual se trabaja podemos considerar dos tipos de selección de atributos: *continua*, la cual conserva todos los elementos $x_i \in X$ ponderándolos con un peso ω_i; y *binaria*, que se refiere a la asignación de pesos binarios 1 y 0 que determinan si un atributo debe o no aparecer en el subconjunto final [25] [26]. La selección de atributos se define como:

[1] Cuando un atributo es considerado como importante o significativo para predecir la clase.

Sea $J(X')$ una medida de evaluación a ser optimizada definida como $J: X' \subseteq X \rightarrow \Re$. La selección de un subconjunto de atributos puede ser vista bajo tres consideraciones:

- *Sea* $|X'| = m < n$. *Encontrar* $X' \subset X$, *tal que* $J(X')$ *es el óptimo.*
- *Fijemos un valor* J_0 *como el mínimo (o máximo)* J *que puede ser tolerado. Encontrar* $X' \subseteq X$ *con el* $|X'|$ *más pequeño, de tal manera que* $J(X') \geq J_0$.
- *Encontrar un compromiso entre minimizar* $|X'|$ *y maximizar* $J(X')$.

Como se aprecia, bajo estas definiciones, un subconjunto óptimo de atributos no es único [3].

2.1.1 Caracterización de los Algoritmos de Selección de Atributos (ASA)

Los ASA se usan en problemas que implican una búsqueda en el espacio de hipótesis. [27] describe un ASA como un esquema de "cuatro problemas identificados para los métodos de selección de atributos", los cuales son agrupados de la siguiente manera:

a) Un método de búsqueda a través del espacio de atributos, que consiste en:
   - Selección de un subconjunto de atributos inicial
   - Generación del siguiente subconjunto de prueba
   - Criterio de parada

b) Una función de evaluación que determina si un subconjunto de atributos es o no considerado óptimo.

Las partes básicas de un ASA se presentan en la Figura 1, donde los tres sub-problemas pertenecientes al método de búsqueda han sido agrupados, mientras que la función de evaluación se define aparte como una métrica de evaluación-aceptación (ver Figura 2.1).

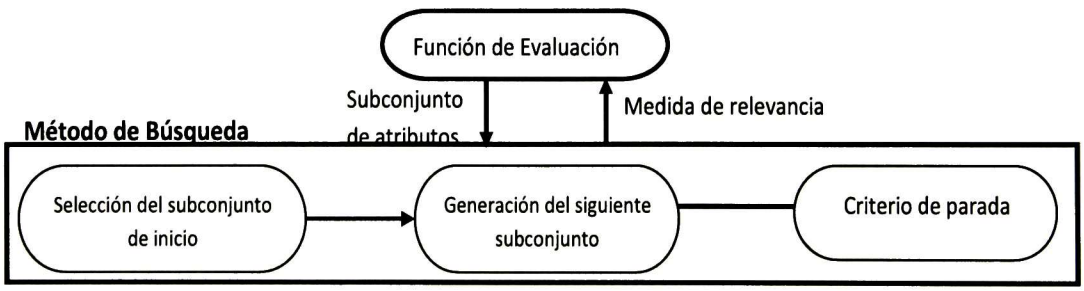


Figura 2.1. Proceso de selección de atributos según [28].

2.1.2 Función de Evaluación (Métricas)

La función de evaluación define una medida de predicción de un subconjunto de atributos para el algoritmo clasificador. Para esto, se toma al subconjunto de prueba S y un conjunto de datos de entrenamiento T y se define una función de tipo $S \times T \rightarrow \mathbb{R}$ que nos provee un valor cuantitativo para el subconjunto.

Hay muchas maneras de definir una función de evaluación dependiendo del criterio o del problema. Un subconjunto óptimo es siempre relativo a cierta función de evaluación, es decir, que un subconjunto elegido como óptimo por una función de evaluación puede no ser el mismo si se usa otra función de evaluación. Dash & Liu [2] dividen las funciones de evaluación en cinco categorías: *distancia, información (incertidumbre), dependencia, consistencia y tasa de error del clasificador.*

a) Distancia (divergencia, separabilidad): Se usa para decidir cuál de dos atributos puede definir más acertadamente la diferencia entre clases usando el cálculo de la distancia de su probabilidad condicional o bien la distancia representada en un espacio. Un ejemplo es la medida de distancia Euclidiana.

b) Información (o incertidumbre): Esta medida determina la ganancia de información que proporciona un atributo; esto es, calculamos la probabilidad posteriori y la comparamos con la probabilidad a priori estimada, para saber qué tanta información se ha ganado con el uso de cierto atributo. Preferimos un atributo si éste provee más información que otro, una manera de medir la información obtenida es la Entropía de Shannon[29].

c) Dependencia: Mide el grado de relación entre dos variables, ya sea entre un atributo y la clase o entre atributos, en el sentido en que si conocemos el valor de un atributo podemos predecir el valor de otro. Elegimos un atributo sobre otro si su dependencia con la clase es mayor. En el caso que exista dependencia de un atributo con otro atributo la medida se convierte en un grado de redundancia. Un estudio de estas medidas se encuentra en el trabajo de Ben-Basset [30].

d) Consistencia: Se indica que para poder predecir una clase, el subconjunto de atributos debe ser consistente, en el sentido en que dos atributos con el mismo valor no deben pertenecer a diferente clase.

e) Tasa de Error del Clasificador: Es una medida usada en los algoritmos llamados "*wrapper*" en donde la decisión de elegir un subconjunto de prueba sobre

otro está basada en la cantidad de errores que éste genera al ser usado por un algoritmo de clasificación. Es también la métrica usada en este trabajo de tesis.

2.1.3 Métodos de Búsqueda

Usualmente los métodos de selección de atributos buscan a través de subconjuntos de atributos intentando encontrar el mejor entre los 2^n subconjuntos candidatos según el criterio previamente definido, sin embargo esta estrategia es una búsqueda exhaustiva que es muy costosa y poco práctica incluso para un tamaño medio de n, lo que lo convierte en un problema NP-Hard [31]. Esta exploración puede ser dirigida a través del espacio de atributos usando una estrategia específica (búsqueda hacia adelante, búsqueda hacia atrás o bien búsqueda combinada) y un criterio de asignación de pesos a cada atributo, que puede ser especificado con valores continuos donde el peso $\omega i \in [0,1]$, o bien discretos, con $\omega i \in \{0,1\}$. Esta asignación determina cuales atributos formarán parte del nuevo subconjunto de prueba que sea posteriormente analizado por la función de evaluación.

Los métodos de búsqueda se definen como una serie de estados posibles alcanzables desde el estado actual. Pueden ser de diversos tipos:

- Búsqueda Secuencial.

  Selecciona entre uno de los posibles estados alcanzables desde el estado actual. El número de pasos está limitado por $O(n)$ y una vez que se avanza hacia un estado no se puede retroceder, por lo que no se garantiza encontrar la solución óptima si ésta se encuentra en uno de los estados no visitados.

- Búsqueda Exponencial.

  Corresponde a una búsqueda con costo computacional acotado por $O(2^n)$. No necesariamente es una búsqueda exhaustiva si se usa una buena heurística. Su ventaja es que garantiza encontrar la solución óptima.

- Búsqueda Aleatoria.

  Ejecuta una búsqueda en el espacio completo de atributos, lo cual evita estancarse en un mínimo local. Estos algoritmos son llamados *AnyTime Algorithms* [32].

Una vez elegido el tipo de búsqueda, deben considerarse sus sub-problemas, como se menciona en la sección 2.2. Estos sub-problemas son:

1) Generación del subconjunto inicial de atributos.

Se puede iniciar ya sea con un subconjunto de prueba vacío $X' = (\emptyset)$, con un subconjunto que contenga todos los atributos donde $|X'| = n$, donde n es el número total de atributos, o bien con un subconjunto inicial aleatorio.

2) Generación del siguiente subconjunto de atributos.

Este paso depende de cuál fue la elección del subconjunto inicial. Si empezamos con un subconjunto vacío, la opción es seguir una estrategia **hacia adelante**, donde vamos añadiendo elementos al conjunto iterativamente, como sigue:

$$X' := X' \cup \{x_i \in X \setminus X' \mid J(X' \cup \{x_i\}) \; es \; mayor\} \tag{2-1}$$

En caso de haber comenzado con un subconjunto que contiene todos los n atributos posibles, la estrategia será **hacia atrás**, donde se van eliminando uno a uno los elementos que hacen que el valor de la función de evaluación $J(X')$ se maximice:

$$X' := X' \setminus \{x_i \in X' \mid J(X' \setminus \{x_i\}) \; es \; mayor\} \tag{2-2}$$

Este método tiene la ventaja de poder analizar la interacción entre atributos, a diferencia de la estrategia hacia adelante, aunque conlleva un mayor costo computacional.

Usando estas ideas podemos usar una **combinación** de estrategias, avanzando hacia adelante un número k de pasos y retrocediendo un número l, dado por la maximización o minimización de la función de evaluación. También podemos considerar la estrategia de selección **aleatoria**, que puede generar subconjuntos difícilmente alcanzables usando las estrategias anteriores.

3) Criterio de Parada

Se debe definir un criterio que permita detener la búsqueda de subconjuntos dependiendo del problema. Un criterio de parada puede ser si se ha llegado al número mínimo de atributos que pueden ser elegidos o si se ha sobrepasado el número de iteraciones posibles, o bien si el agregar o eliminar atributos del subconjunto no mejora el resultado de la función de evaluación.

2.1.4 Categorías de Algoritmos de Selección de Atributos

Se distinguen tres categorías o esquemas en los ASA:

Esquema de Filtro

Cuando el proceso de selección del subconjunto de atributos de prueba se da independientemente de la fase de evaluación. Se dice que este esquema "filtra" el conjunto de atributos seleccionando sólo los que proveen más información a la clase. Su ventaja es su bajo costo computacional, pero no analiza relaciones entre los atributos.

Esquema "Wrapper"

Este esquema es especial para analizar las relaciones existentes entre los atributos. Primero elige un subconjunto de prueba y lo analiza con un algoritmo de clasificación para medir la tasa de error. Su principal ventaja es su capacidad para encontrar dependencias (redundancias) entre los atributos, pero en contra tiene el alto costo computacional de llamar al clasificador como un módulo extra para evaluar cada posible subconjunto.

Esquema Embebido

Es una combinación unitaria de los esquemas anteriores, donde la selección del subconjunto de atributos y su evaluación se perciben como un solo algoritmo indivisible.

2.2 Métodos Bayesianos y análisis de datos.

El aprendizaje de redes neuronales es interpretado como una inferencia acerca de los parámetros más probables de la red dados los datos que se desean analizar. La teoría de probabilidad Bayesiana aplicada a redes neuronales provee un marco de trabajo donde el principal objetivo es encontrar modelos (configuraciones de arquitecturas, modelos de ruido, pre-procesamiento, constantes de regularización, etc.) de redes neuronales que se adapten a los datos y proporcionen predicciones acertadas en la fase de clasificación. Esto sigue de la idea que el mejor subconjunto de parámetros del modelo es conformado por aquellos parámetros que son más probablemente los generadores del subconjunto de datos que estamos analizando.

En el enfoque Bayesiano se inicia con una distribución de probabilidad *a priori* $p(w)$, la cual expresa nuestro conocimiento de los parámetros antes de que los datos sean observados. Una vez observados, podemos usar el Teorema de Bayes [33] para actualizar los valores y obtener la densidad de probabilidad *posterior* $p(w|D)$. Como algunos valores de los atributos son más consistentes con los datos que otros, la distribución posterior indica que un grupo menor de valores que el definido en la probabilidad a priori es más adecuado. Con una buena estimación de una probabilidad a priori se puede evaluar la importancia de cada variable de entrada para la fase de predicción, es decir, hacer un proceso de selección de atributos. Este proceso es llamado *Automatic Relevance Determination (ARD)*[12] [34].

Los métodos Bayesianos son capaces de modelar los datos desde dos niveles de inferencia. El teorema de Bayes para los dos niveles se desarrolla basado en un modelo H_i que posee un vector de parámetros $\mathbf{w}$ y donde cada modelo es definido por la colección de distribuciones de probabilidad a 'priori' $P(\boldsymbol{w}|H_i)$, que son valores que se espera que el modelo tome, y un conjunto de distribuciones condicionales que definen las predicciones $P(D|\boldsymbol{w}, H_i)$ que el modelo hace acerca del conjunto de datos D.

En el primer nivel sólo asumimos que el modelo que proponemos es verdadero e inferimos los posibles valores para los parámetros $\mathbf{w}$, dados los datos D. Usando el teorema de Bayes, la probabilidad posterior de los parámetros $\mathbf{w}$ es:

$$P(\boldsymbol{w}|D, H_i) = \frac{P(D|\boldsymbol{w}, H_i)P(\boldsymbol{w}|H_i)}{P(D|H_i)} \tag{2-3}$$

que es,

$$Posterior = \frac{Likelihood \times Prior}{Evidence}$$

La constante de normalización $P(D|H_i)$ es también llamada la *evidencia* de H_i, importante para la comparación entre modelos. Es común usar métodos basados en gradiente descendiente para encontrar el máximo del posterior, el cual define los valores *más probables* para los parámetros $\mathbf{w}_{MP}$. Los intervalos de confianza (o barras de error) son calculados evaluando la matriz Hessiana en $\mathbf{w}_{MP}$:

$$\mathbf{A} = -\nabla\nabla \log P(\mathbf{w}|D, H_i)|_{\mathbf{w}_{MP}} \tag{2-4}$$

Y expandiendo con series de Taylor el logaritmo de la probabilidad posterior con $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}_{MP}$,

$$P(\mathbf{w}|D, H_i) \cong P(\mathbf{w}_{MP}|D, H_i) \exp(-\frac{1}{2}\Delta\mathbf{w}^{\mathbf{T}}\mathbf{A}\Delta\mathbf{w}) \tag{2-5}$$

Con lo que vemos que el posterior puede ser localmente aproximado a una distribución Gaussiana con matriz de covarianza (equivalente a barras de error) $\mathbf{A}^{-1}$.

En el segundo nivel de inferencia deseamos poder inferir cual modelo es más adecuado dados los datos, para lo cual se calcula la probabilidad posterior de cada modelo como se muestra en la ecuación (2-6).

$$P(H_i|D) \propto P(D|H_i)P(H_i) \tag{2-6}$$

Donde $P(D|H_i)$ es la evidencia de H_i, el segundo término $P(H_i)$ es el priori que expresa una preferencia por un modelo antes de ver los datos. Si el priori fuera idéntico para todos los modelos, entonces el valor determinante para listarlos es el de la evidencia. Es con estas aproximaciones que los métodos Bayesianos intentan complementar las soluciones provistas por las redes neuronales.

2.2.1 Redes neuronales como modelos probabilísticos

Las redes neuronales son dispositivos computacionales no-lineales inspirados en la estructura biológica del cerebro humano, capaces de "aprender" a resolver problemas de predicción o clasificación al crear modelos que son compatibles con los datos. El problema que presentan las redes neuronales es que son influenciadas por correlaciones en los datos, de tal manera que fallan al encontrar los parámetros que describan el modelo de red adecuado al problema. Es ahí donde los métodos Bayesianos juegan un papel complementario, infiriendo los valores más probables para el modelo dados los datos.

Una red neuronal supervisada de tipo Multi–Capa (MLP)[33] es un mapa no-lineal parametrizado de una entrada $\boldsymbol{x}$ a una salida $\boldsymbol{y} = \boldsymbol{y}(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{A})$. La salida es una función continua de parámetros $\boldsymbol{w}$ y la arquitectura de la red es denotada por $\boldsymbol{A}$. La arquitectura de la red usando sólo una capa oculta tiene la forma:

$$\begin{aligned} &\text{Capa Oculta:} \quad a_j^{(1)} = \textstyle\sum_l w_{jl}^{(1)} x_l + \theta_j^{(1)} \qquad h_j = f^{(1)}\left(a_j^{(1)}\right), \\ &\text{Capa de Salida:} \quad a_i^{(2)} = \textstyle\sum_j w_{ij} x_l + \theta^{(1)} \qquad h_j = f^{(1)}\left(a_j^{(1)}\right) \end{aligned} \tag{2-7}$$

donde $f^{(1)}(a) = \tanh(a)$ es una función sigmoide y $f^{(2)}(a) = a$ es lineal (ver Figura 2.2). Los pesos w y *biases* θ juntos hacen el vector de parámetros $\mathbf{w}$.

La red usa un conjunto de datos $D = \left\{\mathbf{x}^{(m)}, \mathbf{t}^{(m)}\right\}$, donde $\mathbf{t}$ son las salidas u objetivos de la red, que pueden tomar valores continuos para problemas de regresión, o bien valores discretos (binarios) para problemas de clasificación. La red usa estas salidas para ajustar iterativamente los parámetros $\mathbf{w}$, que a su vez minimiza la función objetivo:

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_m \sum_i \left(t_i^{(m)} - y_i\left(\mathbf{x}^{(m)}; \mathbf{w}\right)\right)^2 \tag{2-8}$$

o bien:

$$G(\mathbf{w}) = \sum_m t^{(m)} \log y\left(\mathbf{x}^{(m)}; \mathbf{w}\right) + \left(1 - t^{(m)}\right) \log\left(1 - y\left(\mathbf{x}^{(m)}; \mathbf{w}\right)\right), \tag{2-9}$$

que corresponde a problemas de clasificación con salida binaria.

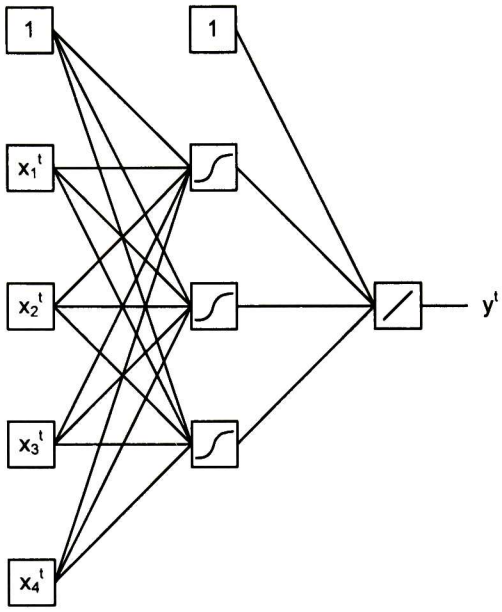


Figura 2.2. Arquitectura de una red neuronal.

Esta minimización se realiza calculando el gradiente de E_D con el método de "*backpropagation*" [35], agregando además un parámetro de regularización llamado "*weight decay*" mostrado en la Ecuación (2-10), el cual favorece los valores pequeños de **w** y evita el sobreentrenamiento.

$$M(w) = \beta E_D + \alpha E_W \tag{2-10}$$

si la función objetivo usada para problemas de regresión, o bien

$$M = -G + \alpha E_W, \tag{2-11}$$

para problemas de clasificación binaria (no incluye el parámetro βE_D), donde

$$E_W = \frac{1}{2}\sum_i w_i^2 \tag{2-12}$$

Estas funciones objetivo contienen dos elementos α y β, los cuales son llamados *hiperparámetros*.

2.2.2 Automatic Relevance Determination (ARD)

La selección de un subconjunto óptimo de atributos relevantes para clasificación es un problema común en muchas aplicaciones de aprendizaje de máquinas. El método de selección de atributos Automatic Relevance Determination [12] trabaja sobre una red neuronal complementada con el marco de trabajo Bayesiano descrito en la sección anterior y haciendo uso de hiperparámetros asociados a cada una de los atributos para determinar su relevancia, midiendo la prominencia de los pesos obtenidos con el cálculo de la matriz Hessiana.

En el modelo de ARD se asocia un hiperparámetro α a cada variable de entrada. Para problemas de clasificación, como es el caso de esta tesis, se usa la función objetivo (2-11) reemplazando βE_D por $-G$, y eliminando el factor de ruido β.

El hiperparámetro α tiene su fundamento probabilístico basado en la regularización de los pesos, factor representado por un priori Gaussiano con media cero para el *weight decay*:

$$P(\mathbf{w}|\alpha, H) = \frac{1}{Z_W(\alpha)} \exp(-\alpha E_W), \tag{2-13}$$

donde α representa la varianza inversa de la distribución y de la constante de normalización dada por

$$Z_W(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{W/2} \tag{2-14}$$

Dado que α es el parámetro para la distribución de otros parámetros (los pesos y los *biases* **w**) es nombrado como hiperparámetro, y se usa además para inferir los valores de los pesos al penalizar aquéllos con grandes magnitudes.

Diferentes valores para los hiperparámetros definen diferentes sub-modelos. Es posible cambiar el valor de estos hiperparámetros durante el entrenamiento de la red usando la *evidencia* obtenida de los datos:

$$P(\alpha, \beta|D, H) = \frac{P(D|\alpha, \beta, H)P(\alpha, \beta|H)}{P(D|H)} \tag{2-15}$$

El máximo de la evidencia $\alpha = \alpha_{MP}$ satisface la siguiente ecuación:

$$\alpha_{MP} = \frac{\gamma}{\sum_i w_i^{MP^2}} \tag{2-16}$$

donde $\mathbf{w}^{\mathbf{MP}}$ es el vector de parámetros usado para minimizar la función objetivo *Mw* (ecuación (2-10)) y γ es el *'número de parámetros bien determinados'* dado por:

$$\gamma = k - \alpha Trace(\boldsymbol{A}^{-1}), \tag{2-17}$$

donde *k* es el número total de parámetros, y la matriz $\boldsymbol{A}^{-1}$ es la matriz de varianza-covarianza que define las barras de error (intervalos de confianza) en los parámetros **w**. Por lo tanto $\gamma \rightarrow k$ cuando los parámetros son todos bien determinados en relación con su rango de valores a priori, el cual está definido por α. La cantidad γ siempre está entre 0 y *k*.

Por simplicidad, se ha asumido que hay una sola clase de pesos, a la cual corresponde solo un hiperparámetro α. Sin embargo los pesos de la red neurona dada por (2-7) se dividen en tres o más clases distintas, por ejemplo $\{w^{(1)}\}$, $\{\theta^{(1)}\}$ y $\{w^{(2)}, \theta^{(2)}\}$. Por consistencia, los pesos de diferentes clases no deben ser modelados como provenientes de un solo priori, sino con un priori asignado a cada una de las clases (ver Figura 2.3). Asumiendo un priori Gaussiano para cada clase, se define:

$$E_{W(c)} = \sum_{i \in c} \frac{w_i}{2}, \tag{2-18}$$

y asignamos el priori:

$$P(\{w_i\}|\alpha_c, H) = \frac{1}{\prod z_{W(c)}} \exp\left(-\sum_c \alpha_c E_{W(c)}\right), \tag{2-19}$$

donde cada clase tiene ahora su propio hiperparámetro o tasa de *'weight decay'* α_c.

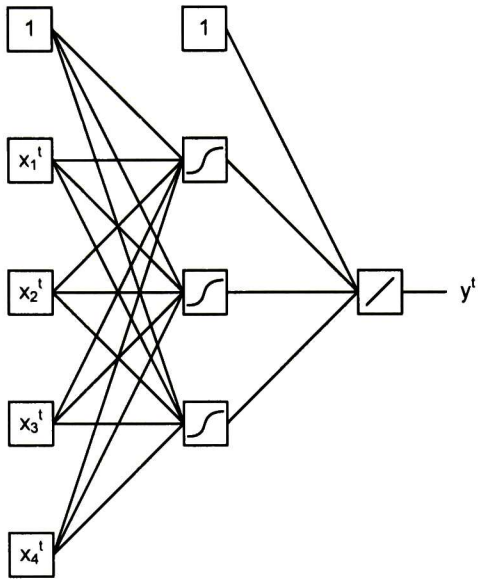


Figura 2.3. Modelado de la red neuronal con múltiples hiperparámetros, uno para cada clase.

De esta manera usando el priori y la evidencia sobre los hiperparámetros el modelo es capaz de inferir cuáles variables son relevantes y llevar hacia cero aquéllas que no lo sean. Esto es logrado de un modo 'suave' al introducir múltiples constantes de 'weight decay' asociadas una a cada entrada o atributo. Las constantes aplicadas a entradas irrelevantes tomarán valores grandes, lo cual hará si varianza inversa pequeña y dichas entradas no tendrán impacto en el proceso de entrenamiento y por consecuente en la salida de la red.

Algunos de los problemas que presenta éste método están relacionados con el tratamiento de los datos. Uno de ellos consiste en que, dado que ARD no elimina las entradas que ha marcado como irrelevantes, es necesario seguir usándolas durante el proceso de entrenamiento, lo que incrementa el tiempo de ejecución. Otro problema es la presencia de un gran número de atributos irrelevantes, lo cual interfiere en el cálculo de la evidencia para diferentes modelos, así como en el de los determinantes de la matriz Hessiana.

Estos problemas no interfieren con el cálculo de la optimización Bayesiana de las constantes de regularización o hiperparámetros, sin embargo diversos estudios [36][13] demuestran con varios experimentos que ARD sí presenta fallas al inferir la relevancia de algunas entradas, especialmente al listar atributos por orden de relevancia cuando dichas entradas presentan correlación o algún grado de redundancia entre ellas.

2.2.3 Algoritmos Evolutivos.

A lo largo de millones de años las formas de vida existentes sobre la Tierra sufrieron grandes cambios a través del proceso de evolución. En este proceso los seres vivos tuvieron que pasar por profundos cambios físicos y de comportamiento para poderse adaptar ante condiciones adversas[37]. Charles Darwin fue el primero en explicar tal proceso de evolución por selección natural a través de mecanismos como capacidad de supervivencia (adaptabilidad al medio ambiente, la fuerza, la competencia entre organismos, etc.) y la capacidad de reproducción (tamaño de la descendencia, tiempo gestacional, etc. [38].

Inspirados por la teoría de Darwin de la evolución por selección natural se desarrollaron hace algunas décadas un grupo de métodos estocásticos utilizados para problemas de búsqueda y optimización, conocidos como Algoritmos Evolutivos.

Los Algoritmos Evolutivos trabajan con una población de individuos que representan posibles soluciones a un problema, aplicando el principio de la supervivencia del más apto para producir mejores individuos. En cada generación de una nueva población en la que se genera un proceso de evolución que guía a los individuos de esta generación para una mejor adaptación al problema [39]. Por lo tanto, los algoritmos evolutivos difieren de otros métodos, tales como *Hill Climbing* [40] y *Simulated Annealing* [41], por utilizar una estrategia de búsqueda basada en la optimización de una población formada con posibles soluciones a un problema en lugar de utilizar sólo una solución [42].

Básicamente los algoritmos evolutivos trabajan de la siguiente manera: inicialmente se genera una población aleatoria de individuos, donde cada individuo representa una posible solución al problema. Una función objetivo se utiliza para medir el rendimiento de cada individuo para resolver el problema. A continuación se lleva a cabo un proceso de selección de los individuos con un mejor rendimiento a ser parte de una nueva población. Estos individuos pasan por transformaciones unitarias (mutación), que crean nuevos individuos al hacer una pequeña modificación al individuo original, y por transformaciones de orden superior (crossover), que crean nuevos individuos mediante la combinación de dos o más individuos de la población original. Estos individuos forman una nueva población que es evaluada y el procedimiento continúa hasta que hay una convergencia de los resultados o hasta que otro criterio de parada sea satisfecho. Por lo tanto, se espera que la mejor solución encontrada sea cercana a la solución óptima [42][43][44].

Una estructura de un algoritmo evolutivo es ilustrada en la Figura 2.4

1. Genera inicialmente una población aleatoria G(0);
2. Asignar i = 0;
3. REPETIR:
   (a) Evaluar a cada individuo en la población;
   (b) Seleccionar a los individuos de la población G(i) con mejor desempeño;
   (c) Someter a estos individuos a transformaciones que produzcan una nueva población G(i+1);
   (d) i = i + 1;
4. REPETIR hasta que el criterio de parada sea satisfecho.

Figura 2.4. Pseudocódigo de la estructura de un algoritmo evolutivo [45].

2.2.4 Algoritmo Genético

Probablemente, el algoritmo genético es la técnica de optimización más difundida de la familia algoritmos evolutivos. Presentado inicialmente por *Holland* [12] en 1962, el algoritmo genético utilizaba sólo los operadores de selección y de cruce. Sin embargo, pronto fue incorporado en el operador de mutación, y con el surgimiento de la mayoría de los investigadores interesados en la técnica, ha ganado con el tiempo tantas versiones y modificaciones que se hace difícil distinguir entre el algoritmo genético y otros algoritmos evolutivos [44].

Como se ha mencionado acerca de los algoritmos evolutivos, el algoritmo genético se compone de una población de individuos que representan soluciones que tienden a mejorar en las generaciones futuras.

El uso de algoritmos genéticos requiere la especificación de los seis procedimientos clave [46]:

Representación o codificación de la solución: Antes de resolver cualquier problema, el algoritmo genético que normalmente representa en cada individuo de la población como un vector que se compone de elementos de un determinado tipo de alfabeto. Podría consistir en un alfabeto de dígitos binarios, números reales, números enteros, símbolos, etcétera [46]. La representación clásica se realiza por un vector de ceros

y unos (binario). Este régimen de representación discreta es un poco más cerca del modelo natural de ADN que la mayoría de las técnicas de Algoritmos Evolutivos [44]. Sin embargo, *Michalewicz* en [43] realizó varios experimentos comparando el binario y los valores reales y llegó a la conclusión de que la representación de los valores reales es computacionalmente más rápido y obtener resultados más exactos.

Generación de la población inicial: Es necesario generar una población inicial para que el algoritmo genético puede ser utilizado y por lo tanto iniciar el procedimiento de optimización. El más utilizado es la generación aleatoria de individuos en la población con los valores dentro de los límites del espacio de búsqueda previamente definidos. Sin embargo, se sabe que es avanzar en una buena solución del problema, que puede ser incorporado en la población inicial o como parte de la misma[46].

Función objetivo: La función objetivo es responsable de evaluar el rendimiento de cada individuo de la población. Se pueden utilizar diferentes funciones objetivos en algoritmos genéticos, así como criterios de optimización como minimizar o maximizar, dadas las modificaciones necesarias.

Operador de Selección: tan pronto como se evalúan los individuos, pasan por un proceso de selección para formar una nueva población. Una selección basada en la probabilidad se lleva a cabo, en la que los individuos que tuvieron mayor rendimiento tienen más probabilidades de ser seleccionados. Varios métodos de selección:

- **Ruleta rusa, ranking lineal y geométrico:** estos métodos de selección implican una probabilidad de un individuo para ser seleccionado, basado en el valor de su rendimiento. El método de la ruleta rusa [43] es el más comúnmente utilizado y el método de selección se creó por primera vez en [89] (ver Figura 2.5).

- **Ranking:** Asocia una probabilidad a cada individuo de a cuerdo a su desempeño, se listan de mayor a menor.

- **Torneo:** este método no relaciona una probabilidad a los individuos. Se realiza una selección de un conjunto de individuos y aquellos con mejor desempeño son elegidos para formar parte de la nueva población [46].

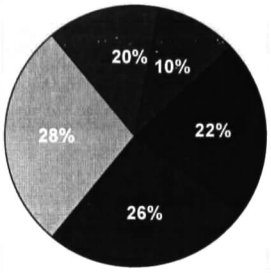


Figura 2.5. Ejemplo de Selección por Ruleta

Operadores genéticos: los operadores genéticos son los que garantizan el mecanismo básico de búsqueda del algoritmo genético. Se aplican a los individuos seleccionados para generar nuevas soluciones. Estos operadores son:

- **Mutación**: La mutación altera a un solo individuo, cambiando aleatoriamente un elemento de él para producir una nueva solución. En las versiones de la representación binaria de las soluciones que funciona mediante la inversión de los elementos del vector, y su probabilidad de ocurrencia es muy pequeño [44]. La mutación surge para restaurar la diversidad de las soluciones del algoritmo genético cuando converge prematuramente a un óptimo local (mínimo o máximo) [47].

- **Crossover**: este operador es el mayor énfasis en el algoritmo genético; tiene la función de combinar los segmentos de diferentes individuos para generar un nuevo individuo, y así una nueva solución [44] (ver Fig. 2.6)

- **Criterio de Parada:** El algoritmo genético se ejecutará generación tras generación hasta que un criterio de parada se cumpla. Algunos de los criterios de parada puede ser: detenerse al llegar a un número máximo de generaciones, cuando hay una convergencia de soluciones y ninguna mejora significativa en las soluciones de una generación a otra, cuando el algoritmo genético logra una solución considerablemente aceptable, etc. Estos y otros criterios también se pueden utilizar juntos.

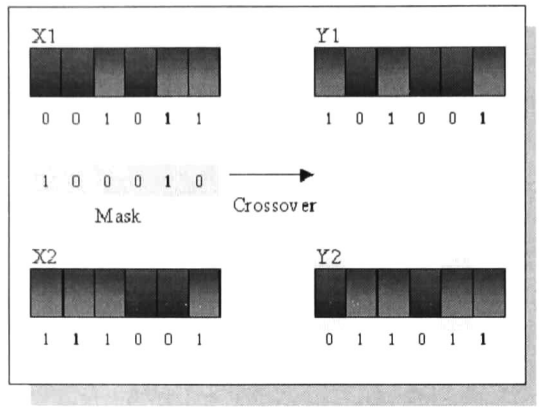


Fig. 2.6. Ejemplo de operación de cruza o crossover

Los algoritmos genéticos han sido una técnica de optimización robusta aplicada a problemas complicados de resolver. Algunas de las diferentes áreas en las que se aplicó con éxito son: reconocimiento de patrones, aplicaciones biológicas, el diseño de motores, entre otros [48][49][50].

Capítulo III

Propuesta de Tesis

En este trabajo de tesis se analiza el comportamiento del algoritmo de selección de atributos ARD cuando se aplica a datos cuyos atributos presentan correlación entre ellos. Primero se describen una serie de experimentos que demuestran que ARD falla al asignar un nivel de relevancia o *'ranking'* a elementos correlacionados. Para mejorar este resultado obtenido por ARD se aplica un esquema híbrido, consistente en la explotación de la correlación de los datos para inicializar la población de un algoritmo genético, el cuál es capaz de analizar y elegir el tamaño y la combinación óptima de atributos.

3.1 Modelo Híbrido para la Selección de Atributos

La selección de atributos tiene como objetivo la obtención de un subconjunto con los atributos más representativos del conjunto original de tal manera que el tiempo de procesamiento de los datos se reduzca, sin llegar a degradar el desempeño del clasificador. La idea es seleccionar las características más importantes de manera que definan un conjunto más pequeño que permita llevar a cabo un entrenamiento de cierto clasificador de una manera más rápida, usando solo los elementos realmente relevantes para ello. Para lograr su objetivo los algoritmos de selección de atributos deben optimizar dos puntos importantes: la longitud del subconjunto elegido y los elementos que contiene tal subconjunto.

El este trabajo analizaremos el caso de estudio del algoritmo de selección de atributos Automatic Relevance Determination [12], el cual permite definir una jerarquía de relevancia del conjunto original de los atributos al asignar a variables relevantes valores muy pequeños, mientras que las variables irrelevantes o redundantes obtienen valores grandes correspondientes a la varianza inversa de la evidencia obtenida por el método. Esta jerarquía no es completamente fiable, ya que como se explicará en breve, un solo modelo de ARD presenta fallas al asignar un nivel de relevancia individual para atributos correlacionados entre sí, es decir, elementos que contiene básicamente la misma información entre ellos pero que no pueden ser descartados por estar relacionados con la clase, a pesar de que no aportan nueva información para la clasificación. Por otro lado, el método ARD no encuentra el conjunto mínimo de rasgos, sólo provee la jerarquización.

Para ilustrar este comportamiento hemos reproducido una serie de experimentos realizados por [13][51], tal como se explica en el trabajo experimental.

3.1.1 Trabajo Experimental

a) Base de Datos Seno (2*π*x1) + ruido

En este experimento reprodujimos la base de datos sintética usada por [34] para demostrar el funcionamiento del algoritmo ARD en la determinación de la relevancia de las variables de entrada de un problema. Se creó una base de datos consistente en tres variables: *x1* es un valor aleatorio muestreado uniformemente en el rango (01,) al cuál se le agrega un bajo grado de ruido Gaussiano. La variable *x2* es una copia de *x1* pero con un mayor grado de ruido añadido, mientras que *x3* es elegido aleatoriamente de una distribución Gaussiana. El problema consiste en resolver la función dada por:

$$t = sin(2 * \pi * x1) \tag{3-1}$$

Por lo tanto podemos definir varios grados de relevancia para las variables de entrada: *x1* es muy importante, debido a que es indispensable para resolver el problema, *x2* tiene poca relevancia debido a que es una copia de *x1* con ruido añadido, y *x3* es totalmente irrelevante para la función objetivo.

La red neuronal usada fue un MLP con una capa de entrada con tres neuronas, una para cada variable, una capa oculta con tres neuronas y una capa de salida con una neurona, como se muestra en la Figura 3.1. La red es entrenada y se obtienen los valores de la varianza inversa para cada uno de los hiperparámetros asociados a los atributos de entrada. Como vemos en la Figura 3.2, el valor que le asigna ARD a la variable *x1* es muy pequeño, lo que corresponde a que sus pesos asociados son muy grandes e influyen en el entrenamiento de la red, al contrario de *x3* cuyo hiperparámetro obtiene un valor muy alto, correspondiente a un peso casi cero que no tiene relevancia para el funcionamiento de la red ni para resolver la función objetivo.

Figura 3.1. Diagrama de entradas y salidas de la red neuronal.

Atributo	*Valor ARD*	*Rank*
X1	0.19471	*1*
X2	34.77990	*2*
X3	428491.42098	*3*

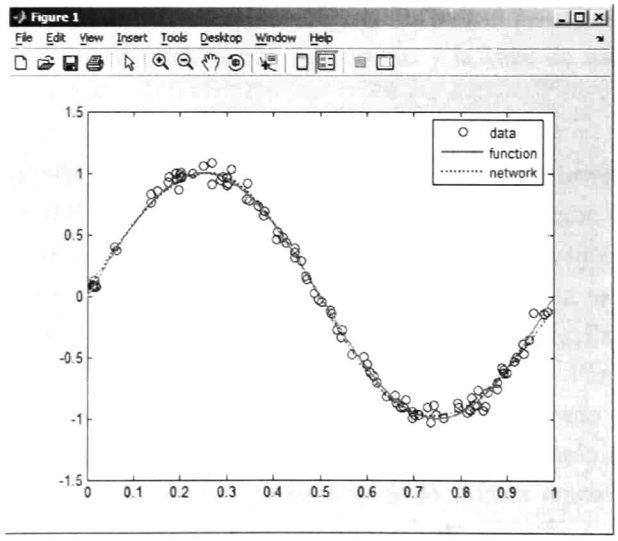


Figura 3.2. Resultado del ranking dado por ARD para la función Seno + Ruido

b) Base de Datos Sintética LIC1

LIC1 es una base de datos sintética de un problema bien formulado, usada por [52]. Esta base de datos contiene valores creados bajo la función:

$$LIC1 = \begin{cases} 1, & d\big((x1, y1), (x2, y2)\big) > length \\ 0, & otherwise \end{cases} \quad (3\text{-}2)$$

donde $d\big((x1, y1), (x2, y2)\big) > length$ es la distancia Euclidiana entre los puntos $(x1, y1)$ y $(x2, y2)$ de $\Re^2$, generados aleatoriamente. El problema es si la distancia calculada entre los dos puntos es o no mayor a un valor *length*, generado también aleatoriamente. Cada atributo de entrada contiene 1500 valores creados de manera independiente dentro del intervalo [0,1]. Las entradas poseen un rango de relevancia: el atributo *length* se encuentra en el primer grado de relevancia por ser crucial para la decisión del valor de la función, los atributos *y1* y *y2* son considerados en un segundo grado de importancia por determinar la distancia implícitamente mientras que *x1* y *x2* son considerados en un rango menor de relevancia sólo por ser los puntos iniciales, aunque son igual de imprescindibles que el resto de los atributos.

Para la implementación del experimento se usó el *Toolkit* llamado **NetLab** [53], el cual contiene un software disponible del algoritmo ARD configurado para aceptar la base de datos LIC1 con una red neuronal tipo MLP con la siguiente arquitectura: 6 neuronas de entrada (una por cada atributo), una sola capa oculta con 3 neuronas y la capa de salida conteniendo solo una neurona.

Para comprobar el funcionamiento del algoritmo ARD sobre atributos irrelevantes y/o correlacionados se hizo una modificación a la base de datos LIC1 al añadir una entrada extra llamada *in6*. Se probaron dos esquemas diferentes: para el primer esquema al atributo *in6* se le asignó un valor aleatorio no relacionado con el *length* ni con la salida del la red, o dicho en otras palabras, ruido. A esta variable irrelevante se le llamó "*dummy*" y se esperaba que sea identificada correctamente por ARD como ruido. Para el segundo esquema se le asigna a *in6* el valor de (*x1-x2*), que es un valor correlacionado con los atributos *x1, x2,* y se espera que tenga el mismo nivel de relevancia comparado con otros atributos, debido a que *x1, x2, y1* y *y2* fueron generados de la misma manera e independientes entre ellos.

La red fue entrenada 2 veces por 300 épocas usando el algoritmo de aprendizaje *Scale Conjugate Gradient 'SCG'* [58]. Después de la ejecución del algoritmo para el

primer esquema en que *in6* es dummy, se obtuvieron los siguientes resultados mostrados en la Figura 3.3, donde la posición 1 del ranking es la más relevante:

Atributo	Valor ARD	ranking
X1	0. 0417	2
X2	0.0463	3
Y1	1.5246	4
Y2	1.5417	5
length	0.0440	1
In6	36.674	6

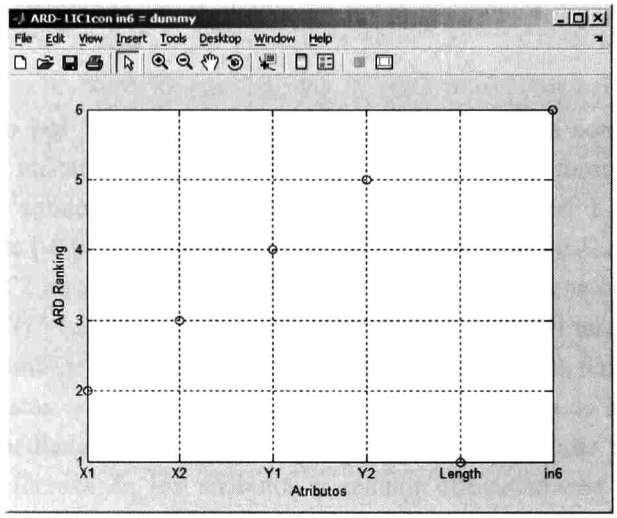


Figura 3.3. Resultado del ranking dado por ARD para LIC1 con *in6* = dummy

ARD fue capaz de distinguir a *in6* como irrelevante, asignándole la posición 6 perteneciente a la prioridad más baja.

Para el segundo esquema donde *in6* toma el valor de (*x1-x2*). Los resultados demuestran que ARD interpreta a *in6* como más importante que *x1* y *x2*, pero menos importante que *y1* y *y2* (ver Figura 3.4), contrario a la asignación de prioridad esperada.

Atributo	Valor ARD	Rank
X1	0.84859	6
X2	0.59708	5
Y1	0.16723	4
Y2	0.14636	3
length	0.06303	1
In6	0.09013	2

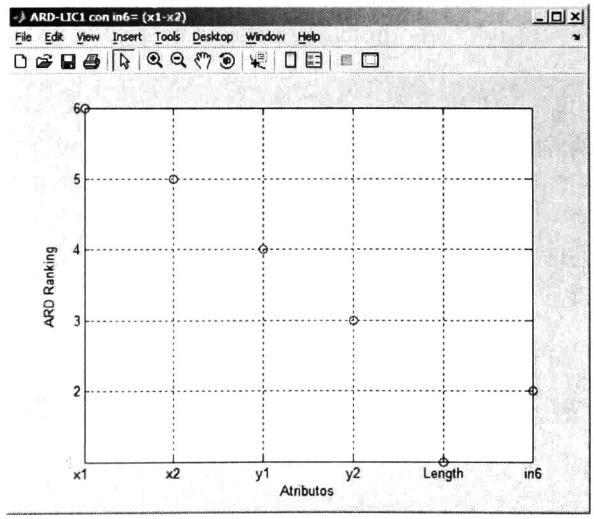


Figura 3.4. Resultado del ranking dado por ARD para LIC1 con *in6* = (*x1-x2*).

c) Base de Datos Sintética LIC1 con ensamble de redes neuronales

En este experimento se usó la función creadora de la base de datos LIC1 definida en el experimento anterior, pero implementando un ensamble de redes neuronales con las siguientes características:

El ensamble está conformado por 3 redes neuronales tipo MLP, cada una con 6 neuronas de entrada (una por cada atributo), un número variable entre 1 y 19 neuronas para la capa oculta y la capa de salida conteniendo solo una neurona. La red 1 es inicializada con pesos en el rango de [-0.01, 0.01] y usa el algoritmo de aprendizaje *Scale Conjugate Gradient 'SCG'* . La red 2 inicia sus pesos entre [-0.001, 0.001] y entrena con el algoritmo *Conjugate Gradient 'CG'*. La tercera red inicia sus pesos en dentro del rango de [-0.0001, 0.0001] y usa el algoritmo de aprendizaje *Quasi Newton 'QN'*. Las tres redes son entrenadas 2 veces por 300 épocas, y el valor resultante del hiperparámetro α de los atributos de las tres redes es promediado. El uso de un ensamble de redes permite un realizar mejor el análisis de la evidencia de los atributos y reducir eficientemente la incertidumbre, como se demuestra en [36].

El resultado obtenido en este experimento es similar al obtenido en el experimento anterior, a diferencia que esta vez ARD asigna *y1* y a *y2* mayor relevancia y relega a in6 al cuarto lugar, antes que *x1* y *x2* (ver Figura 3.5). Una posible explicación es que al contener *in6* la información condensada de los atributos *x1* y *x2*, ARD le asigna a un nivel superior de importancia a pesar de que pertenece al mismo nivel que los demás atributos, lo que nos sugiere que el algoritmo falla al asignar una prioridad a atributos correlacionados que poseen un mismo nivel de relevancia, es decir, titubea y no tiene una clara decisión acerca de cuál debe ser la posición en la categorización.

Atributo	Promedio ARD	Rank
X1	1.0517	5
X2	1.2446	6
Y1	0.2970	3
Y2	0.2732	2
length	0.1404	1
In6	0.4711	4

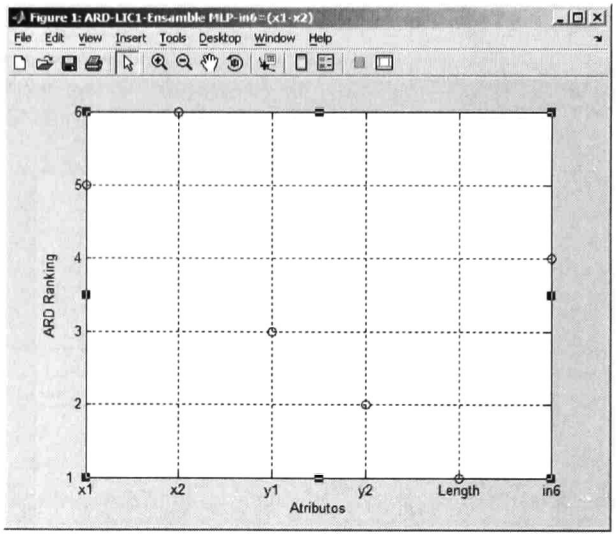


Figura 3.5. Resultado del ranking dado por ARD con ensamble de MLP para la base de datos LIC1, con *in6* = (*x1-x2*).

d) Base de Datos Sintética LIC4

La base de datos LIC4 es un problema más complejo que su predecesora LIC1. En LIC4 se calcula el área de un triángulo basado en 3 puntos (x, y) que son valores reales entre [0,1] y que representan puntos en un plano coordenado. Si Δ es el área del triángulo definido por los puntos *(x1,y1), (x2,y2) y (x3,y3)*, entonces:

$$LIC4 = \begin{cases} 1, & si\ \Delta > AREA \\ 0, & otherwise \end{cases} \quad (3\text{-}3)$$

En este experimento se usó *in8=(x1*y2)*, que representa el valor implícito de la distancia de un lado del triángulo y se espera que sea de igual o menor importancia que los demás puntos. Se usó también un MLP con la misma configuración descrita en el experimento 1 con la base de datos LIC1. Los resultados obtenidos sugieren que *in8* es más relevante que su elemento correlacionado *y2,* pero menos sobresaliente que *x1.* Además, su posición en la categorización sugiere que el nuevo elemento es incluso más importante que *y2* y *y3,* que son puntos que delimitan la distancia entre puntos y que no poseen correlación con él, lo cual es incorrecto.

Atributo	Promedio ARD	Rank
X1	1.63972	3
X2	24.33497	8
X3	2.72959	6
Y1	1.57579	2
Y2	12.52267	7
Y3	2.68503	5
Área	0.05856	1
In8	1.93013	4

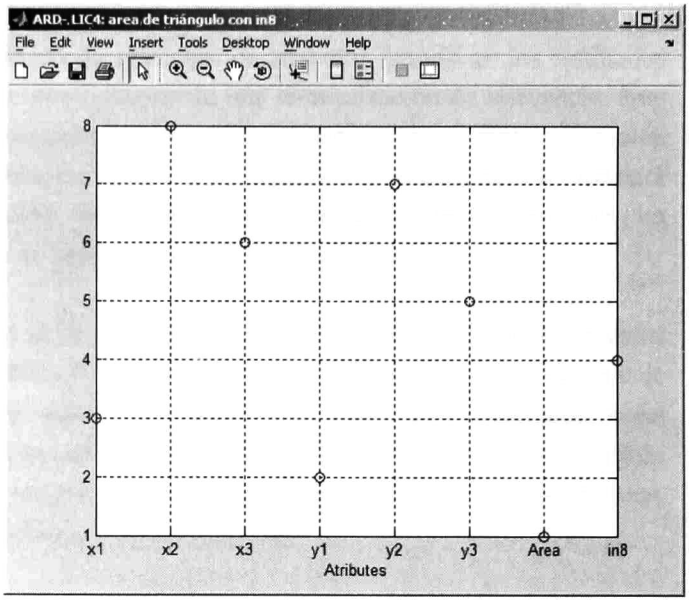


Figura 3.6. Resultado del ranking dado por ARD con un único MLP para la base de datos LIC4, con in8 = (x1*y2)

Como resultado del trabajo experimental se pudo comprobar que el algoritmo ARD falla al asignar un nivel de relevancia cuando existe correlación entre los atributos asignando niveles de prioridad cambiantes, lo que hace poco fiable el resultado obtenido. Pensamos corregir este aspecto al proponer un esquema híbrido utilizando computación evolutiva, debido a que es necesario buscar dentro de todo el espacio de atributos por la mejor combinación que nos permita elegir un subconjunto de un tamaño mínimo y libre de elementos irrelevantes.

3.2 Esquema híbrido ARD + Algoritmos Genéticos

Se presenta un esquema híbrido de selección de atributos basado en los resultados del algoritmo ARD como primera aproximación de una jerarquización de relevancia. Este esquema debe explotar la correlación entre los atributos que ARD no evalúa adecuadamente para crear una población inicial de un algoritmo genético, el cuál efectuará una búsqueda por la combinación de atributos que mejore el rendimiento de un clasificador lineal, cuyo resultado es usado como función de evaluación.

Para describir la propuesta se ha usado como caso de estudio la base de datos pública llamada **Waveform Database Generator (Version 2) Data Set** [54]. Esta base de datos es usada por que ofrece datos correlacionados. Los datos obtenidos están compilados en el archivo *Waveform+noise*, que contiene 40 atributos con ruido agregado que describen 3 clases de ondas (*wavelets*) formadas por la **combinación** de 3 ondas base, lo que le da un potencial de correlación, y 5000 instancias.

Se ejecutó el algoritmo ARD en la base de datos Waveform+noise para obtener una jerarquización de relevancia de los atributos de la base de datos. Este ranking o lista de relevancia no es considerada totalmente fiable debido a que los atributos derivados de la combinación de las ondas base están potencialmente correlacionados [19], y como vimos en el trabajo experimental ARD puede fallar en la asignación de niveles de relevancia. Para conocer cuáles atributos pudieran estar mal posicionados en el ranking se propone el uso de un algoritmo genético AG similar al de [55]. Este algoritmo genético usa el *Genetic Algorithm and Direct Search Toolbox* contenido en el software **MatLab** [56], y está especialmente adaptado para hacer selección de atributos sobre la base de datos Waveform+noise.

3.2.1 Esquema Algoritmo Genético-RANDOM

El uso de este algoritmo está enfocado en obtener una segunda opinión acerca de cuáles deberían ser los atributos más relevantes. La ejecución del algoritmo genético para selección de atributos inicia con la selección de un conjunto o '*pool*' de individuos que constituyen la población inicial. Esta población es creada de manera aleatoria con una distribución uniforme para los 40 atributos de la base de datos. Cada individuo es evaluado por una función de *fitness*, consistente en el resultado de la aplicación de un clasificador tipo *Linear Discriminant Analysis (LDA)* [59]. Los mejores individuos son seleccionados para la siguiente generación, mientras que los demás son sometidos a operaciones de cruza y mutación.

El algoritmo es inicializado con una población aleatoria con distribución uniforme de tamaño n definido por el usuario, que contiene 100 individuos. Se utilizaron las funciones de cruza y mutación de [55], las cuales contienen una validación que evita que no existan elementos repetidos en los subconjuntos que dan como salida. El algoritmo se detiene al cumplir 10000 unidades de tiempo o bien al llegar a producir 100 generaciones.

Esta configuración del algoritmo genético fue llamada "RANDOM" debido a que su inicialización fue simplemente aleatoria, es decir, no usa ni explota la correlación entre los atributos, lo cual es un procedimiento ineficiente [19].

3.2.2 Esquema ARD+AG

El algoritmo ARD no es capaz por sí solo de hacer una buena evaluación de la relevancia de los atributos cuando presentan correlación, por lo cual se propone el uso de un algoritmo genético que sea capaz de determinar la importancia de los atributos explotando la correlación puede refinar el resultado de ARD. Para usar toda la capacidad de búsqueda que ofrecen los algoritmos genéticos es necesario que los datos estén agrupados por sus correlaciones, de tal manera que el espacio de búsqueda se concentre en los atributos que pueden ser descartados como irrelevantes.

Para tener una idea de cuáles atributos pudieran ser afectados por la correlación se realizó un análisis entre la jerarquización que se obtuvo del algoritmo ARD y el resultado de 15 iteraciones de un algoritmo genético con esquema RANDOM. Los subconjuntos de atributos elegidos como resultado de cada iteración fueron usados para realizar la gráfica de frecuencias que se presenta en la Figura 3.7. En esta gráfica se aprecia que la frecuencia de aparición de atributos se marca para los primeros 20 atributos, y después de manera intermitente para los atributos de 25 al 40.

Una vez obtenido este muestreo fue comparado con la salida de ARD para encontrar elementos en común entre las dos soluciones. Como resultado se obtuvo que los primeros 10 atributos aparecen en casi todas las soluciones dadas por el AG, al igual que un bloque de atributos en la parte media del vector, pero con discrepancias entre los dos resultados. A partir del atributo 30 las coincidencias comienzan a ser esporádicas y poco marcadas, lo que indica que son atributos poco relevantes o bien que pueden estar correlacionados con otros pero que son significativos para la predicción. Con esta comparativa pudimos establecer que existen *bloques* de atributos que presentan mayor frecuencia de aparición en los subconjuntos finales, pero otros bloques son aun indefinidos.

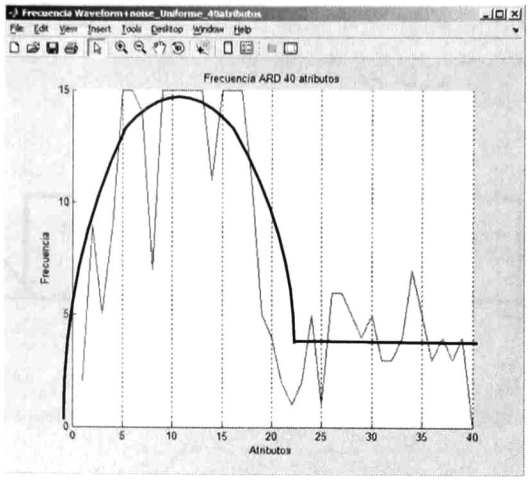


Figura 3.7. Frecuencia de los atributos de Waveform+noise.

El resultado del análisis fue usado para proponer un nuevo esquema de selección de atributos que tome en cuenta el ranking obtenido de ARD como punto de partida para inicializar un algoritmo genético. Para esto construimos un método de inicialización para el algoritmo genético basado en dos fases:

a) Selección de atributos de ARD con distribución Poisson-Uniforme

Los bloques resultantes del análisis anterior muestran que sólo tenemos información certera de que los primeros 10 atributos son realmente relevantes, ya que aparecen en todas las soluciones tanto de ARD como del AG, por lo que deben aparecer dentro del subconjunto final. A estos atributos se les asigna una probabilidad tipo *Poisson* de ser elegidos para formar parte del pool. Creemos que esta distribución de probabilidad modela aceptablemente la presencia de atributos como se muestra en la Figura 3.7. Note que la concentración de los atributos más relevantes se localiza más a la izquierda. Los demás atributos presentan pocas coincidencias de aparición, por lo tanto no podemos estar seguros de su relevancia. A este bloque se le asigna una distribución uniforme, como se muestra en la Figura 3.7 y en la Figura 3.8.

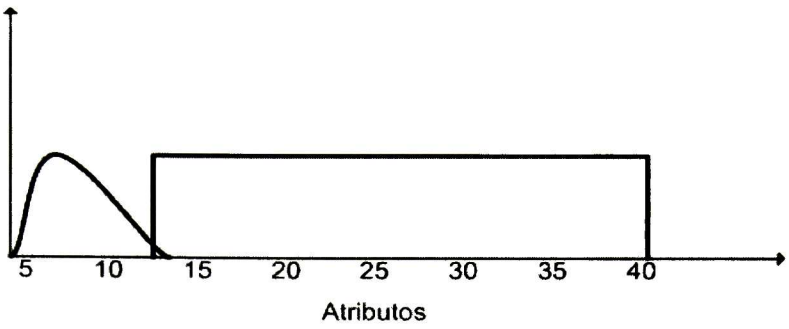


Figura 3.8. Esquema Poisson-Uniforme para selección de atributos de ARD

Esta combinación de distribuciones nos da un criterio de selección de los atributos listados por ARD, sin embargo, como se menciona anteriormente, es necesario tomar en cuenta la correlación entre los atributos para mejorar el rendimiento del algoritmo genético. Para lograr esto se añade una segunda fase al esquema.

b) Matriz de Correlación entre atributos.

La agrupación de atributos correlacionados permite al algoritmo genético distinguir cuáles de esos atributos son realmente relevantes. Para realizar esta agrupación se propone calcular la distancia existente entre los atributos. De [19] se obtiene que los atributos altamente correlacionados están cerca en distancia. Esta distancia está dada por:

$$dist(F_\alpha F_\beta) = \frac{1}{N} - \sum_i \left(\frac{F_\alpha(i) - mean(F_\alpha)}{std(F_\alpha)}\right) \cdot \left(\frac{F_\beta(i) - mean(F_\beta)}{std(F_\beta)}\right) \qquad (3\text{-}4)$$

El resultado es una matriz de 40x40 conteniendo la distancia (correlación) de cada atributo con los otros 39. A esta matriz la llamaremos *MC* o *matriz de correlación.*

El esquema propuesto de selección de atributos combina estas dos fases como se muestra en la Figura 3.9. Se inicia obteniendo el ranking dado por ARD, posteriormente se calcula un vector de posiciones usando la distribución Poisson-Uniforme, que nos servirá para seleccionar los elementos contenidos en el vector de resultados de ARD. Una vez elegido el elemento indicado por el índice, nos dirigimos a la matriz de correlación, con el objetivo de seleccionar alguno de los atributos que presenten más correlación con el elemento elegido. Para dar oportunidad a varias combinaciones diferentes, se calcula una

columna aleatoria dentro de las primeras 10 de la matriz, debido a que están ordenados por los valores más altos. Una vez que obtenemos un elemento de la matriz, lo agregamos como parte del nuevo individuo de la población inicial que estemos creando. Este proceso se repite hasta completar la longitud del individuo que consiste en *n* valores, la cual corresponde al tamaño del subconjunto que deseamos obtener como solución.

El objetivo de éste método es crear un conjunto de individuos correlacionados a los atributos elegidos por ARD como relevantes para reducir el espacio de búsqueda del algoritmo genético y sólo determinar cuáles elementos correlacionados son en verdad relevantes.

La matriz de Población Inicial contiene 100 vectores, donde cada vector es llamado un '*individuo*' y está formado por un número *n* de elementos. Cada individuo es validado para evitar contener elementos repetidos. Esta población sirve como entrada al algoritmo genético, que se encargará de seleccionar los individuos que obtengan mejor rendimiento del algoritmo clasificador y finalmente determinará la longitud necesaria del conjunto mínimo.

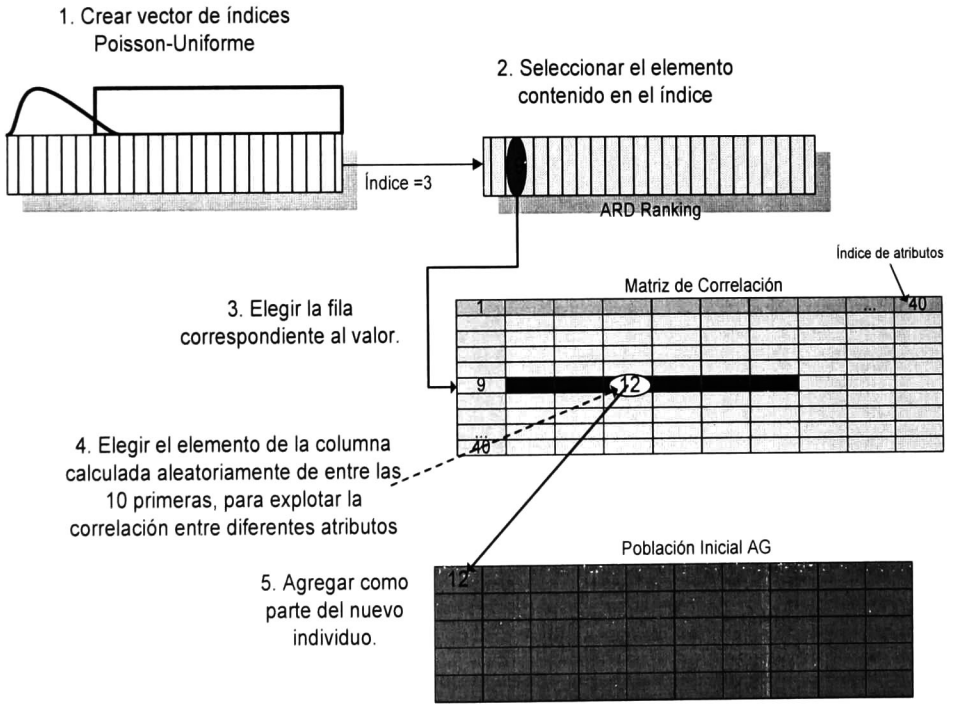


Figura 3.9. Esquema propuesto basado en la distribución Poisson-Uniforme y la Matriz de Correlación MC

Capítulo IV

Análisis de Resultados

Este capítulo presenta el análisis experimental que nos servirá para comprobar el rendimiento de la propuesta basado en el esquema ARD+AG, en comparación de un algoritmo genético con esquema RANDOM. Se describen los objetivos y se realiza una comparación entre resultados.

4.1 Trabajo Experimental

El trabajo experimental aquí presentado tiene como objetivo probar el funcionamiento del método propuesto para corregir el resultado obtenido del algoritmo ARD, así como compararlo con el rendimiento de un algoritmo genético que no explota la correlación en los datos. El caso de estudio para todos los experimentos realizados fue la muy conocida base de datos **Waveform+noise** [54] y el algoritmo genético con la configuración descrita en el capítulo III.

4.2 Experimentos con Esquema RANDOM

Para el primer experimento se usó el algoritmo genético con esquema RANDOM que, como se mencionó anteriormente, crea una población inicial aleatoria con una distribución uniforme para los 40 atributos de la base de datos. Este esquema explora todo el espacio de búsqueda posible, es decir de entre todas las posibles combinaciones de

atributos. Su objetivo es encontrar un subconjunto que maximice el rendimiento del clasificador que sirve como función de evaluación. Como se desconoce el tamaño del subconjunto a elegir, se probó con individuos de largo variable n=16,17,18...30. Dado que el algoritmo genético es capaz de encontrar diferentes soluciones sub-óptimas, se realizaron 15 ejecuciones para cada tamaño n, con la finalidad de conocer los atributos más comúnmente elegidos para formar parte del subconjunto final.

Los resultados promedio de cada set de 15 iteraciones para cada tamaño n se muestran en la Figura 4.1 y su comportamiento y algunas estadísticas para este esquema se muestran en la Figura 4.2.

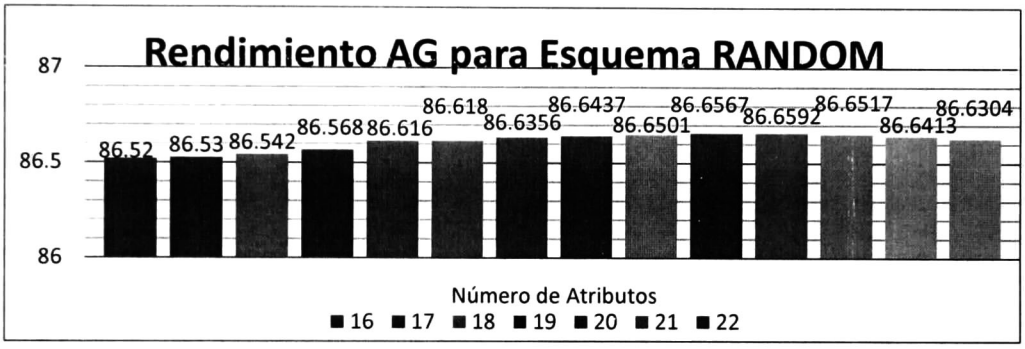


Figura 4.1. Rendimiento del AG para cada set de 15 iteraciones del AG con el esquema RANDOM

# n	RANDOM	
Atributos	Media %	Desv. Est.
16	86.52	0.0513
17	86.53	0.06532
18	86.5427	0.061814
19	86.568	0.072427
20	86.616	0.0519
21	86.618	0.0754
22	86.6356	0.0863
23	86.6437	0.0853
24	86.6501	0.0873
25	86.6567	0.0949
26	86.6592	0.1005
27	86.6517	0.1056
28	86.6413	0.1095
29	86.6304	0.112
30	86.6202	0.126

Figura 4.2. Resultados del rendimiento del esquema RANDOM para diversos tamaños n de individuo.

Para este esquema el mejor rendimiento del clasificador se obtiene cuando el tamaño *n* del subconjunto es de 26 atributos.

4.3 Experimentos con esquema ARD+AG

En este experimento se desea probar el resultado del algoritmo genético cuando su población inicial es creada con nuestra propuesta. El método propuesto usa ARD como primera aproximación de relevancia y un algoritmo genético especialmente modificado para explotar la correlación entre los atributos.

Se inicia con la ejecución del algoritmo ARD sobre la base de datos Waveform+noise, obteniendo el ranking que se muestra en la Tabla 4-1 y ordenándolo ascendentemente por número de atributo. Posteriormente se calcula el vector de índices usando la combinación de distribuciones Poisson-Uniforme y la Matriz de Correlación usando la fórmula de distancia (3-4).

Posición	Atributo	Valor α		Posición	Atributo	Valor α
1	9	2.49963		**21**	10	14.81739
2	15	2.90702		**22**	34	16.41111
3	16	2.92164		**23**	38	17.68594
4	17	2.94226		**24**	37	18.23409
5	5	3.11441		**25**	19	22.93719
6	22	4.93484		**26**	25	23.65007
7	11	5.19482		**27**	26	24.24982
8	6	5.52838		**28**	35	26.07207
9	7	5.59979		**29**	32	27.32954
10	28	5.80971		**30**	31	34.66444
11	8	7.07109		**31**	40	35.12972
12	13	7.15838		**32**	29	38.41897
13	12	9.25104		**33**	36	49.65134
14	14	9.75337		**34**	2	51.70938
15	4	10.20448		**35**	27	52.57761
16	3	10.28092		**36**	21	53.27015
17	18	11.50778		**37**	23	58.64654
18	1	11.50893		**38**	33	64.45299
19	39	11.65315		**39**	24	79.96368
20	20	14.45657		**40**	30	128.6699

Tabla 4-1. Ranking o jerarquía de relevancia obtenida por ARD para la base de datos Waveform+noise.

Para crear un vector de la población inicial del algoritmo genético, se toma del ranking ordenado de ARD un elemento de la posición indicada por el vector de índices, posteriormente usando ese elemento se obtiene de la matriz de correlación uno de los

atributos altamente correlacionado a él, el cual formará parte del vector del individuo, ver Figura 3.9

En este experimento se ejecutó el algoritmo genético con la población inicial creada basado enl método ARD+GA con 15 iteraciones para cada tamaño de individuo *n*, donde n=16,17,18...30. Los resultados promedio de cada set de 15 iteraciones para cada tamaño *n* se muestran en la Figura 4.3 y su comportamiento y algunas estadísticas para este esquema se muestran en la Figura 4.4.

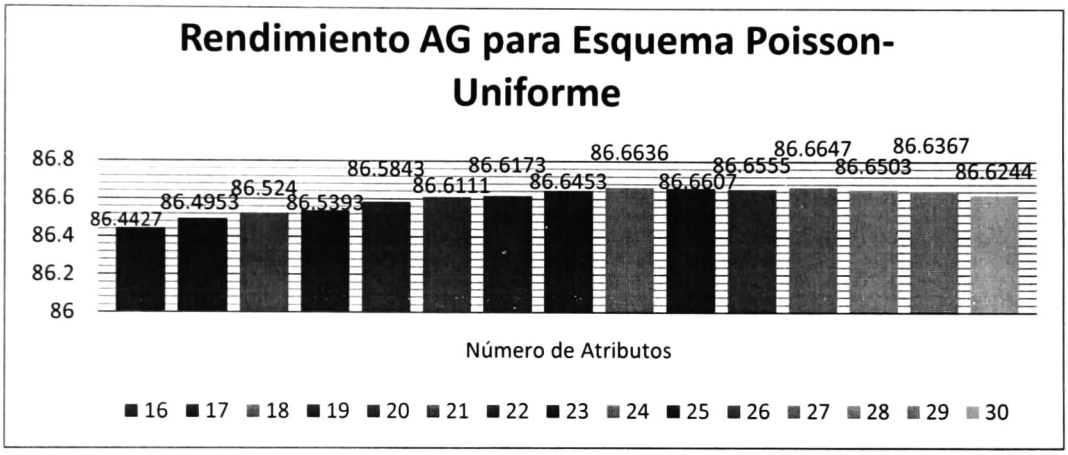


Figura 4.3. Rendimiento del AG para cada set de 15 iteraciones del AG con el esquema ARD+AG

# *n* Atributos	Poisson-Uniform	
	Media %	Desv. Est.
16	86.4427	0.0688
17	86.4953	0.0932
18	86.524	0.1274
19	86.5393	0.1058
20	86.5843	0.11
21	86.6111	0.1114
22	86.6173	0.0881
23	86.6453	0.0964
24	86.6636	0.1085
25	86.6607	0.1105
26	86.6555	0.1124
27	86.6647	0.1107
28	86.6503	0.1223
29	86.6367	0.1255
30	86.6244	0.1332

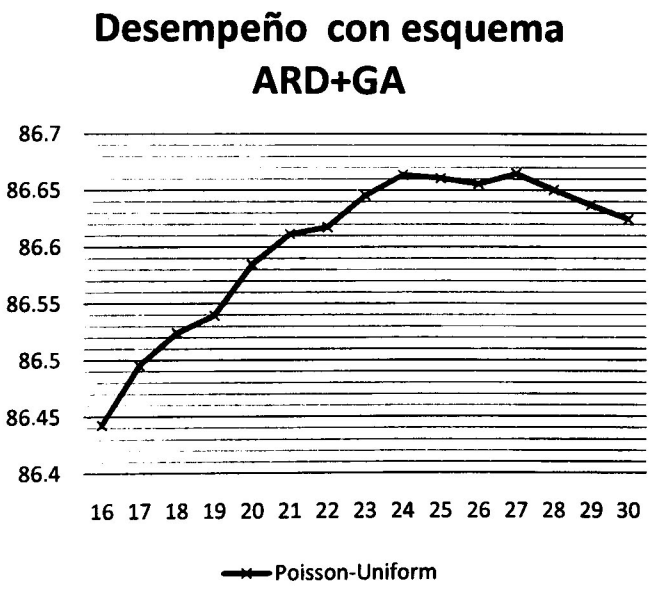


Figura 4.4. Resultados del rendimiento del esquema ARD+GA para diversos tamaños *n* de individuo.

Para este esquema el mejor rendimiento del clasificador se obtiene cuando el tamaño n del subconjunto es de 27 atributos. En la Figura 4.5 se puede ver la comparación entre los resultados de los experimentos anteriores, donde sobresale el rendimiento de nuestra propuesta.

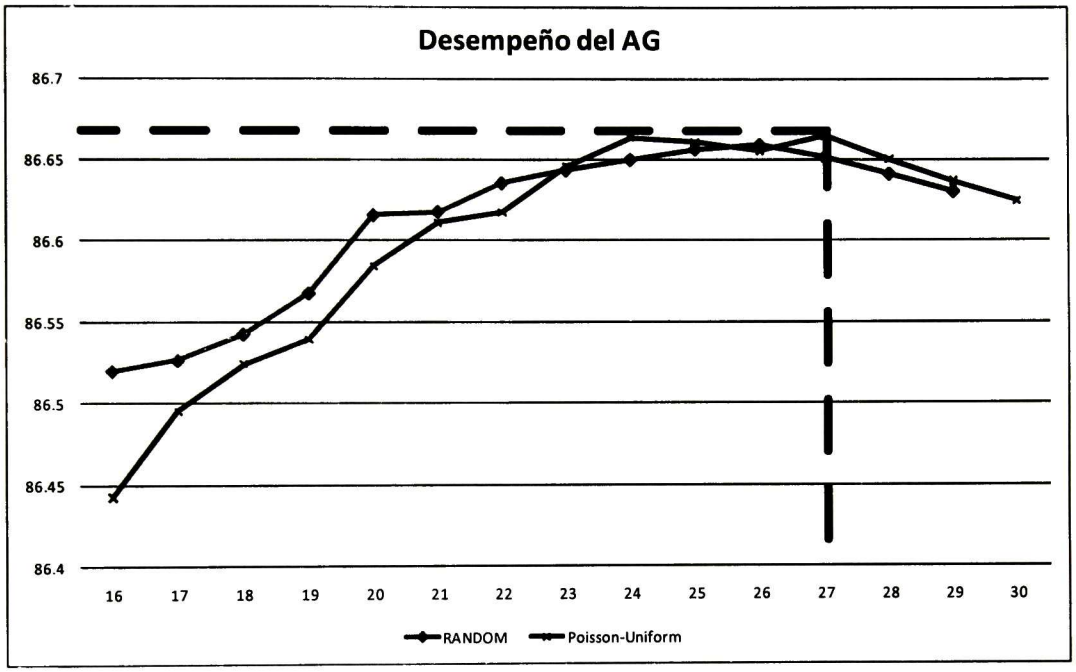


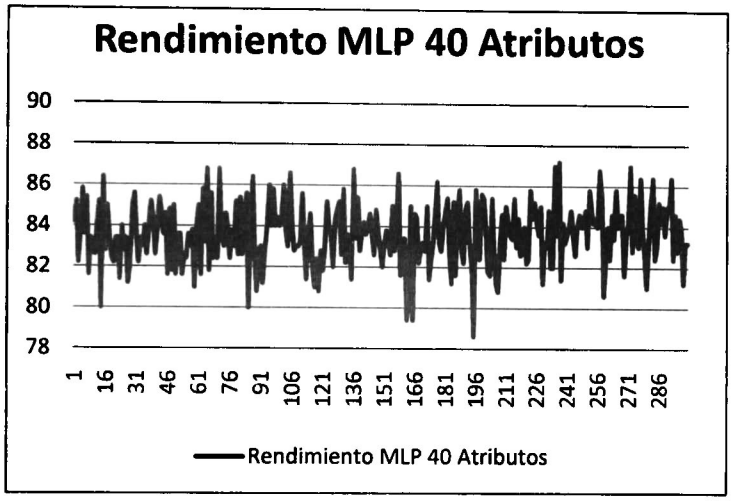
Figura 4.5. Resultados del rendimiento del esquema ARD+GA para diversos tamaños n de individuo.

En los resultados de los experimentos anteriores vemos que el algoritmo genético con esquema RANDOM alcanzó su mejor rendimiento cuando el tamaño del subconjunto era de 26 atributos, con un 86.6592%; mientras que el algoritmo genético con esquema Poisson-Uniforme+ MC obtuvo un 86.6647% con n=27 atributos. Con estos datos, podemos elegir a 27 como el tamaño óptimo del subconjunto L_q. Estos resultados sugieren que el algoritmo genético que no explota la correlación se ve afectado en su rendimiento y en su capacidad para elegir el tamaño óptimo de subconjunto.

Siguiendo esta idea se realizaron pruebas con un clasificador de tipo *MultiLayer Perceptron (MLP)* para evaluar el ranking asignado por ARD y el subconjunto elegido por el algoritmo genético con el esquema Poisson-Uniforme+MC. Para la realización de estas pruebas se utilizó el software **WEKA** (*Waikato Environment for Knowledge Analysis*) [57], con el cual se ejecutaron 30 iteraciones del clasificador MultilayerPerceptron con *ten fold cross validation*. La configuración de la red fue la usada por default por WEKA, la

cual consiste en una capa de entrada con una neurona para cada atributo de la base de datos, una capa escondida con a neuronas, donde $a = (atributos + numeroclases) / 2$ y una capa de salida que contiene una neurona para cada una de las 3 clases. La red aprende con un *LearningRate* de 0.3 y un *momentum* de 0.2 durante 500 épocas.

En la primera prueba se usó el clasificador MLP en el conjunto completo de 40 atributos de la base de datos, dado que inicialmente no se conoce el tamaño óptimo del subconjunto ni cuáles son los elementos que deben ser elegidos como más relevantes. Los resultados se muestran en la Figura 4.6.



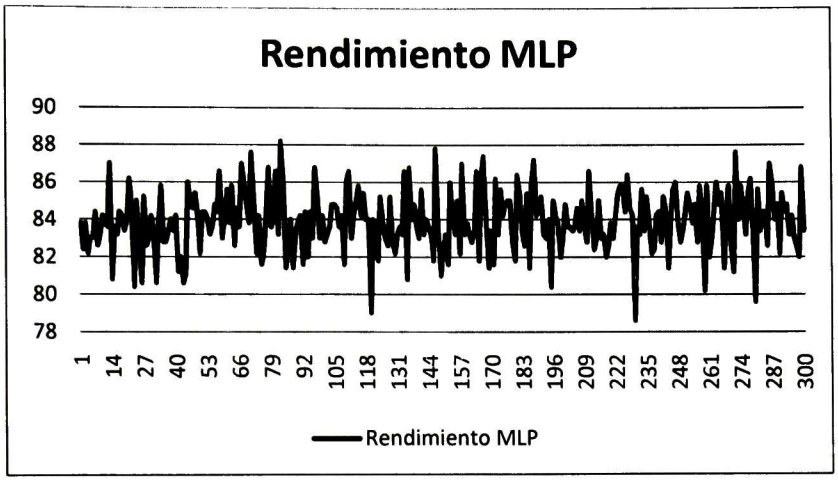
Promedio	83.623
Desv. Est.	1.510

Figura 4.6. Resultado de 30 iteraciones de MLP para los 40 atributos de la base de datos Waveform+noise

Para la segunda prueba se usó el clasificador MLP con la misma configuración que la prueba anterior, esta vez para los primeros n elementos del ranking dado por ARD (ver Figura 4.7), donde n=27 debido a que fue el tamaño óptimo encontrado por nuestro método gracias al algoritmo genético. Los resultados se muestran en la Figura 4.8.

1	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22	25	26	28	34	37	38	39

Figura 4.7. Subconjunto de los primero 27 atributos del ranking de relevancia de ARD.



Promedio	83.9107%
Desv. Est.	1.607

Figura 4.8. Resultado de 30 iteraciones de MLP para los primeros 27 atributos del ranking dado por ARD.

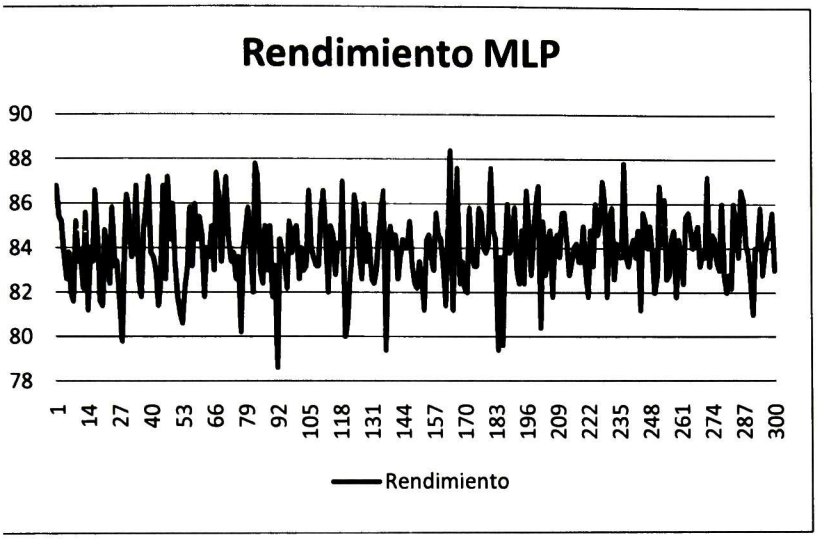
Los resultados de las 30 iteraciones del MLP dieron un promedio de 83.9107% de clasificaciones correctas, lo que indica que la reducción propuesta por el ranking de ARD da una aproximación a un subconjunto que indica cuáles pueden ser los elementos más relevantes.

Debido a que nuestros experimentos demuestran que la correlación de la base de datos afecta el desempeño de ARD, se usó para la tercera prueba el subconjunto de atributos de tamaño n = 27 que presentó el mayor rendimiento con el esquema propuesto ARD+GA, el cual se muestra en la Figura 4.9.

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	26	27	28	32	34	35	37	39

Figura 4.9. Subconjunto de atributos que presentaron mejor rendimiento usando el algoritmo genético, con tamaño n=27.

Como se puede ver en la Figura 4.10, el rendimiento promedio de todos los 10 *folds* por cada una de las 30 iteraciones fue de 83.97% de éxito del clasificador, lo cual indica que algunos elementos que ARD había elegido como relevantes fueron cambiados por el algoritmo genético, encontrando un subconjunto que explota la correlación de los atributos y aumenta el desempeño del clasificador. Este aspecto de correlación del método híbrido es lo más destacado de su funcionamiento.



Promedio	83.97%
Desv. Est.	1.677

Figura 4.10. Resultado del MLP para el mejor subconjunto de Poisson-Uniforme+MC con n=27.

Los resultados de las pruebas demuestran que el rendimiento del subconjunto elegido por nuestra propuesta es más alto que el rendimiento de los atributos que eligió ARD como los más relevantes, lo que demuestra que ARD falla al asignar un nivel de relevancia y que, por lo tanto, nuestra propuesta ayuda tanto a encontrar el tamaño mínimo del subconjunto y los atributos que son realmente relevantes para la predicción de las clases, además de que explota la correlación entre los datos y ayuda a refinar el resultado propuesto por ARD.

Capítulo V

Conclusiones y Trabajo Futuro

5.1 Conclusiones

En este trabajo de tesis se realizó un estudio del comportamiento del algoritmo de selección de atributos Automatic Relevance Determination (ARD) basado en su capacidad para asignar un nivel de relevancia a elementos correlacionados. Se demostró por medio de un análisis experimental que tal algoritmo no es capaz de definir una jerarquía individual para elementos que pertenecen al mismo nivel de relevancia, es decir, cuando los atributos presentan correlación entre ellos; además de no ser capaz de proveer la longitud mínima del conjunto de atributos imprescindibles.

El objetivo de este trabajo fue demostrar que la combinación de métodos basados en modelos probabilísticos usando inferencia Bayesiana y algoritmos genéticos, por el principio fundamental de sinergia y cooperación se debía obtener un método mejor que subsanara las deficiencias que presentan los métodos usados individualmente.

El algoritmo ARD, basado en teoría Bayesiana y entrenamiento hacia atrás (*backpropagation*), puede hacer una categorización de la relevancia de los atributos. Nuestra experiencia demuestra que esta categorización presenta cierta ambigüedad, además de que ARD no es capaz de indicar el subconjunto mínimo. Por otro lado, empleando un método de computación evolutiva se puede encontrar el tamaño mínimo para un buen resultado del clasificador. En ambos métodos no se considera la correlación entre los atributos, que es el factor determinante en la correcta evaluación de cada rasgo.

Por esta razón es necesario establecer un método que ayude a obtener un subconjunto mínimo conteniendo únicamente los atributos más relevantes, explotando la correlación existente en los datos.

En este sentido se propuso un método que usa la categorización dada por ARD y una aproximación de los elementos más correlacionados para crear un *pool* de individuos que conformen la población inicial del algoritmo genético. Este algoritmo genético encontró el conjunto mínimo usando los mejores rasgos sugeridos por ARD. Cada individuo en el pool fué formado en dos fases: primero se toma de ARD un conjunto de elementos elegidos basado en una combinación de distribuciones: los primeros 10 elementos del vector de jerarquías tienen más probabilidad de ser parte de la solución, por lo que se les asignó una distribución de tipo *Poisson*. Los elementos restantes son igualmente probables, por lo que se les asignó una distribución de tipo *Uniforme*, lo que aumenta la diversidad de los individuos. Como paso siguiente se tomaron los elementos más correlacionados a cada elemento obtenido de estas distribuciones, lo que le da capacidad al algoritmo genético de analizar las combinaciones de atributos que aportan información relevante y ayudó a descartar las que sólo repiten información.

El resultado de nuestra propuesta es que hace un gran énfasis en la correlación entre atributos para lograr que por un lado el algoritmo genético obtenga la longitud del conjunto mínimo descartando los elementos que habían sido mal posicionados en la categorización de ARD, asegurando el buen rendimiento del algoritmo clasificador.

Esta tesis presenta una alternativa nueva no antes reportada en ninguna publicación revisada, usando como caso de estudio una base de datos de Wavelets aplicada para usar rasgos correlacionados. Finalmente, el lector y usuario puede explotar las ventajas de nuestro método para la extracción de atributos de diversas bases de datos, para las cuales el solo uso de ARD o únicamente un algoritmo genético no basta, como demuestra nuestro análisis experimental.

5.2 Trabajo Futuro

De acuerdo a los resultados de este trabajo, el estudio puede continuar hacia el desarrollo de un método que nos defina un mejor tratamiento de la evidencia otorgada por ARD para determinar la relevancia individual y grupal de cada atributo, de tal manera que se aumente el poder predictivo de ARD. Por otro lado, poder postular un procedimiento más eficiente del algoritmo genético que haciendo uso de código paralelizado pueda tomar en cuenta variaciones en el tamaño del subconjunto de atributos.

Referencias

[1] TomMitchell. ***Machine Learning***. McGraw Hill, 1997

[2] M.Dash, H.Liu. ***Feature Selection for Classification***. *Intelligent Data Analysis* 1, 1997

[3] Luis Carlos Molina, Lluís Belanche, Ángela Nebot. ***Feature Selection Algorithms: A Survey and Experimental Evaluation***. *ICDM 2002*: 306-313

[4] Huan Liu, Hiroshi Motoda as Guest Editors' Introduction: ***Feature Transformation and Subset Selection***. *IEEE Intelligent Systems* 13(2): 26-28 (1998)

[5] Arauzo-Azofra, A.; Benitez, J.M. ***Empirical Study of Feature Selection Methods in Classification.*** *Hybrid Intelligent Systems, 2008.* HIS apos;08. Eighth International Conference on Volume , Issue , 10-12 Sept. 2008 Page(s):584 - 589

[6] Jun Zhao Guo-Yin Wang Zhong-Fu Wu Hong Tang Hua Li . ***The study on technologies for feature selection***. *Machine Learning and Cybernetics, 2002.* Proceedings. Date: 2002 2002 International Conference on Publication. Volume: 2, On page(s): 689- 693 vol.2, ISBN: 0-7803-7508-4

[7] Huan Liu, Lei Yu: ***Toward Integrating Feature Selection Algorithms for Classification and Clustering.*** *IEEE Trans. Knowl. Data Eng. 17*(4): 491-502 (2005)

[8] Le Song, Alex J. Smola, Arthur Gretton, Karsten M. Borgwardt, Justin Bedo: ***Supervised feature selection via dependence estimation***. ICML 2007: 823-830

[9] Deisy, C. Subbulakshmi, B. Baskar, S. Ramaraj, N. ***Efficient Dimensionality Reduction Approaches for Feature Selection***. *Conference on Computational Intelligence and Multimedia Applications, 2007.* International Conference on Publication. Date: 13-15 Dec. 2007 Volume 2, On page(s): 121-127, Location: Sivakasi, Tamil Nadu,

[10] Qu, G. Hariri, S. Yousif, M. ***A new dependency and correlation analysis for features.*** *Knowledge and Data Engineering, IEEE Transactions on. Publication Date*: Sept. 2005. Volume: 17, Issue: 9 On page(s): 1199- 1207

[11] Isabelle Guyon, André Elisseeff: ***An Introduction to Variable and Feature Selection***. *Journal of Machine Learning Research* 3: 1157-1182 (2003)

[12] MacKay, David J.C. ***Probable networks and plausible predictions - a review of practical Bayesian methods for supervised neural networks***. *Network: Computation in Neural Systems*, 6:3,469-505.

[13] Yu Fu, Antony Browne: ***Investigating the influence of feature correlations on automatic relevance determination***. *IJCNN 2008*: 661-665

[14] Yuan (Alan) Qi, Thomas P. Minka, Rosalind W. Picard, Zoubin Ghahramani: ***Predictive automatic relevance determination by expectation propagation***. *ICML 2004.*

[15] Huanhuan Chen and Xin Yao:***Evolutionary Multiobjective Ensemble Learning Based on Bayesian Feature Selection.*** *In Proceedings of the IEEE Congress on Evolutionary Computation* (CEC'06),pages 971-978. Vancouver, Canada, 2006.

[16] Poland, Jan. ***On the Robustness of Update Strategies for the Bayesian Hyperparameter alpha***, Nov 2,2001

[17] David P. Wipf, Srikantan Nagarajan: ***A New View of Automatic Relevance Determination***. *NIPS 2007*

[18] Bai-Ning Jiang, Xiang-Qian Ding, Lin-Tao Ma, Ying He, Tao Wang, Wei-Wei Xie: ***A Hybrid Feature Selection Algorithm: Combination of Symmetrical Uncertainty and Genetic Algorithms.*** *The Second International Symposium on Optimization and Systems Biology (OSB'08).* Lijiang, China, October 31– November 3, 2008 Copyright © 2008 ORSC & APORC, pp. 152–157

[19] G.Van Dijck, M.M. Van Hulle, and M. Wevers: ***Genetic Algorithm for Feature Subset Selection with Exploitation of Feature Correlations from Continuous Wavelet Transform: a real-case Application.*** *International Journal of Computational Intelligence*, 1(1) 2004,pp. 1-12.

[20] Holland, J. H. ***Outline for a logical theory of adaptive systems***. *Journal of the Association for Computing Machinery*, vol. 3, p. 297 - 314, 1962.

[21] Jacek Jarmulak and Susan Craw: ***Genetic Algorithms for Feature Selection and Weighting***. *Appears in Proceedings of the IJCAI'99 workshop on Automating the Construction of Case Based Reasoners, 1999.*

[22] Laetitia Jourdan, Clarisse Dhaenens, El-Ghazali Talbi. ***A Genetic Algorithm for Feature Selection inData-Mining for Genetics***. *MIC'2001 4th Metaheuristics International Conference*. Porto, Portugal, July 16-20, 2001.

[23] Maria J. Martin-Bautista, Maria-Amparo Vila. ***A Survey of Genetic Feature Selection in Mining Issues.*** *Appears in: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on.* 1999 Washington, DC, USA.

[24] John G.H., Kohavi, R. and Pfleger, K. ***Irrelevant Features and the Subset Selection Problem***. In: *Proceedings of the Eleventh International Conference of Machine Learning*. 121-129, 1994

[25] H. Almuallim and T. G. Dietterich. ***Learning with many Irrelevant Features***. In *Proc. Of the 9th National Conf. On Artificial Intelligence*, volume 2 pages 547-552, Anaheim, CA, 1991. AAAI Press.

[26] C. Cardie. ***Using Decision Trees to improve Case-Based Learning***. In Proc. Of the 10th Int. Conf. On Machine Learning, pages 25-32, Amherst, MA, 1993. Morgan Kaufmann.

[27] Langley, P. (1994). Selection of relevant features in machine learning. In Procedings of the AAAI fall symposium on relevance, New Orleans, LA. Menlo Park, CA: AAAI Press.

[28] Antonio Arauzo-Azofra, José Manuel Benítez, Juan Luis Castro. ***Consistency Measures for Feature Selection***. *Springer Science + Business Media*, LLC 2007.

[29] Shannon, C.E. ***A Mathematical Theory of Communication***, *Bell Syst. Tech.* (1948),J., 27, 379-423, 623-656.

[30] Ben-Bassat, M., ***Pattern Recognition and Reduction of Dimensionality***. In: *Handbook of Statistics,* (P. R. Krishnaiah and L. N. eds.), North Holland, 773-791, 1982

[31] A. Skowron and C. Rauszer. ***The discernibility matrices and functions in information systems***. In R. Slowinsky, editor, *Handbook of Applications and advances of the Rough Set Theory*, pages 331-162. Kluwer Academic publishers, Dordrecht, 1992.

[32] H. Liu and H. Motoda. ***Feature Selection for Knowledge Discovery and Data***.

[33] Richard O. Duda, Peter E. Hart, David G. Stork. ***Pattern Classification***, 2nd Edition. ISBN: 978-0-471-05669-0

[34] Ian T. Nabney. ***NETLAB: Algorithms for Pattern Recognition***. ISBN: 1-85233-440-1

[35] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. ***Learning representations by back-propagating errors***. *Nature*, (1986) 323, 533--536.

[36] Fu,Y. & Browne, A. (2007). ***Using Ensembles of Neural Networks to improve Automatic Relevance Determination.*** *Proceedings of the 2007 IEEE International Joint Conference on Neural Network*, 1590-1594, Orlando, FL.

[37] Austin, S. ***An introduction to genetic algorithms***. *AI Expert*, vol. 5, n. 3, p. 48 -53, March 1990.

[38] Dukkipati, A.; Murty, M. N. ***Selection by parts: "selection im two episodes" in evolutionary algorithms***. *IEE Proceedings of the 2002 Congress on Evolutionary Computation*, vol. 1, p. 657 - 662, May 2002.

[39] Plagianakos, V. P.; Magoulas, G. D.; Vrahatis, M. N. ***Supervised training using global search methods***. *Advances in convex analysis and global optimisation*, vol. 54, p. 421 - 432, 2001.

[40] Michalewicz, Z.; Fogel, D. ***How to Solve It: Modern Heuristics***. *[S.l.]:Springer-Verlag*, 2000.

[41] Laarhoven, P. V.; Aarts, E. ***Simulated Annealing: Theory and Applications***. *[S.l.]: Kluwer Academic Publishers*, 1987

[42] Omran, M. G. H. ***Particle Swarm Optimization Methods for Pattern Recognition and Image Processing***. Thesis (Doctor) | *University of Pretoria,* Pretoria, November 2004.

[43] Michalewicz, Z. ***Genetic Algorithms + Data Structures = Evolution Programs***. *Third, revised and extended edition.* USA: Springer-Verlag, 1996.

[44] Back, T.; Schwefel, H.-P. ***An overview of evolutionary algorithms for parameter optimization***. *MIT Press, Cambridge*, MA, USA, vol. 1, n. 1, p. 1 - 23, 1993.

[45] Austin, S. ***An introduction to genetic algorithms***. *AI Expert*, vol. 5, n. 3, p. 48 -53, March 1990.

[46] Houck, C. R.; Joines, J. A.; Kay, M. G. ***A Genetic Algorithm for Function Optimization: A Matlab Implementation.*** [S.l.], 1995

[47] Ghoshray, S.; Yen, K. K. ***More efficient genetic algorithm for solving optimization problems***. *IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, p. 4515 - 4520, October 1995

[48] Chaiyaratana, N.; Zalzala, A. M. S. ***Recent developments in evolutionary and genetic algorithms: Theory and applications.*** *IEEE Second International Conference On Genetic Algorithms in Engineering Systems: Innovations and Applications*, n. 2 - 4, p. 270 - 277, September 1997.

[49] Kingdon, J.; Dekker, L. ***Development needs for diverse genetic algorithm design***. *IEEE Colloquium on Applications of Genetic Algorithms*, p. 1 - 11, March 1994.

[50] Lucasins, C. B.; Kateman, G. ***Application of genetic algorithms in chemometric***. *Proceedings of the Third International Conference on Genetic Algorithms*, p. 170 - 176, 1989.

[51] N. Griffith, D. Partridge. ***Self-Organizing Decomposition of Functions***. Volume 1857/2000. ISBN 978-3-540-67704-8

[52] D. Partridge. ***Network generalization differences quantified***. *Technical Report* 291, Department of Computer Science, University of Exeter, 1994.

[53] Software ***NETLAB***. http://www.ncrg.aston.ac.uk/netlab/index.php

[54] Asuncion, A. & Newman, D.J. (2007). ***UCI Machine Learning Repository*** [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science]

[55] Erich Huang, Skye H Cheng, Holly Dressman, Jennifer Pittman, Mei-Hua Tsou, Cheng-Fang Horng, Andrea Bild Edwin S. Iversen, Ming Liao, Chii-Ming Chen, Mike West, Joseph R Nevins and Andrew T Huang. ***Gene Expression Predictors of Breast Cancer Outcomes***. Lancet (2003) 361, 1590-1596

[56] ***MATLAB*** http://www.mathworks.com/

[57] Software ***WEKA*** http://www.cs.waikato.ac.nz/ml/weka/

[58] Møller, Martin F. ***A scaled conjugate gradient algorithm for fast supervised learning***. *Appears in Neural Networks (1993).*

[59] Ronald Fisher. ***The Use of Multiple Measurements in Taxonomic Problems***. In: *Annals of Eugenics*, 7, (1936) p. 179--188

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.
UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Selección de Atributos para Clasificación Usando Aprendizaje Bayesiano y Computación Evolutiva

del (la) C.

Ana Isabel RODRÍGUEZ BARRAGÁN

el día 08 de Marzo de 2010.

Dr. Eduardo José Bayro Corrochano
Investigador CINVESTAV 3D
CINVESTAV Unidad Guadalajara

Dr. Mario Angel Siller González Pico
Investigador CINVESTAV 2A
CINVESTAV Unidad Guadalajara

Dr. Humbero Sossa Azuela
Profesor Investigador
Centro de Investigación en Computación del Instituto Politécnico Nacional