UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD #6

PROPUESTA DE ALGORITMOS PARA LA REDUCCIÓN DEL ESPACIO MUESTRAL

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

AUTORES:

TONYSÉ DE LA ROSA MARTÍN.

HERMES LÁZARO HERRERA MARTÍNEZ.

TUTORES:

DR. RAMÓN CARRASCO VELAR.	**CQF**
MSC. AURELIO ANTELO COLLADO.	**UCI**
ING. YAIKIEL HERNÁNDEZ DÍAZ.	**UCI**

CIUDAD DE LA HABANA, CUBA

JUNIO, 2008

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Hermes Lázaro Herrera Martínez (Autor) Tonysé de la Rosa Martín (Autor)

_____ _____

Dr. Ramón Carrasco Velar (Tutor) MSc. Aurelio Antelo Collado (Tutor)

_____

Ing. Yaikiel Hernández Díaz (Tutor)

“Si alguien puede convencerme y probarme que pienso o actúo mal, seré feliz corrigiéndome, porque busco la verdad, que nunca ha hecho daño a nadie. Pero se perjudica quien persiste en su error y en su ignorancia.”
-Marco Aurelio-

Datos de Contacto

Autores:

Hermes Lázaro Herrera Martínez **email:** **hlherrera@estudiantes.uci.cu**

Tonysé de la Rosa Martín **email:** **tdelarosa@estudiantes.uci.cu**

Tutores:

Ramón Carrasco Velar **email:** **rcarrasco@uci.cu**

Aurelio Antelo Collado **email:** **aantelo@uci.cu**

Yaikiel Hernández Díaz **email:** **yhernandezd@uci.cu**

Agradecimientos

De Tonysé:

A mi familia que me hizo el hombre que soy y que sin ella no podría estar hoy donde estoy.
A Yaikiel, a Aurelio y al Doctor Ramón Carrasco, mis tres tutores, sin la sabiduría y consejo de ellos no se podría haber hecho este trabajo.
A Hermes, mi compañero de tesis, sin sus conocimientos y esfuerzo no se hubiese terminado este trabajo.
A la Mamá de Hermes por su constante preocupación para que este trabajo salga bien.
A mis amigos que han estado conmigo estos 5 años, los que han hecho mi vida más divertida e interesante.
A mis profesores en los 5 años de carrera en la UCI, que me enseñaron lo que se.
A la Revolución y en especial al Comandante en Jefe Fidel Castro, por construirnos una universidad tan maravillosa.

A todos,

Muchas gracias.

Agradecimientos

De Hermes:

A mi familia, especialmente a mi mamá.

A mis compañeros y profes de la UCI que han compartido conmigo un pedazo de vida, por haber compartido tantos sueños y desvelos.

A mi comandante Fidel Castro Ruz, por ser siempre un humilde soldado de la patria, a nuestra revolución por defender valores al precio de cualquier sacrificio.

Al Dr. Manuel Mariño, por mostrarme el camino para ser un ingeniero de estos tiempos.

A mi profesor Medina que ya no está, existen pocos matemáticos como el.

A mis tres tutores, en especial a Yaikiel por su ayuda y preocupación constante, al Dr. Carrasco por complejizar las soluciones y ayudarnos a descubrir los problemas.

A mi compañero de tesis Tonysé, alguien que no puede faltar, sino sería contradictorio e imposible haber llegado a una solución tan compleja, de un modo tan sencillo.

Dedicatoria

De Tonysé:

A mi madre Maritza, las más grande de todas, no hay palabras para describirla porque está por encima de lo natural, es mi fuerza y mi espada.

A mi padre Antonio quien siempre me ha aconsejado y guiado en tiempos buenos y malos, me dio la poca sabiduría que tengo.

A mi hermana Claudia, que aunque no lo crea, la quiero mucho, y deseo lo mejor para ella.

A mi otro padre William, pilar importante en mi vida, lo respeto y lo quiero mucho, véase en este trabajo también sus manos.

A mi tía Ada Amelia, otra madre que tengo, y su esposo Jorge quien es como un padre para mí, por el amor y la comprensión que me han dado en todos estos años.

A mis dos abuelas, la que no está, Juana, y la que está Marta, ellas me dieron lo noble y lo honrado, mi amor para ellas.

A mis abuelos Arsenio y el Gallego.

A mi tía Odalys, mi tío Alberto a sus hijos Fanny y El Chino (el hermano que no me dieron).

A mi tía Gladys, su esposo Pedro y a sus hijos Yalia (otra hermana) y Yaxel.

A mi tío Toto y sus hijas, una de ellas, Oly, espero verla un día con el título de Ingeniera en Ciencias Informáticas.

A mi primo Dinielys, a su esposa Maydi y su niño Diniel Antonio.

A mi tío Juanci y sus niñas.

A mi tío Fidel, su esposa y sus hijos.

En fin dedicada a toda mi familia, lo más importante que existe en el mundo para mi.

Los quiero.

De Hermes:

Especialmente para mi mamá Migdalia, que abnegadamente me ha apoyado y guiado en todo momento, no hay espacio ni tiempo capaz de describir mi amor por ella.

A mi padre, de quien admiro insaciablemente su inteligencia.

A todos mis hermanos, que tanto me aman y ayudan, y los quiero con mi vida.

A mi tío Reynold y a mi abuelita Olga.

A mi tita linda que tanto amo.

A mi primo dulcero, que tanto me admiró mientras estuvo a mi lado.

A todos mis amigos.

Resumen

En el presente trabajo se presentan la implementación y evaluación de varios métodos de reducción de la dimensionalidad basados en técnicas de inteligencia artificial, y aborda uno de sus complejos problemas, como es el identificar y reducir un conjunto representativo de atributos para así contribuir al mejoramiento de los modelos de clasificación y predicción. La búsqueda de subconjuntos óptimos de atributos para la clasificación de conjuntos de datos presenta el inconveniente de su complejidad temporal. Se implementaron procedimientos de búsqueda por algoritmos genéticos, enfriamiento simulado, búsqueda secuencial y una hibridación entre este último y algoritmos genéticos, con tal de alcanzar mayor robustez y eficiencia. Se implementan además varias medidas de asociación entre subconjuntos variables, a partir de conceptos de la estadística clásica o tomadas de la Teoría de la Información de Shannon. En todos los casos experimentados se reduce el espacio muestral en más del 65%. Los mejores resultados se alcanzan con el algoritmo híbrido y la búsqueda secuencial, obteniendo reducciones en más del 70%. Todos estos procedimientos de búsqueda presentan una complejidad temporal de orden polinomial, esto demuestra la viabilidad práctica en costo y recursos computacionales de cada procedimiento implementado. Se presenta el diseño de clases de la aplicación desarrollada así como el uso de varios patrones de diseño, como contribución a su posterior inclusión en la plataforma.

Palabras Claves: *reducción, dimensionalidad, clasificación, predicción, subconjuntos, atributos, inteligencia artificial.*

AGRADECIMIENTOS ..... **I**
DEDICATORIA ..... **III**
INTRODUCCIÓN **1**
CAPÍTULO 1 ..... **5**
1.1 INTRODUCCIÓN A LA SELECCIÓN DE VARIABLES **6**
1.2 CONCEPTOS BÁSICOS **8**
***1.3* MÉTODOS DE SELECCIÓN DE VARIABLES O CARACTERÍSTICAS** **9**
1.4 CLASIFICACIONES DE LOS MÉTODOS DE SELECCIÓN DE CARACTERÍSTICAS **10**
1.5 DESCOMPOSICIÓN MODULAR DE LOS MÉTODOS DE SELECCIÓN **11**
1.6 MÉTODOS DE BÚSQUEDA **12**
1.6.1 ESPACIO DE BÚSQUEDA 12
1.6.2 ESTRATEGIAS DE BÚSQUEDA 13
1.6.2.1. Búsqueda secuencial *13*
1.6.2.2. Búsqueda completa *13*
1.6.2.3. Búsqueda probabilística *14*
1.6.2.4. Búsqueda heurísticas y con metaheurísticas *14*
1.7 LOS MÉTODOS Y TÉCNICAS DE BÚSQUEDA HEURÍSTICA Y METAHEURÍSTICAS. **15**
1.8 SOFTWARES VINCULADOS A LA SELECCIÓN DE VARIABLES A NIVEL MUNDIAL **18**
CAPÍTULO 2 ..... **21**
2.1 LOS ALGORITMOS GENÉTICOS. **22**
2.1.1 PASOS PARA CONSTRUIR UN ALGORITMO GENÉTICO 22
2.1.2 ESTRATEGIA DE SELECCIÓN. 23
2.1.3 ALGUNOS ESQUEMAS DE SELECCIÓN 23
2.1.3 VENTAJAS Y DESVENTAJAS DE LOS ALGORITMOS GENÉTICOS 23
2.1.4 ¿POR QUÉ UTILIZAR ALGORITMOS GENÉTICOS? 24
2.1.5 APLICACIONES DE LOS ALGORITMOS GENÉTICOS. 25
2.2 ALGORITMO DE ENFRIAMIENTO SIMULADO **25**
2.2.1 ESQUELETO DE UN TA (ALGORITMOS DE UMBRAL) 26
2.2.2 APLICACIONES DEL ALGORITMO ENFRIAMIENTO SIMULADO. 27
2.3 MEDIDAS DE EVALUACIÓN **27**
2.3.1 MEDIDAS DE EVALUACIÓN SOBRE VARIABLES INDIVIDUALES. 28
2.3.2 MEDIDAS DE EVALUACIÓN SOBRE CONJUNTOS VARIABLES. 29
2.3.2.1 Medidas basadas en consistencia *29*

2.3.2.2 Medidas basadas en la Teoría de la Información *30*
2.3.2.3 Medidas Basadas en la Distancia *32*
2.3.2.4 Medidas Basadas en la Dependencia *32*
2.4 METODOLOGÍAS Y HERRAMIENTAS PARA EL DESARROLLO DEL SISTEMA. **33**
2.4.1 PLATAFORMA DE DESARROLLO Y LENGUAJE DE PROGRAMACIÓN. 33
2.4.2 ENTORNO DE DESARROLLO. 34
2.4.3 HERRAMIENTA CASE (COMPUTER AIDED SOFTWARE ENGINEERING). 34
2.4.4 LENGUAJE DE MODELADO. 35
2.4.5 GESTOR DE BASE DE DATOS UTILIZADO. 36
RESULTADOS Y DISCUSIÓN. **38**
3.1 MODELO CONCEPTUAL DEL SISTEMA **39**
3.2 DIAGRAMA DE CLASES Y PATRONES UTILIZADOS. **40**
3.2.1 DIAGRAMA DE CLASES. 40
3.2.2 PATRONES UTILIZADOS EN LA RESOLUCIÓN DEL PROBLEMA. 42
3.2.2.1 Patrón *42*
3.2.3.2 Importancia de utilizar Patrones *42*
3.2.3.3 Patrones GRASP *42*
3.3 DESCRIPCIÓN DE LA SOLUCIÓN **49**
3.3.1 ALGORITMOS IMPLEMENTADOS 49
3.3.1.1 Algoritmo Genético *49*
3.3.1.2 Enfriamiento Simulado *52*
3.3.1.2 Hibridación entre Algoritmo Genético y algoritmo de Búsqueda Secuencial *54*
3.5 ANÁLISIS DE LOS RESULTADOS **58**
CONCLUSIONES GENERALES **62**
RECOMENDACIONES **63**
REFERENCIAS BIBLIOGRÁFICAS **64**
BIBLIOGRAFÍA **68**
ANEXOS **71**
GLOSARIO DE TÉRMINOS **77**

Introducción

Nos movemos desde el último medio siglo inmersos en una época que tradicionalmente ha sido llamada “la era de la Ciencia”, y la sociedad de tal era no puede comprenderse sin destacar una de esas múltiples disciplinas científicas, que sin duda, es y será un referente para la investigación futura: la farmacología. Representando una respuesta alternativa y eficaz a los nuevos problemas en materia de salud que surgen en la sociedad actual. Disciplina científica que no fuera posible sin la Química, ciencia experimental que ordena la materia, clasificando los compuestos como: alcoholes, cetonas, ácidos carbónicos, así como azúcares, grasas y proteínas y los tipos de reacciones tanto de sustitución, eliminación entre otras. Relacionando las diferentes materias y describe sus transformaciones.

Entre las muchas funciones de la farmacología la más importante es la creación de medicamentos de alta calidad para la preservación de la salud de los seres humanos, de ahí que los medicamentos son la base para casi cualquier programa de salud pública intencionado a reducir la morbilidad o mortalidad en los países en desarrollo y subdesarrollados, aunque en los desarrollados también se tiene la cuestión del lucro con ellos, un ejemplo claro es el hecho de que medicamentos necesarios para el tratamiento de ciertas enfermedades tropicales han empezado a desaparecer del mercado porque no son comercialmente rentables.

El gasto farmacéutico es una alta proporción del gasto total en salud de un país, y todos los programas de salud pública necesitan medicamentos de alta calidad para mejorar los programas de supervivencia infantil, cuidados prenatales, tratamiento de patógenos entéricos y respiratorios, control de tuberculosis y malaria, poliomielitis entre muchas otras enfermedades curables, y en la mayoría de los casos especialmente en los países del tercer mundo no se cuenta con los fondos monetarios suficientes para la obtención de estos medicamentos. También existen otras enfermedades importantes de salud pública para los que no hay un tratamiento farmacéutico eficaz como el cáncer y el Virus de Inmunodeficiencia Humana (VIH) que no pueden curarse pero para los cuales se han creado medicamentos con el objetivo hacer más llevadera la enfermedad, estos cuentan con el inconveniente de que su precio se torna prohibitivo para personas pobres o incluso de clase media, ejemplo de esto, el de los medicamentos antirretrovirales para el tratamiento de pacientes con el síndrome de la inmunodeficiencia adquirida el cual no es posible afrontar dado su alto coste, muchos de los cuales han sido comercializados recientemente y, por tanto, todavía están protegidos por patentes.

Nuestro país se ve privado de la obtención de estos medicamentos de alta calidad debido a dos causas fundamentales, la primera el férreo bloqueo económico y financiero que nos ha impuesto el gobierno de los Estados Unidos de Norteamérica por más de 45 años, impidiendo con este que ninguna empresa o país que fabrica medicamentos de alta calidad y que a la vez son muy necesarios para el programa de salud cubano puedan comercializarse con nuestro país, pues estas empresas que los exportan se verían ante el problema de pagar una multa que oscila entre los 100 millones y los 500 millones de dólares norteamericanos al gobierno de los EE.UU. por prestar ayuda a un país "sancionado internacionalmente". La segunda causa es el alto coste de estos medicamentos en el mercado mundial, Cuba cuando encuentra una de estas empresas productoras de medicamentos esenciales para algunas enfermedades como el cáncer o el VIH y esta crea algún acuerdo para vender sus productos los cobra a muy elevado precio, más elevado que el precio del producto en el mercado mundial por el riesgo de ser multados.

Esta situación ha obligado al país a pensar y crear estrategias para solucionar el problema de la creación de medicamentos de alta calidad, que ayuden al pueblo cubano a tener una vida más duradera y para aquellos pacientes que no tengan cura efectiva para su enfermedad puedan convivir con ella más tiempo. También la creación de medicamentos representaría una ayuda importante en concepto de bienes monetarios para el país al exportarlos y con ellos se ayudaría a otras naciones amigas necesitadas de estos medicamentos que a tan alto precio se obtienen en el mercado mundial.

Por lo anterior expuesto el presente trabajo se encuentra enmarcado dentro del proyecto de investigación científica conjunta entre el Centro de Química Farmacéutica y el grupo de Bioinformática de la Universidad de las Ciencias Informáticas (BioSoft) denominado Plataforma para la Predicción de Actividad Biológica de Compuestos Orgánicos (GRAph-TOol).

Actualmente la plataforma GRAph-TOol cuenta con bases de datos de gran tamaño formada por moléculas y sus descriptores asociados. Estos son utilizados por los métodos de inteligencia artificial implementados en la plataforma para la predicción de actividad biológica asociando esta a la estructura química. Dichos métodos realizan la predicción utilizando una cantidad elevada de descriptores topológicos, topográficos e híbridos, aunque solo algunos de ellos aportan información útil para el establecimiento de los modelos. La generalidad de esos descriptores parten de formulismos que se basan en la matriz de conectividad de los vértices o aristas del grafo químico por lo que se encuentra elevada redundancia en la información que ellos contienen. Otro problema es el elevado consumo de

los recursos de cómputo cuando se necesita procesar una cifra tan elevada de datos. Por lo tanto, se hace necesario implementar una herramienta que contribuya a la reducción del espacio muestral de descriptores, con el fin de eliminar gran parte de la redundancia de información en la base de datos y para mejorar la eficiencia y costo computacional del establecimiento de los modelos y la realización de las predicciones.

Son diferentes los procedimientos que se emplean en la actualidad para la reducción de la dimensión en una muestra dada. Entre los más modernos se destacan las técnicas de inteligencia artificial, que se emplean solas o combinadas con técnicas clásicas de la estadística avanzada.

Teniendo en cuenta los diferentes aspectos analizados se plantea como ***hipótesis*** de trabajo, que es posible encontrar una técnica de IA que permita la reducción de la dimensión en muestras con datos numerosos y de elevada redundancia.

Se define por lo tanto como **Objeto de Estudio:**

La selección de variables para la reducción del espacio muestral

Y como **Campo de Acción:**

Los algoritmos de búsqueda y evaluación para la selección de variables.

Por lo que se adopta como **Objetivo General:**

Proponer algoritmos de búsqueda y evaluación para la reducción del espacio muestral.

Para dar cumplimiento al objetivo general se definieron como **Objetivos Específicos:**

- ✓ Identificar algoritmos aplicables a la reducción de espacio muestral.
- ✓ Implementar los algoritmos seleccionados.
- ✓ Validar los algoritmos implementados.

Para dar cumplimiento a los objetivos específicos se definieron las siguientes tareas:

- ✓ Revisión del estado del arte acerca de algoritmos de selección de variables y de criterios de evaluación reportados en la literatura.
- ✓ Aplicación de técnicas estadísticas para la reducción inicial del espacio de búsqueda.
- ✓ Implementación de los algoritmos seleccionados para dar respuesta a la reducción del espacio muestral.
- ✓ Realización de las pruebas de validación del algoritmo seleccionado.

Como aporte práctico se espera:

Introducción

Proponer algoritmos eficientes para la reducción del espacio muestral, que faciliten la creación de una aplicación incorporable como módulo a la plataforma GRAph-TOol.
El presente trabajo de diploma cuenta con tres Capítulos:

Capítulo # 1: Reseña Bibliográfica:
En este capítulo se hace una introducción al problema de selección de variables o reducción de la dimensión exponiéndose conceptos fundamentales del mismo. Se plantean las técnicas y métodos que se usan a nivel mundial, según la bibliografía consultada, presentando además algunos sistemas automatizados que se utilizan estos métodos.

Capítulo # 2: Métodos y materiales:
Se presentan los algoritmos de búsqueda de la Inteligencia Artificial para la resolución de problemas de reducción de la dimensionalidad. Se explican las características favorables que presentan los mismos, así como las medidas de evaluación utilizadas por estos procedimientos, mostrando además las distintas herramientas utilizadas para la implementación y evaluación de los algoritmos seleccionados.

Capitulo # 3: Resultados y Discusión:
En este capítulo se presenta un modelo conceptual del sistema para lograr una mayor claridad del sistema implementado. Se presenta el diagrama de clases del diseño del sistema y los patrones de diseño empleados en la programación orientada a objetos. Se realiza una descripción detallada de los algoritmos implementados y la validación de los mismos a través de pruebas con datos reales.

1

CAPÍTULO
Reseña Bibliográfica.

1.1 Introducción a la Selección de Variables

El problema de la selección de variables consiste en encontrar un subconjunto de variables que sean relevantes para una aplicación y lograr el máximo rendimiento con el mínimo esfuerzo, con las que se pueda llevar a cabo la tarea de clasificar de forma óptima. Reducir la dimensión conlleva diversas ventajas como la de reducción del coste en la adquisición de datos, mejora en la comprensión del modelo final obtenido, incremento de la eficiencia del clasificador y mejora en la eficacia del clasificador. En resumen seria:

Menos datos → los algoritmos pueden aprender más rápidamente.
Mayor exactitud → el clasificador generaliza mejor.
Resultados más simples → más fácil de entender.

La búsqueda de un subconjunto de variables es un problema de tipo No Polinomial completo (NP-completo) [1], los que universalmente no tienen solución práctica, el uso de meta-heurísticas permite obtener soluciones razonablemente buenas sin explorar todo el espacio de soluciones.

Si tratáramos de seleccionar un subconjunto de '*m'* características de entre un conjunto original de '*n'* características candidatas, bajo algún criterio de desempeño. Encontraríamos un total de:

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}$$ Subconjuntos.

El número de posibilidades crece exponencialmente, haciendo impráctica la búsqueda exhaustiva, aun para valores moderados de '*n'* (**Tabla 1.1**). Si se evalúa todo el espacio de posibles combinaciones, el costo computacional es muy alto. Ejemplo de esto sería.

Si '*n*' es la cantidad de características identificadas y '*m*' es la cantidad de características deseadas, el número total de posibles subconjuntos a evaluar es:

$$S = \sum_{m} C(n,m) = \sum_{m} \frac{n!}{m!(n-m)!} \qquad Si\ n = m \Rightarrow S = 2^n$$

Tabla 1.1 Total de subconjuntos generados para algunos valores de 'n'.

n	2^n
10	***1 024***
20	***1 048 576***
30	***1 073 741 824***
40	***1 099 511 627 776***
406	**No pudo ser calculado**

Nos encontraríamos entonces frente al problema conocido como Maldición de la Dimensión debido a la gran cantidad de datos que generalmente se manejan, y en la mayoría de los casos estamos en presencia de problemas de tipo No Polinomial Completo (NP-completo).

En la plataforma específicamente presenta un problema de este tipo, pues cuenta con una base de datos que contiene alrededor de 406 descriptores moleculares, los cuales se utilizan para la predicción de actividad biológica en los clasificadores implementados en la plataforma. Y para lograr una buena predicción es necesario seleccionar un subconjunto de estos descriptores que aporten buena información.

La inteligencia artificial brinda varios artificios para la resolución de muchos problemas de manera eficiente, problemas que por lo general son complejos o muy complejos, lo que imposibilita en gran medida hallarles una solución exacta. Por lo cual, este nuevo campo de la ciencia de la computación ha venido como anillo al dedo a esta era, no solo para la resolución de los problemas matemáticos , estadísticos, naturales, sociales, etc., sino también para la mejora de los mismos hasta que esta se encuentre cerca de lo que se conoce como solución óptima o mejor solución posible.

El hecho de encontrar un método eficaz para la selección de variables es una tarea ardua y difícil, debido a la poca información que se puede hallar al respecto.

Antes de enfrentarse a un problema de selección de características o de reducción de espacio muestral es necesario conocer ciertos elementos básicos del entorno del problema para una mejor comprensión del mismo. Entre ellos están el Dominio, Universo de Discurso, Atributo y Espacio Muestral.

1.2 Conceptos Básicos

Inteligencia Artificial

Se denomina Inteligencia Artificial (IA) a la ciencia que desarrolla procesos que imitan a la inteligencia de los seres vivos. La principal aplicación de esta ciencia es la creación de máquinas para la automatización de tareas que requieran un comportamiento inteligente, así como brindar algoritmos y soluciones para problemas reales de los seres humanos.

Entre las muchas aplicaciones que presenta a la IA se encuentran en el área de control de sistemas, planificación automática, la habilidad de responder a diagnósticos y a consultas de los consumidores, reconocimiento del habla y reconocimiento de patrones. Los sistemas de IA actualmente son parte de la rutina en campos como la medicina, ingeniería y la milicia, y se ha usado en gran variedad de aplicaciones de software, juegos de estrategia como ajedrez de computador y otros.

El matemático sudafricano, Seymour Papert, es considerado pionero en esta ciencia. [2]

Dominio

Un dominio es un conjunto de valores del mismo tipo. Aunque existen distintas clasificaciones de los dominios, para los propósitos de esta investigación se distinguen dos tipos: continuo (conjunto infinito de valores reales) y nominal (conjunto finito de valores discretos) representa *Dom* ().

Universo de Discurso

Se denomina Universo de discurso al entorno donde se define un determinado problema y viene representado como el producto cartesiano de un conjunto finito de dominios.

Atributo

Un atributo, o también denominado característica o variable, es la descripción de alguna medida existente en el universo de discurso que toma valores en un determinado dominio.

El atributo i-ésimo se representa X_i, su valor x_i y su dominio como $Dom(x_i)$, que según la clasificación descrita previamente puede ser discreto o continuo. Si es continuo existe un rango $[a;b] \in R$ de valores posibles, y si es discreto existe un conjunto finito de valores posibles. Se denomina vector atributos $x = x_1;...; x_n$ al conjunto de valores correspondiente a cada uno de los atributos, y X al espacio formado por el conjunto de los atributos, $X = Dom(x_1)x....xDom(x_n)$ siendo '*n*' el total de atributos.

Espacio Muestral

Se denomina resultado básico o elemental, comportamiento individual o punto muestral a cada uno de los posibles resultados de un experimento aleatorio. Los resultados básicos elementales serán definidos de forma que no puedan ocurrir dos simultáneamente pero si uno necesariamente.

Se denomina conjunto universal, espacio muestral o espacio de comportamiento al conjunto de todos los resultados elementales del experimento aleatorio. Pueden ser de varios tipos, como por ejemplo el Espacio Muestral Discreto y el Espacio Muestral Continuo [3].

En esta investigación se define un espacio muestral discreto constituido por el conjunto de valores que toman los descriptores y fragmentos moleculares, variables (características) predictivas de la actividad biológica de un experimento dado.

1.3 Métodos de selección de Variables o Características

Desde su origen, la IA se encuentra encarada con problemas para los que no existe método analítico alguno que permita obtener, con seguridad y en un tiempo conveniente, el óptimo teórico. Éste es, por ejemplo, el caso de los problemas combinatorios en que el sentido común da por imposible la enumeración. Es más que normal que el tamaño y la naturaleza de ciertos problemas combinatorios nos prohíban abordarlos por la vía del sentido común. Dicha investigación distingue particularmente los problemas NP-completos, para los cuales no existe un algoritmo que en tiempo polinomial sea capaz de encontrar la solución [4]. De siempre, la investigación de operaciones ha establecido, por tales razones, métodos denominados heurísticos o meta-heurísticos, incapaces de proporcionar el óptimo

formal, pero susceptibles de llegar a soluciones buenas, tanto más fiables en cuanto que permiten determinar al mismo tiempo una cota (superior o inferior) del óptimo teórico con el que se comparan.

En un sistema de aprendizaje, pueden utilizarse diversas herramientas en combinación con el propio algoritmo de aprendizaje, como por ejemplo: la discretización o continuación de características, la sustitución de valores nulos, etc. En esta tesis, trabajaremos con modelos basados en aprendizaje supervisado. El objetivo de la selección de características (variables) es reducir la dimensión de los datos. Esto se consigue eligiendo características que sean útiles para resolver el problema de aprendizaje y descartando las demás. Aunque, teóricamente, si la distribución estadística completa se conociese, usar mas características solo podría mejorar los resultados, en los escenarios prácticos de aprendizaje puede ser mejor usar un conjunto reducido de características [5].

1.4 Clasificaciones de los métodos de selección de características

La metodología de filtro (filter): Es probablemente la primera y más conocida. En ella, se aplica primero el algoritmo de selección de características y, posteriormente, el de aprendizaje empleando solo las características seleccionadas.
La única información que intercambian es el conjunto de características, lo que aporta la principal ventaja de este tipo de métodos: que son independientes del algoritmo de aprendizaje. Por ello, pueden ser utilizados con cualquier algoritmo de aprendizaje, independientemente de su eficiencia u otras propiedades, que sí afectan a otros modos de aplicación.

En la estrategia envolvente (wrapper): El método de selección de características usa el algoritmo de aprendizaje para evaluar la calidad de los conjuntos de características, utilizando alguna medida de la calidad de las soluciones que obtiene este con cada uno de los conjuntos de características candidatos. En este proceso, hay un flujo de información en ambos sentidos. En uno, el método de selección de características indica un conjunto de características a usar y, en el otro sentido, se devuelve una evaluación de lo útiles que son esas características. Esto se repite hasta que finalmente se selecciona un conjunto definitivo.

La principal ventaja de esta estrategia es que la selección de características tiene una evaluación de las características en el entorno real en que serán aplicadas y, por tanto, tiene en cuenta las posibles particularidades del algoritmo de aprendizaje que se va a usar. Sin embargo, se genera una relación de dependencia entre ambos algoritmos que impone ciertos requisitos sobre el algoritmo de aprendizaje,

ya que este debería ser capaz de trabajar con los conjuntos de características que determine probar el método envolvente.

Inmersa (embebed): En algunos algoritmos de aprendizaje, la selección de características esta incluida en el mismo como una parte no separable. En este caso, la ventaja es que la selección de características este diseñada de forma especifica para ese aprendizaje, con lo que se espera que su rendimiento sea mejor.

1.5 Descomposición Modular de los métodos de selección

Dicha descomposición modular para el proceso de selección de características esta basada en las cuatro funcionalidades identificadas por Langley [6]. Las funciones son similares a las propuestas por Dash y Liu [7].

La descomposición modular se muestra en la **figura 1.2**. En ella, agrupamos las funciones en dos bloques principales:

- Evaluación de las características.
- Método de búsqueda en el espacio de conjuntos de características.

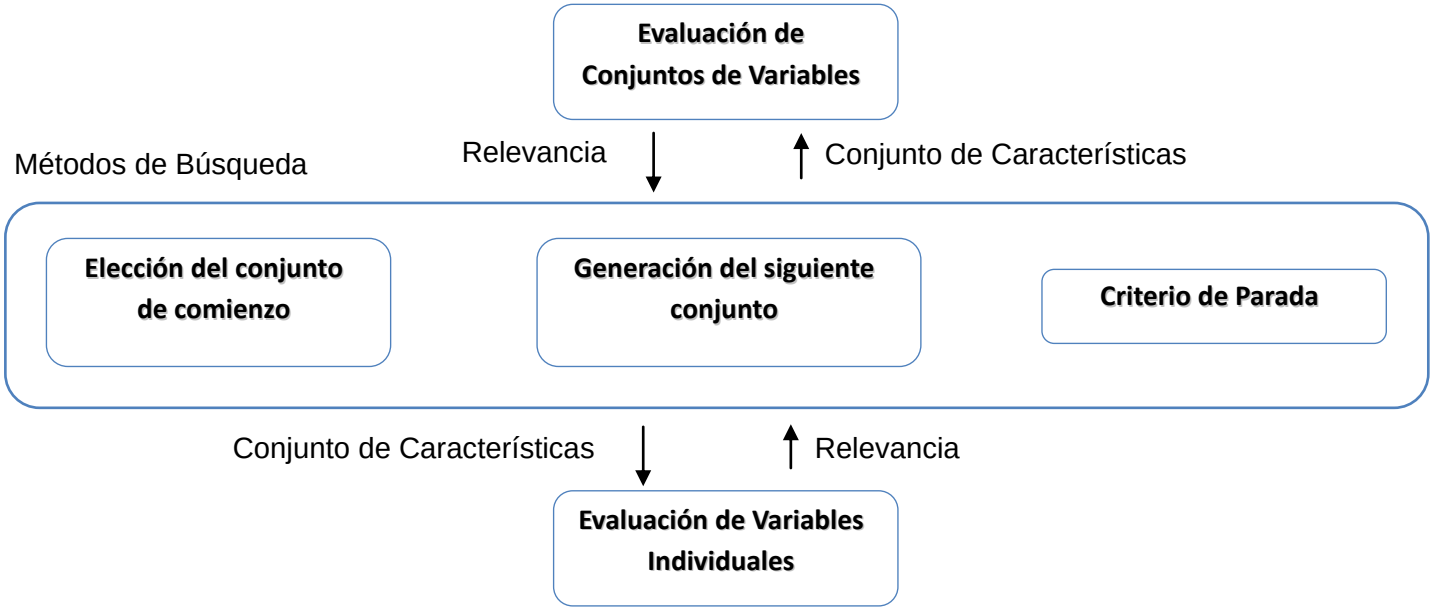


Figura 1.2 Descomposición Modular de los Métodos de Selección

El método de búsqueda gobierna el flujo de control del algoritmo, mientras que las funciones de evaluación de características son herramientas usadas por este. Dentro del método de búsqueda, consideramos tres partes importantes:

- la elección del punto de comienzo de la búsqueda, que puede ser inmediata o elegirse mediante un proceso más elaborado.
- estrategia de orientación de la búsqueda, o lo que es lo mismo, elección del siguiente conjunto a evaluar.
- el criterio de parada, que puede depender de muy diversos factores, como el número de evaluaciones, tiempo de ejecución, o el alcance de alguna condición sobre los resultados.

En la evaluación de las características, es importante diferenciar dos tipos de medidas. Por un lado, las que evalúan conjuntos de características y, por el otro, las que evalúan características individualmente. La visión modular del proceso de selección de características presentada nos aporta diversas ventajas. En primer lugar, nos permite alcanzar una mejor comprensión de los métodos de selección de características al permitirnos comprender su estructura interna de forma más ordenada. Usando este modelo, también es posible crear una gran variedad de algoritmos de selección de características simplemente combinando diferentes funciones de evaluación y mecanismos de búsqueda.

1.6 Métodos de Búsqueda

El problema de la selección de características puede ser visto como un problema de búsqueda en el conjunto potencia del conjunto de las características disponibles [8] [6]. Su objetivo es encontrar un subconjunto de características que nos permita mejorar, en algún aspecto, una actividad de aprendizaje.

1.6.1 Espacio de Búsqueda

Cuando se va a implementar un problema de búsqueda es importante conocer la forma y propiedades del espacio donde se va a realizar la búsqueda. En el caso de la reducción de espacio muestral, el espacio de búsqueda es el conjunto potencia de las características. Para un problema con ***"n"*** características es posible formar 2^n subconjuntos diferentes, de modo que el tamaño del espacio de búsqueda es 2^n, esta crece exponencialmente con el número de características.

Teniendo en cuenta el espacio de búsqueda al que nos enfrentamos, se puede demostrar que el problema de la selección de características es un problema NP-duro o completo [1] [9]. Podría incluso ser más duro, si la evaluación de los conjuntos de características no fuese polinomial.

1.6.2 Estrategias de Búsqueda

1.6.2.1. Búsqueda secuencial

Los algoritmos glotones (greedy) son aquellos que van tomando decisiones sin replantearse las anteriores y se caracterizan por su simplicidad y eficiencia, por lo que habitualmente se emplean para implementar heurísticas. En el caso de los procedimientos de búsqueda, siguen un camino sin volver nunca hacia atrás. También se les denomina de búsqueda secuencial porque siguen una secuencia de pasos sin vuelta atrás. Son los algoritmos búsquedas más simples.

Existen varios algoritmos secuenciales entre estos tenemos el SFS (búsqueda secuencial hacia delante) que parte del conjunto vacío de características y va añadiendo la variable que en mayor medida mejora la selección. Por otro lado, tenemos el que hace la búsqueda en sentido inverso SBS (búsqueda secuencial hacia atrás), partiendo del conjunto de todas las características y eliminando las de menor significación. Pueden además encontrarse algoritmos que combinan ambas estrategias, conocidos como bidireccionales para tratar generar estados que son imposibles de hallar mediante los algoritmos anteriores.

1.6.2.2. Búsqueda completa

Estas estrategias de búsqueda garantizan encontrar el conjunto de características óptimas, según el criterio de evaluación utilizado. La más obvia es la búsqueda exhaustiva, que evalúa todos los subconjuntos posibles para quedarse con el mejor. Algunos ejemplos de este tipo de búsqueda son A*, IDA*, Ramificación y Poda (Branch & Bound) entre otros.

De todos, uno se destaca en especial: el A*. Este algoritmo, a pesar de haber sido creado entorno a los años 60, sigue en la actualidad siendo uno de los más utilizados. Desafortunadamente, es ineficiente en cuanto al uso de memoria durante el proceso de búsqueda. Por ello, en las décadas de los 80 y 90, aparecieron algoritmos basados en el propio A*, pero que limitaban el uso de memoria. Dos de los algoritmos más representativos de esta última tendencia son el IDA* (Iterative-Deepening A*) y el SMA* (Simplified Memory-bounded A*).

1.6.2.3. Búsqueda probabilística

Los algoritmos probabilísticos [10] son aquellos que dejan algunas de sus decisiones al azar. En esta búsqueda podemos ver algoritmos que exploran aleatoriamente el espacio de búsqueda, o por zonas, o siguiendo algún criterio que dependa de algún factor aleatorio. LVF (Las Vegas Filter) [11] es un algoritmo de selección de características probabilístico pensado para medidas de evaluación de tipo filtro monótonas. Explora aleatoriamente conjuntos con igual o menor numero de características que el mejor que ha encontrado hasta el momento, y finaliza después de un numero de pasos especificados en un parámetro.

LVW (Las Vegas Wrapper) [12] es un algoritmo similar, pero pensado para aplicar la estrategia envolvente. También es apropiado para medidas de tipo filtro no monótonas. En este algoritmo, los conjuntos evaluados aleatoriamente pueden tener más o menos características que el mejor encontrado hasta el momento.

1.6.2.4. Búsqueda heurísticas y con metaheurísticas

Heurística

Se denomina heurística a la capacidad de un sistema para realizar de forma inmediata innovaciones positivas para sus fines. La capacidad heurística es un rasgo característico de los humanos, desde cuyo punto de vista puede describirse como el arte y la ciencia del descubrimiento y de la invención o de resolver problemas mediante la creatividad y el pensamiento lateral o pensamiento divergente. [13]

La etimología de heurística es la misma que la de la palabra eureka, cuya exclamación se atribuye a Arquímedes en un episodio tan famoso como apócrifo. "Heurística" proviene (de ευρισκειν, heuriskein, eureka, "encontrar"). [13]

La heurística trata de métodos exploratorios durante la resolución de problemas en los cuales las soluciones se descubren por la evaluación del progreso logrado en la búsqueda de un resultado final. Se suele usar como adjetivo, caracterizando las técnicas por las cuales se mejora en promedio el resultado de una tarea de solución de problemas. En informática, se utilizan algoritmos heurísticos para obtener soluciones subóptimas de problemas cuya optimización no es factible en tiempos polinómicos. [14]

Se habla de heurística para referirse a una técnica, método o procedimiento inteligente para realizar una tarea que no es producto de un riguroso análisis formal, sino de razonamiento común con conocimiento experto sobre la tarea. En especial, se aplica el término heurístico a un procedimiento si

trata de aportar soluciones a un problema con un buen rendimiento, en lo referente tanto a la calidad de las propuestas como a los recursos empleados, pero sin una garantía total de su optimalidad. [15]

Metaheurística

Una meta-heurística es un método heurístico para resolver un tipo de problema computacional general, usando los parámetros dados por el usuario sobre unos procedimientos genéricos y abstractos de una manera que se espera eficiente. Normalmente, estos procedimientos son heurísticos. El nombre combina el prefijo griego "meta" ("más allá", aquí con el sentido de "nivel superior") y "heurístico" (de ευρισκειν, heuriskein, "encontrar"). [16]

El término meta-heurísticas se obtiene de anteponer a heurística el sufijo meta que significa “más allá” o “a un nivel superior”. El término meta-heurística apareció por primera vez en el artículo seminal sobre búsqueda tabú de Fred Glover en 1986. [17]

En otras palabras, una meta-heurística puede verse como un marco de trabajo general para algoritmos que pueden aplicarse a diversos problemas de optimización. Durante los últimos años han aparecido una serie de métodos, denominados meta-heurísticos, cuya finalidad es la de encontrar buenas soluciones a problemas de optimización (lineal o no lineal y con o sin restricciones). Algunas metaheurísticas muy conocidas son:

- ✓ Optimización aleatoria.
- ✓ Ascensión de colinas con re-inicialización aleatoria.
- ✓ Enfriamiento simulado.
- ✓ Optimización basada en colonias de hormigas.
- ✓ Búsqueda Tabú.
- ✓ Algoritmos genéticos.
- ✓ GRASP.
- ✓ Inteligencia enjambre.
- ✓ Búsqueda por difusión estocástica.

1.7 Los métodos y técnicas de Búsqueda Heurística y Metaheurísticas.

Las búsquedas heurísticas en IA se refieren a algoritmos heurísticos y meta-heurísticos que normalmente encuentran buenas soluciones, aunque en ocasiones no hay pruebas de que la solución no pueda ser arbitrariamente errónea; o se ejecuta razonablemente rápido, aunque no existe tampoco prueba de que deba ser así. [18]

Estos tipos de búsquedas conllevan a soluciones aceptables; reducen el espacio de búsqueda y son capaces de determinar su proximidad a una solución y la calidad de la misma utilizando conocimiento a priori.

A menudo, pueden encontrarse instancias concretas del problema donde la heurística producirá resultados muy malos o se ejecutará muy lentamente. Aún así, estas instancias concretas pueden ser ignoradas porque no deberían ocurrir nunca en la práctica por ser de origen teórico, y el uso de heurísticas es muy común en el mundo real. Es difícil encontrar la heurística adecuada. Las heurísticas se tienen que definir dependiendo del problema que estamos tratando de resolver, debido a la complejidad de la reducción de la dimensionalidad, es difícil encontrar la heurística adecuada. Normalmente no se necesita una solución óptima sino una buena aproximación a esta.
Entre las metaheurísticas mas utilizadas se encuentran:

Los algoritmos genéticos («Genetic Algorithms»): fueron introducidos por Holland [18] para imitar algunos de los mecanismos que se observan en la evolución de las especies. Los mecanismos no son conocidos en profundidad pero sí algunas de sus características: la evolución ocurre en los cromosomas; un ser vivo da vida a otro mediante la decodificación de los cromosomas de sus progenitores, el cruce de los mismos, y la codificación de los nuevos cromosomas formando los descendientes; las mejores características de los progenitores se trasladan a los descendientes, mejorando progresivamente las generaciones.

Basándose en estas características, Holland creó un algoritmo que genera nuevas soluciones a partir de la unión de soluciones progenitoras utilizando operadores similares a los de la reproducción, sin necesidad de conocer el tipo de problema a resolver.

Los algoritmos de recocido simulado: («Simulated Annealing») fueron introducidos por Cerny [19] y Kirkpatrick [20] para la optimización de problemas combinatorios con mínimos locales. Utilizan técnicas de optimización no determinista: no buscan la mejor solución en el entorno de la solución actual sino que generan aleatoriamente una solución cercana y la aceptan como la mejor si tiene menor coste, o en caso contrario con una cierta probabilidad p; esta probabilidad de aceptación irá disminuyendo con el número de iteraciones y está relacionada con el empeoramiento del coste.

Estos algoritmos derivan de la analogía termodinámica con el proceso metalúrgico del recocido: cuando se enfría un metal fundido suficientemente despacio, tiende a solidificar en una estructura de mínima energía (equilibrio térmico); a medida que disminuye la temperatura, las moléculas tienen menos probabilidad de moverse de su nivel energético; la probabilidad de movimiento se ajusta a la función de Boltzmann.

Algoritmos de búsqueda no informada: Estos algoritmos no tienen en cuenta el coste de la solución durante la búsqueda. Su funcionamiento es sistemático, siguen un orden de visitas de nodos fijo, establecido por la estructura del espacio de búsqueda. Los principales ejemplos de estos algoritmos son el de anchura prioritaria, el de profundidad prioritaria y el de profundidad iterativa.

Entre los distintos métodos y técnicas heurísticas meta-heurísticas de resolución de problemas combinatorios surge, en un intento de dotar de "inteligencia" a los algoritmos de búsqueda local, el algoritmo de búsqueda tabú («tabu search») [21].

La búsqueda tabú, a diferencia de otros algoritmos basados en técnicas aleatorias de búsqueda de soluciones cercanas, se caracteriza porque utiliza una estrategia basada en el uso de estructuras de memoria para escapar de los óptimos locales, en los que se puede caer al "moverse" de una solución a otra por el espacio de soluciones. Al igual que en la búsqueda local, la búsqueda tabú selecciona de modo agresivo el mejor de los movimientos posibles en cada paso. Al contrario que sucede en la búsqueda local, se permiten movimientos a soluciones del entorno aunque se produzca un empeoramiento de la función objetivo, de manera que sea posible escapar de los óptimos locales y continuar estratégicamente la búsqueda de mejores soluciones.

Una ventaja importante que presentan las heurísticas y meta-heurísticas frente a las técnicas que buscan soluciones exactas es que, por lo general, permiten una mayor flexibilidad para el manejo de las características del problema. No suele ser complejo utilizar algoritmos heurísticos y meta-heurísticas que en lugar de funciones lineales utilicen no linealidades. Habitualmente las heurísticas proponen un conjunto de soluciones, ampliando de esta forma las posibilidades de elección del decisor, especialmente cuando existen factores no cuantificables que no han podido ser reflejados en el modelo pero deben ser tenidos en cuenta. Se trata de un enfoque diferente al utilizado por los sistemas expertos en el campo de la inteligencia artificial.

Su aplicación a los problemas de secuenciación de todo tipo es una finalidad típica y clásica. Es más, prácticamente todos ellos están basados en intentar resolver, de la mejor forma posible, problemas típicos de Organización de la Producción. Así, los problemas típicos de secuenciación de trabajos en máquinas, de equilibrado de líneas de montaje, de asignación de rutas, de planificación de la producción, etc. han sido, son y, casi con toda seguridad, serán el banco de pruebas de las más modernas técnicas de búsqueda de soluciones a problemas en los que, de entrada, se declina la posibilidad de encontrar la solución óptima.

Una ventaja importante que presentan las heurísticas frente a las técnicas que buscan soluciones exactas es que, por lo general, permiten una mayor flexibilidad para el manejo de las características del problema. No suele ser complejo utilizar algoritmos heurísticos que en lugar de funciones lineales utilicen no linealidades. Habitualmente las heurísticas proponen un conjunto de soluciones, ampliando de esta forma las posibilidades de elección del decisor, especialmente cuando existen factores no cuantificables que no han podido ser reflejados en el modelo pero deben ser tenidos en cuenta. Se trata de un enfoque diferente al utilizado por los sistemas expertos en el campo de la inteligencia artificial.

Los Algoritmos Genéticos (AG) y de enfriamiento simulado han sido aplicados al problema de selección de variables alcanzándose, en ambos casos, resultados prometedores.

1.8 Softwares vinculados a la Selección de Variables a nivel mundial.

RapidMiner (anteriormente Yale)

Es un software de código abierto para el análisis inteligente de datos, descubrimiento de conocimientos, minería de datos, aprendizaje automático, visualización, etc., con numerosas características y funciones para la selección de variables. Constituye además un entorno de aprendizaje automático y de extracción de datos para todo tipo experimentos. Permite que los experimentos sean realizados con un gran número de variables arbitrarias, las cuales se escriben en archivos XML que son fácilmente creados con la interfaz gráfica de RapidMiner. Ofrece más de 400 operadores para los principales procedimientos de aprendizaje de máquinas, incluidos los de entrada, salida, pre procesamiento de datos y visualización de los mismos.
Está escrito en el lenguaje de programación Java y, por tanto, pueden trabajar en todos los sistemas operativos populares. También integra todos los sistemas de aprendizaje y de atributo de los evaluadores Weka.

Cuenta con una licencia GNU GPL, Propietaria y Comercial.

Keel

Es un software para evaluar la evolución de los algoritmos de minería de datos y problemas de regresión entre ellos, clasificación, agrupamiento, patrón de la minería. Contiene una gran colección de algoritmos clásicos de extracción de conocimientos, técnicas de pre procesamiento (selección de instancias, selección de características, discretización, métodos de imputación de valores, etc.), Inteligencia Computacional de aprendizaje basado en algoritmos, incluido el estado evolutivo de algoritmos de aprendizaje basados en diferentes enfoques (Pittsburgh, Michigan y IRL) y modelos híbridos como sistemas difusos genéticos, redes neuronales evolutivas, etc. Nos permite realizar un análisis completo de cualquier modelo de aprendizaje en comparación con los existentes, incluido un módulo de prueba estadística para la comparación entre ellos.

El uso más común de esta herramienta para un investigador será la ejecución automatizada de los experimentos y el análisis estadístico de sus resultados. Esta herramienta no está diseñada para ofrecer un tiempo real del progreso de los algoritmos. Trabaja muy bien en ambiente distribuido de sistemas.

Fue diseñado con doble objetivo: la investigación y la educación. Cuenta con licencia comercial, lo que lo convierte Software propietario.

Weka

Es un paquete de software de Java para la extracción de conocimientos desde bases de datos incluye además una recopilación de algoritmos de aprendizaje automático para tareas de minería de datos. Este software ha sido desarrollado en la universidad de Waikato (Nueva Zelanda) bajo licencia GPL lo cual ha impulsado que sea una de las suites más utilizadas en el área en los últimos años.

Este ave da nombre a una extensa colección de algoritmos de Máquinas de conocimiento desarrollados por la universidad de Waikato (Nueva Zelanda) implementados en Java; útiles para ser aplicados sobre datos mediante las interfaces que ofrece o para embeberlos dentro de cualquier aplicación.

Además Weka contiene las herramientas necesarias para realizar transformaciones sobre los datos, tareas de clasificación, regresión, clustering, asociación y visualización. Está diseñado como una

herramienta orientada a la extensibilidad por lo que añadir nuevas funcionalidades es una tarea sencilla.

Sin embargo, y pese a todas las cualidades que Weka posee, tiene un gran defecto y éste es la escasa documentación orientada al usuario que tiene junto a una usabilidad bastante pobre, lo que la hace una herramienta difícil de comprender y manejar sin información adicional. La licencia de Weka es GPL, lo que significa que este programa es de libre distribución y difusión. Además, ya que Weka está programado en Java, es independiente de la arquitectura, ya que funciona en cualquier plataforma sobre la que haya una máquina virtual Java disponible.

Una de las propiedades más interesantes de este software, es su facilidad para añadir extensiones y modificar sus métodos.

2

CAPÍTULO

Métodos y Materiales.

Métodos

2.1 Los Algoritmos Genéticos.

Los algoritmos genéticos son procesos de búsqueda basados en los principios de la selección y la evolución natural. Las posibles soluciones a un problema son codificadas en forma de cadenas binarias, y la búsqueda se inicia con una población de posibles soluciones generadas aleatoriamente. [22]

Un algoritmo genético es un método de búsqueda dirigida basada en probabilidad. Bajo una condición muy débil (que el algoritmo mantenga elitismo, es decir, guarde siempre converge en probabilidad al mejor elemento de la población sin hacerle ningún cambio) se puede demostrar que el algoritmo al óptimo. En otras palabras, al aumentar el número de iteraciones, la probabilidad de tener el óptimo en la población tiende a 1. [23]

Los algoritmos genéticos son algoritmos matemáticos altamente paralelos que transforman un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta al modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud.[24]

2.1.1 Pasos para construir un Algoritmo Genético

- ✓ Diseñar una representación.
- ✓ Decidir cómo inicializar una población.
- ✓ Diseñar una forma de evaluar un individuo.
- ✓ Diseñar un operador de mutación adecuado.
- ✓ Diseñar un operador de cruce adecuado.
- ✓ Decidir cómo seleccionar los individuos para ser padres.
- ✓ Decidir cómo reemplazar a los individuos.
- ✓ Decidir la condición de parada.

2.1.2 Estrategia de Selección.

Debemos de garantizar que los mejores individuos tienen una mayor posibilidad de ser padres (reproducirse) frente a los individuos menos buenos.

Debemos de ser cuidadosos para dar una oportunidad de reproducirse a los individuos menos buenos. Éstos pueden incluir material genético útil en el proceso de reproducción.

Esta idea nos define la presión selectiva que determina en qué grado la reproducción está dirigida por los mejores individuos.

2.1.3 Algunos esquemas de selección

Selección por Torneo (TS): Escoge al individuo de mejor aptitud de entre Nts individuos seleccionados aleatoriamente $(Nts = 2,3,\ldots)$.

Orden Lineal (LR): La población se ordena en función de su aptitud y se asocia una probabilidad de selección a cada individuo que depende de su orden.

Selección Aleatoria (RS): Emparejamiento Variado Inverso (NAM): Un padre lo escoge aleatoriamente, para el otro selecciona *Nnam* padres y escoge el más lejano al primer $(Nnam = 3,5,\ldots)$. Está orientado a generar diversidad.

Selección por Ruleta: Se asigna una probabilidad de selección proporcional al valor de aptitud del cromosoma. (Modelo clásico)

2.1.3 Ventajas y desventajas de los algoritmos genéticos

La aplicación más común de los algoritmos genéticos ha sido la solución de problemas de optimización, en donde han mostrado ser muy eficientes y confiables debido a que:

No necesitan conocimientos específicos sobre el problema que intentan resolver.

Operan de forma simultánea con varias soluciones, en vez de trabajar de forma secuencial como las técnicas tradicionales.

Cuando se usan para problemas de optimización maximizar una función objetivo- resultan menos afectados por los máximos locales (falsas soluciones) que las técnicas tradicionales.

- ✓ Resulta sumamente fácil ejecutarlos en las modernas arquitecturas masivamente paralelas.

- ✓ Usan operadores probabilísticos, en vez de los típicos operadores determinísticos de las otras técnicas.
- ✓ Sin embargo, no todos los problemas pudieran ser apropiados para la técnica, y se recomienda en general tomar en cuenta las siguientes características del mismo antes de intentar usarla:
- ✓ Su espacio de búsqueda (posibles soluciones) debe estar delimitado dentro de un cierto rango.
- ✓ Debe poderse definir una función de aptitud que nos indique qué tan buena o mala es una cierta respuesta.
- ✓ Las soluciones deben codificarse de una forma que resulte relativamente fácil de implementar en la computadora.

Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en cierta medida los parámetros que se utilicen tamaño de la población, número de generaciones, etc.

Pueden converger prematuramente debido a una serie de problemas de diversa índole.

El primer punto es muy importante, y lo más recomendable es intentar resolver problemas que tengan espacios de búsqueda discretos aunque éstos sean muy grandes. Sin embargo, también podrá intentarse usar la técnica con espacios de búsqueda continuos, pero preferentemente cuando exista un rango de soluciones relativamente pequeño.

2.1.4 ¿Por qué utilizar Algoritmos Genéticos?

La razón del creciente interés por los algoritmos genéticos es que estos son un método global y robusto de búsqueda de las soluciones de problemas. La principal ventaja de estas características es el equilibrio alcanzado entre la eficiencia y eficacia para resolver diferentes y muy complejos problemas de grandes dimensiones.

- ✓ Trabajan con una codificación de un conjunto de parámetros, no con los parámetros mismos.

- ✓ Trabajan con un conjunto de puntos, no con un único punto y su entorno (su técnica de búsqueda es global.) Utilizan un subconjunto del espacio total, para obtener información sobre el universo de búsqueda, a través de las evaluaciones de la función a optimizar. Esas evaluaciones se emplean de forma eficiente para clasificar los subconjuntos de acuerdo con su idoneidad.

- ✓ No necesitan conocimientos específicos sobre el problema a resolver; es decir, no están sujetos a restricciones. Por ejemplo, se pueden aplicar a funciones no a continuas, lo cual les abre un amplio campo de aplicaciones que no podrían ser tratadas por los métodos tradicionales.

- ✓ Utilizan operadores probabilísticos, en vez de los típicos operadores determinísticos de las

técnicas tradicionales.

- ✓ Resulta sumamente fácil ejecutarlos en las modernas arquitecturas masivas en paralelo.
- ✓ Cuando se usan para problemas de optimización, resultan menos afectados por los máximos locales que las técnicas tradicionales (son métodos robustos).

2.1.5 Aplicaciones de los Algoritmos genéticos.

Entre las muchas aplicaciones que presentan los algoritmos genéticos esta su utilidad en:

- ✓ El control de procesos químicos.
- ✓ La optimización estructural.
- ✓ La Optimización combinatoria y en dominios reales.
- ✓ La Modelado e identificación de sistemas.
- ✓ La planificación y control.
- ✓ La Ingeniería.
- ✓ La Vida artificial.
- ✓ La planificación de sistemas de Producción.
- ✓ El aprendizaje, la Clasificación y la minería de datos.
- ✓ Internet y los Sistemas de Recuperación de Información.

2.2 Algoritmo de Enfriamiento Simulado

El enfriamiento simulado (Simulated Annealing (SA)) [25] [19] es una meta-heurística para problemas de optimización global que se basa en conceptos de la mecánica estadística. Fue propuesto por primera vez por Metrópolis [26] y usado en optimización combinatoria por Kirkpatrick [27]. Este método heurístico se basa en los conceptos descritos originalmente por la mecánica estadística que describe el proceso físico sufrido por un sólido al ser sometido a un baño térmico.

Se sabe en ingeniería, que una manera de encontrar los estados de energía de sistemas complejos, tales como sólidos, consiste en utilizar la técnica de enfriamiento, en la que el sistema se calienta primero a una temperatura en la que sus granos deformados recristalizan para producir nuevos granos, y luego se enfría suavemente y de esta manera, cada vez que se baja la temperatura, las partículas se

reacomodan en estados de más baja energía hasta que se obtiene un sólido con sus partículas acomodadas conforme a una estructura de cristal (estado fundamental). En la fase de enfriamiento, para cada valor de la temperatura, debe permitirse que el sistema alcance su equilibrio térmico [28].

De forma análoga, en el algoritmo de enfriamiento simulado los estados del sistema corresponden a las soluciones del problema, la energía de los estados a los criterios de evaluación de la calidad de la solución (generalmente se utiliza la función objetivo), estado fundamental a la solución óptima del problema, los estados meta estables serán los equivalentes a los óptimos locales, y la temperatura está asociada a una variable de control. "El éxito del Enfriamiento Simulado se basa en la escogencia de una buena temperatura inicial y una adecuada velocidad de enfriamiento" [29]

"La característica principal de este algoritmo es que al buscar una nueva solución S_{n+1} dada una solución S_n, acepta en ocasiones una de inferior calidad a la de Sn por medio de una función probabilística la cual depende del parámetro variable de temperatura y de la calidad ofrecida por las dos soluciones S_n y S_{n+1}. Mientras más bajo sea el parámetro de temperatura, menor será la probabilidad de aceptar una solución peor, y viceversa" [29].

El método del Recocido Simulado (RS) es una poderosa herramienta de búsqueda estocástica que se ha hecho muy popular dado el amplio espectro de problemas que puede resolver. En particular en el área de la optimización combinatoria y la selección de variables o de características.

2.2.1 Esqueleto de un TA (Algoritmos de umbral)

El algoritmo de recocido simulado (Simulated Annealing Algorithm - SAA) pertenece una clase de Algoritmos de búsqueda global (Global Search Algorithms – GSA) comúnmente llamada Algoritmos de Umbral (Threshold Algorithm - TA). Hay dos razones por las cuales los TA resultan interesantes dentro de los GSA:

- ✓ Parecen andar bien en una amplia gama de problemas reales (prácticos)
- ✓ Algunos TA, como el SAA, tienen características que permiten hacer un análisis de la convergencia.

Sea ('S', 'c') una instancia de un problema de optimización combinatoria, donde:

- ✓ 'S' es el conjunto de soluciones factibles
- ✓ 'c' es la función costo (a valores reales positivos)

El problema es hallar una ‘i’ en ‘S’ que minimice ‘c’.
Para implementar un TA son necesarios además:

- ✓ Una función entorno ‘N’ de ‘S’ en partes de ‘S’.
- ✓ Una sucesión ‘t_k‘ (los llamados threshold)

La manera de elegir los t_k y el criterio de aceptación de una nueva solución definen 3 tipos de TA:
Dado *i* en *S* en la iteración *k*
Genero *j* en *N (i)*
Utilizo los valores $c(j) - c(i)$ y t_k para decidir aceptar o no la solución *j*.

Mejora Contínua (Local Search Improvement): $t_k = 0$ (para todo *k*)

Si $c(j) - c(i) < t_k = 0$ entonces acepto *j*

Umbral de Aceptación (Threshold Accepting): se fija la sucesión t_k tal que $t_k = t_{k+1}, t_k > 0$, y t_k tiende a 0 cuando k tiende a infinito.

Si $c(j) - c(i) < t_k$ entonces acepto j

En este caso, todas las soluciones que disminuyen el costo son aceptadas, y las que incrementan el costo son aceptadas en forma limitada. A medida que aumenta k (progresa el algoritmo) solo se aceptan incrementos pequeños, hasta que eventualmente solo se aceptan mejoras.

2.2.2 Aplicaciones del algoritmo Enfriamiento Simulado.

Entre las muchas aplicaciones que este algoritmo se encuentran:

- ✓ Su utilización en varios tipos de industrias (metalúrgica, básica, etc.) para obtener materiales mas resistentes.
- ✓ Su utilización en la simulación de procesos para mejorar las cualidades de un material específico.

2.3 Medidas de Evaluación

En todo proceso de selección de características, hay algún momento en que es necesario valorar la utilidad de estas en la resolución del problema de aprendizaje a abordar. En la mayor parte de los métodos, puede hacerse la valoración de forma independiente al proceso de búsqueda.

Hay dos tipos de aproximaciones principales al problema de la valoración de relevancia de las características. La primera es valorar cada una de las características de forma independiente. La ventaja de este tipo de medidas es que suelen ser medidas bastante simples y, como solo se pueden hacer tantas valoraciones como características tenga el conjunto de datos, apoyarse en estas medidas es muy rápido. Como contrapartida, el inconveniente principal es que estas medidas no aportan ninguna información sobre la posible redundancia que haya entre características. La segunda aproximación a la valoración de características es trabajar con conjuntos de características. En este caso, al valorar conjuntos completos, se puede obtener información sobre las posibles interrelaciones entre características y, de esta forma, averiguar por ejemplo si la inclusión de una característica aporta algo o es completamente redundante a las de un conjunto dado. En cambio, el inconveniente es que valorar todos los posibles subconjuntos de características normalmente no es factible.

2.3.1 Medidas de evaluación sobre variables individuales.

Entre las muchas medidas de evaluación de características individuales se describen las más comúnmente usadas.

Ganancia de información (información mutua)

De la teoría de la información, se usa la cantidad de información que aporta una característica sobre la clase a predecir, para valorar la relevancia de dicha característica. Quinlan utilizaba la información mutua para elegir las características que dividirán nodos en generación de árboles [30].

Gain Ratio

La medida anterior de ganancia de información favorece a las características con muchos valores. Puede ocurrir que esta sobre-estimación no sea un comportamiento deseable y para evitarlo se puede usar como medida el ratio entre la ganancia de información y la entropía de la característica. [31]

Índice de Gini

El índice de Gini (Gini índex) toma su nombre del estadístico italiano Conrado Gini.

El índice, más conocido como una medida de la desigualdad usada en economía [32], fue introducido como medida para la generación de árboles de clasificación y regresión por Breiman [33]. La medida se puede interpretar como la probabilidad de que dos ejemplos elegidos aleatoriamente tengan una clase diferente.

ReliefF

Originalmente, Kira y Rendell [34] propusieron Relief como un método de selección de características. Estaba basado en una valoración novedosa de las características y en la elección de aquellas que obtuviesen una valoración mayor que un umbral dado. La valoración original de relevancia de las características solo estaba definida para problemas lógicos (cuyo resultado solo puede ser verdadero o falso) pero, posteriormente, Kononenko desarrollo extensiones [35] que pueden trabajar con problemas de clasificación y tolerar valores nulos.

2.3.2 Medidas de evaluación sobre conjuntos variables.

Las medidas sobre conjuntos de características son funciones que, dado un conjunto de datos de entrenamiento $T \in R$, denominamos R a todos los posibles conjuntos de entrenamiento y un subconjunto de características $(S \subset P(F))$, devuelven una valoración de la relevancia de esas características. El resultado será un numero real, normalmente dentro de un intervalo, como $[0,1]$ o $[-1,1]$, pero también puede ser un resultado discreto, por ejemplo en $\{0,1\}$, representando un valor booleano que indique si el conjunto es aceptable o no como resultado de la selección.

Hay una gran variedad de medidas de relevancia para conjuntos de características. Algunas trabajan tanto con características discretas como continuas, otras solo aceptan uno de los dos tipos. Los algoritmos que las calculan pueden ser exactos o aproximados, determinísticos o no. Las medidas cumplen o no diversas propiedades como la monotonía o la invariabilidad a transformaciones lineales.

Una categorización de las medidas basada en el tipo de cualidad que mide esta propuesta en [6]. Las categorías identificadas son: medidas de distancia, medidas de información, medidas de dependencia, medidas de consistencia y las basadas en el porcentaje de acierto del algoritmo de aprendizaje. Se describe a continuación las medidas más importantes de cada una de las categorías.

2.3.2.1 Medidas basadas en consistencia

Para poder predecir correctamente la clase asociada a las instancias de un conjunto de datos, es necesario que este sea consistente. Un conjunto de datos se considera consistente siempre que en el no haya ningún par de instancias que, perteneciendo a clases distintas, tengan los mismos valores en

todas sus características. Si en un conjunto de datos se eliminan algunas características, dejando solo las seleccionadas, habrá menos valores que diferencien las instancias y, por tanto, podrán aparecer más casos de inconsistencia. La idea que persiguen las medidas basadas en consistencia es valorar el nivel de consistencia del conjunto de datos que tiene únicamente las características seleccionadas.

Como siempre que se aumenta el número de características, aumenta el número de hipótesis consistentes que se pueden definir, el requisito de presentar consistencia suele acompañarse con el de tener un numero reducido de características. En cualquier caso, la búsqueda de un conjunto de características pequeño es un objetivo común de todos los métodos de selección de características. Así, esta estrategia no es una particularidad exclusiva de los métodos basados en medidas de consistencia, sino que también es aplicada en algoritmos de búsqueda que se usan con otros tipos de medidas.

Hay otros métodos basados en consistencia que, aunque no definen medidas específicas, puede ser interesante tener en cuenta. Schlimmer [36] describe un algoritmo para deducir determinaciones lógicas usando el menor numero posible de características, que, de hecho, es un algoritmo de selección de características inmerso, también existen otras medidas como las de consistencia básica [37], la consistencia de Liu[38] y la consistencia de la teoría de Rougth Sets [39].

2.3.2.2 Medidas basadas en la Teoría de la Información

Las medidas de este apartado se basan en la teoría de la información de Shannon [40]. Midiendo la información que aportan las características sobre la clase podemos saber cuales son más informativas, siendo estas, desde el punto de vista de la teoría de la información, las más apropiadas para la clasificación. Muchos algoritmos de aprendizaje se basan en principios de la teoría de la información, lo que, además de indicar que el uso de estas medidas es prometedor, nos lleva a pensar que habrá una sinergia positiva entre los métodos de selección de características que usen estas medidas y los algoritmos de aprendizaje basados en teoría de información. Algunas de estas medidas como la Incertidumbre Simétrica y la Información Mutua se describen a continuación.

Información Mutua

La teoría de la información establece una forma básica de medir la información que aporta el conocimiento de los valores que toman una o más variables sobre otra. Sea C la variable aleatoria que define la clase de un problema de clasificación. La entropía de C viene dada por:

$$H(C) = -\sum_{c \in C} p(c) \log p(c) \qquad \text{(I)}$$

El objetivo del algoritmo de aprendizaje es reducir la incertidumbre sobre el valor de la clase. Para ello, el conjunto de características seleccionadas S aporta la cantidad de información dada por:

$$I(C,S) = H(C) - H(C \mid S) \qquad \text{(II)}$$

Lo ideal seria encontrar el menor conjunto de características que determine completamente C, esto es $I(C,S) = H(C)$, pero no siempre es posible. Esta medida cumple la propiedad de la monotonía. Al añadir una característica más, esta siempre aportara algo de información o, en el peor de los casos, nada, pero nunca reducirá la información que aportan las características ya seleccionadas.

Los algoritmos de búsqueda que usen esta medida deberán tener en cuenta la propiedad de monotonía pues en un conjunto de datos no completamente determinado se tendera seleccionar todas las características. Así se debe buscar un conjunto, que aporte mucha información, pero con un numero reducido de características.

Incertidumbre Simétrica

La medida de información mutua es simétrica, de (II) se tiene:

$$I(C,S) = H(C) - H(C \mid S) = H(S) - H(S \mid C) = I(S,C) \qquad \text{(III)}$$

Por otra parte, la medida de información mutua tiende a dar mayor valor a las características con más valores. Si queremos que todas las características sean valoradas equitativamente, podemos usar:

$$U(C,S) = \frac{H(C) - H(C \mid S)}{H(S)} \qquad \text{(IV)}$$

Sin embargo, con esta definición, la medida dejaría de ser simétrica. Por esta razón, se define la medida de incertidumbre simétrica [41] como:

$$SU(C,S) = 2 \cdot \frac{H(C) - H(C \mid S)}{H(S) + H(C)} \qquad \text{(V)}$$

Intuitivamente, esta medida puede interpretarse como la razón (ratio) entre la cantidad de información que aportan las características seleccionadas y la cantidad de información total que contienen y, por tanto, la información que podrá aportar. Al ser un ratio, cuyo denominador puede crecer más rápido que el numerador al incluir características, esta medida no cumple la propiedad de monotonía.

MDL (Longitud Mínima de Descripción)

Las medidas con la propiedad de monotonía tienen el inconveniente de no indicar directamente que características son completamente irrelevantes, pues al añadir una característica más, esta incrementa la medida aunque sea muy ligeramente. La medida MDL [42] pretende resolver este problema aplicando el criterio MDLC (longitud mínima de la descripción de un modelo). Este criterio sostiene que entre varios modelos de ajuste se debe elegir el que tenga una descripción mas corta.

Se puede ver como el principio de la navaja de Ockham aplicado a la teoría de la información. De esta forma, se tiene en cuenta que cuantas más características hay, más complejo es el modelo, permitiéndose discernir cuando ya no merece la pena usar más características.

2.3.2.3 Medidas Basadas en la Distancia

Estas medidas valoran las distancias que hay entre las distribuciones de probabilidad de cada clase. La idea detrás de estas medidas es que aquellas características que hagan mayor la distancia entre las distribuciones las separaran mejor y, por tanto, deben permitir hacer mejor la clasificación [43].

2.3.2.4 Medidas Basadas en la Dependencia

CFS (Correlación Basada en Selección de Características)

CFS es una heurística para evaluar subconjuntos de atributos pero que al mismo tiempo, tiene en cuenta el valor predictivo de cada elemento del subconjunto sobre la clase y la intercorrelación entre estos. Así la hipótesis que plantea esta heurística es la siguiente:

Buenos subconjuntos de atributos, poseen una alta relación con el atributo clase (alta cohesión) y una baja relación entre estos (bajo acoplamiento)

Esta medida de evaluación se formaliza en la siguiente ecuación:

$$CFS = \frac{k \cdot \bar{r}_{ic}}{\sqrt{k + k(k-1) \cdot \bar{r}_{ij}}} \qquad \text{(VI)}$$

Donde:

- ✓ $\bar{r}_{ic}$ representa la correlación promedio de los elementos del subconjunto con respecto a la variable de salida (correlación intra-clase).
- ✓ $\bar{r}_{ij}$ representa la correlación promedio de los elementos del subconjunto (correlación inter-clase).
- ✓ k representa la cardinalidad del subconjunto.

Esta heurística es aplicable tanto a modelos lineales como no lineales, donde para el primer caso su valor depende del cálculo de los coeficientes de Pearson **"r"**. Esta medida permite eliminar elementos redundantes y poco significativos. Es inversamente proporcional a la cardinalidad del subconjunto, si aumenta el número de elementos entonces disminuye el valor (mérito) de CFS. En el siguiente capitulo abordaremos mas sobre esta medida de evaluación.

Dentro de esta investigación se han seleccionado Los Algoritmos Genéticos y de Enfriamiento Simulado como algoritmos de propósito general para lograr una buena reducción de la dimensionalidad.

Materiales

2.4 Metodologías y herramientas para el desarrollo del sistema.

2.4.1 Plataforma de Desarrollo y Lenguaje de Programación.

En cuanto a plataforma escogida se optó por jdk version 1.5.0_10.

Como lenguaje de programación se escogió Java el cual es un lenguaje de programación orientado a objetos desarrollado por Sun Mycrosystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o

memoria.

Entre sus características fundamentales se encuentran:

- ✓ Usa la metodología de la programación orientada a objetos.
- ✓ Permite la ejecución de un mismo programa en múltiples sistemas operativos.
- ✓ Incluye por defecto soporte para trabajo en red.
- ✓ Esta diseñado para ejecutar código en sistemas remotos de forma segura.
- ✓ Es fácil de usar y toma lo mejor de otros lenguajes orientados a objetos, como C++.
- ✓ Es independiente de la plataforma en la que se ejecuta, esto significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware, tal como reza el axioma de Java, ''write once, run everywhere''.("escrito una vez, corre donde sea")

2.4.2 Entorno de Desarrollo.

Se utilizó Eclipse como entorno de desarrollo, es una plataforma de software de código abierto, extensible. Esta plataforma, típicamente ha sido usada para desarrollar entornos integrados de desarrollo, como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se embarca como parte de Eclipse y que son usados también para desarrollar este entorno. Sin embargo, también se puede usar para otros tipos de aplicaciones cliente además de ser un entorno de desarrollo integrado que ofrece el control del editor de códigos, del compilador y del depurador desde una única interfaz de usuario. Este entorno de desarrollo integrado ofrece, el control del editor de código, del compilador y del depurador desde una única interfaz de usuario. Además Eclipse es una plataforma universal para integrar herramientas de desarrollo, basada en plug-ins.

2.4.3 Herramienta Case (Computer Aided Software Engineering).

La herramienta Case utilizada fue VISUAL PARADIGM la cual pertenece a una compañía del mismo nombre y se encuentra entre las principales compañías de herramientas CASE. Tiene disponible distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community (que es gratuita). La compañía facilita licencias especiales para fines académicos.

Esta herramienta CASE utiliza "UML": como lenguaje de modelaje. Soporta hasta la fecha UML 2.1 completo y BPMN. Permite además, que a partir de un modelo relacional en Sql Server, MySql, etc. es capaz de desplegar todas las clases asociadas a las tablas.

Para gestionar la persistencia y el mapeo de estas clases con la base de datos utiliza Hibernate para Java y NHibernate en el caso de un proyecto .Net. Además, la herramienta es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como Web o .Pdf, y permite control de versiones, ingeniería inversa, generación de código, importación desde Rational Rose, exportación/importación XMI, generador de informes, editor de figuras, integración con MS Visio, plug-in, integración IDE con Visual Studio, IntelliJ IDEA, Eclipse, NetBeans y otros. Entre sus nuevas características se incluyen el modelado colaborativo con CVS y Subversion, interoperabilidad con modelos UML2 a través de XMI. Cabe destacar igualmente su robustez, usabilidad y portabilidad.

Se integra con las siguientes herramientas Java:

- ✓ Eclipse/IBM WebSphere
- ✓ JBuilder
- ✓ NetBeans IDE
- ✓ Oracle JDeveloper
- ✓ BEA Weblogic

En definitiva, Visual Paradigm es una herramienta muy a tener en cuenta a la hora de ponerse manos a la obra con un proyecto importante. Su Licencia: Gratuita y Comercial.

2.4.4 Lenguaje de modelado.

El lenguaje de modelado utilizado fue UML ("Unified Modeling Language"), el cual está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo.

Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aún cuando todavía no es un estándar oficial, está respaldado por el OMG (Object Management Group). Mediante UML es posible establecer la serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código.

Además es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como

expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

La decisión de utilizar UML como notación para el desarrollo del software esta apoyada además en que se ha convertido en un estándar con muchas características favorable como las siguientes:

- ✓ Permite modelar sistemas utilizando técnicas orientadas a objetos (OO).
- ✓ Permite especificar todas las decisiones de análisis y diseño, construyéndose así modelos precisos, no ambiguos y completos.
- ✓ Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- ✓ Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- ✓ A pesar de tener gran expresividad esta notación es fácil de aprender.
- ✓ UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

2.4.5 Gestor de Base de Datos utilizado.

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB—desde enero de 2008 es una subsidiaria de Sun Microsystems— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero las empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario que proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es propiedad y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código*.*

Aplicaciones

MySQL es muy en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. MySQL es una base de datos muy rápida en la lectura

cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación.

CAPÍTULO 3

Resultados y Discusión.

Capítulo 3 Resultados y Discusión

3.1 Modelo conceptual del sistema

Un modelo conceptual explica los conceptos más significativos en un dominio del problema, identificando los atributos y las asociaciones, y es la herramienta más importante del análisis orientado a objetos. Los casos de uso son una herramienta para el análisis de requerimientos, pero realmente no están orientados a objetos. Un modelo conceptual representa el funcionamiento real del sistema, sin componentes del software. En UML se representa mediante un grupo de diagramas de estructura estática donde no se define ninguna operación. En estos diagramas se muestran conceptos (objetos), asociaciones entre conceptos (relaciones) y atributos de conceptos (atributos).

Este Modelo presenta cualidades significativas para lograr un buen entendimiento del problema entre las que se que se encuentran:

- Claridad y Simplicidad: Significación no ambigua.

- Coherencia: Ausencia de contradicciones o confusión.

- Completitud: Sin buscar la exhaustividad, se representa lo esencial de los fenómenos.

- Fidelidad: Representación sin desviaciones y sin deformaciones.

- No Redundancia: Sólo se representan elementos estrictamente necesarios, y únicamente una vez.

Por todas las bondades que este modelo presenta y la importancia de este en el entendimiento del flujo de eventos dentro del sistema implementado es que se tomó la decisión de incluirlo en este trabajo.

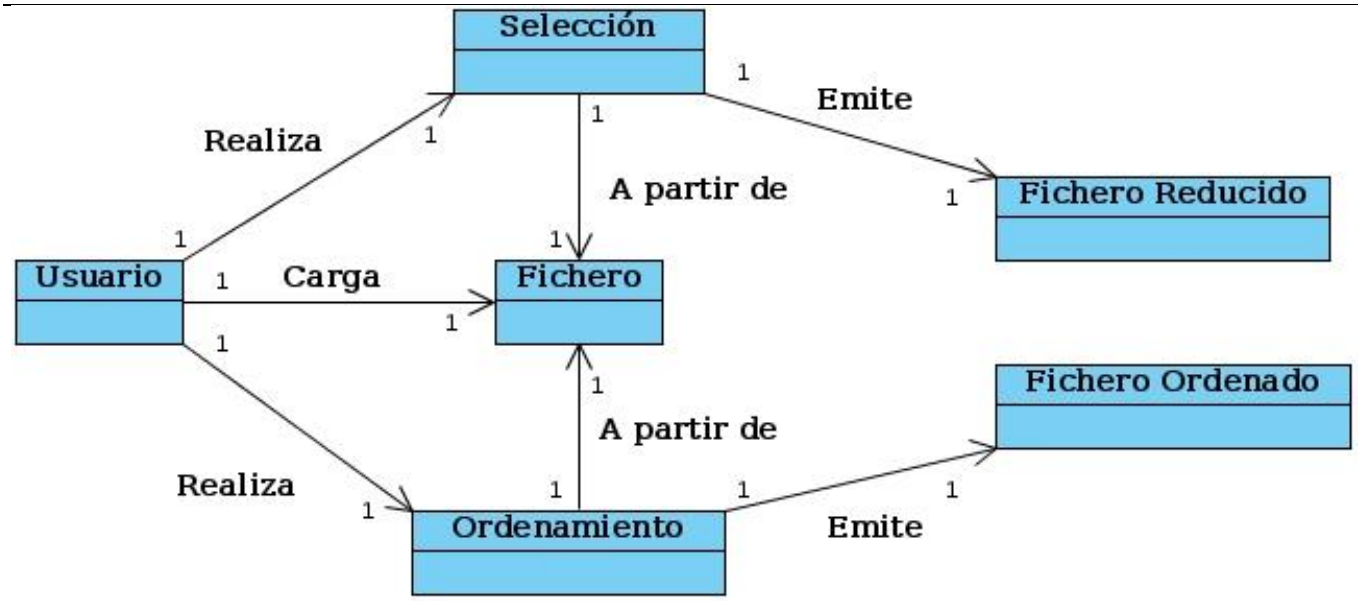


Figura 3.1 Modelo Conceptual

3.2 Diagrama de clases y Patrones utilizados.

3.2.1 Diagrama de clases.

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases y las relaciones entre ellas. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.

También estos diagramas son los más comunes en el modelado de sistemas orientados a objetos y no solo para la visualización, especificación y documentación del modelo estructural, sino también para la construcción de sistemas ejecutables y para la realización de Ingeniería hacia adelante e ingeniería inversa.

A continuación se muestra el diagrama de clases del diseño (**figura 3.2**.) utilizado en nuestra aplicación.

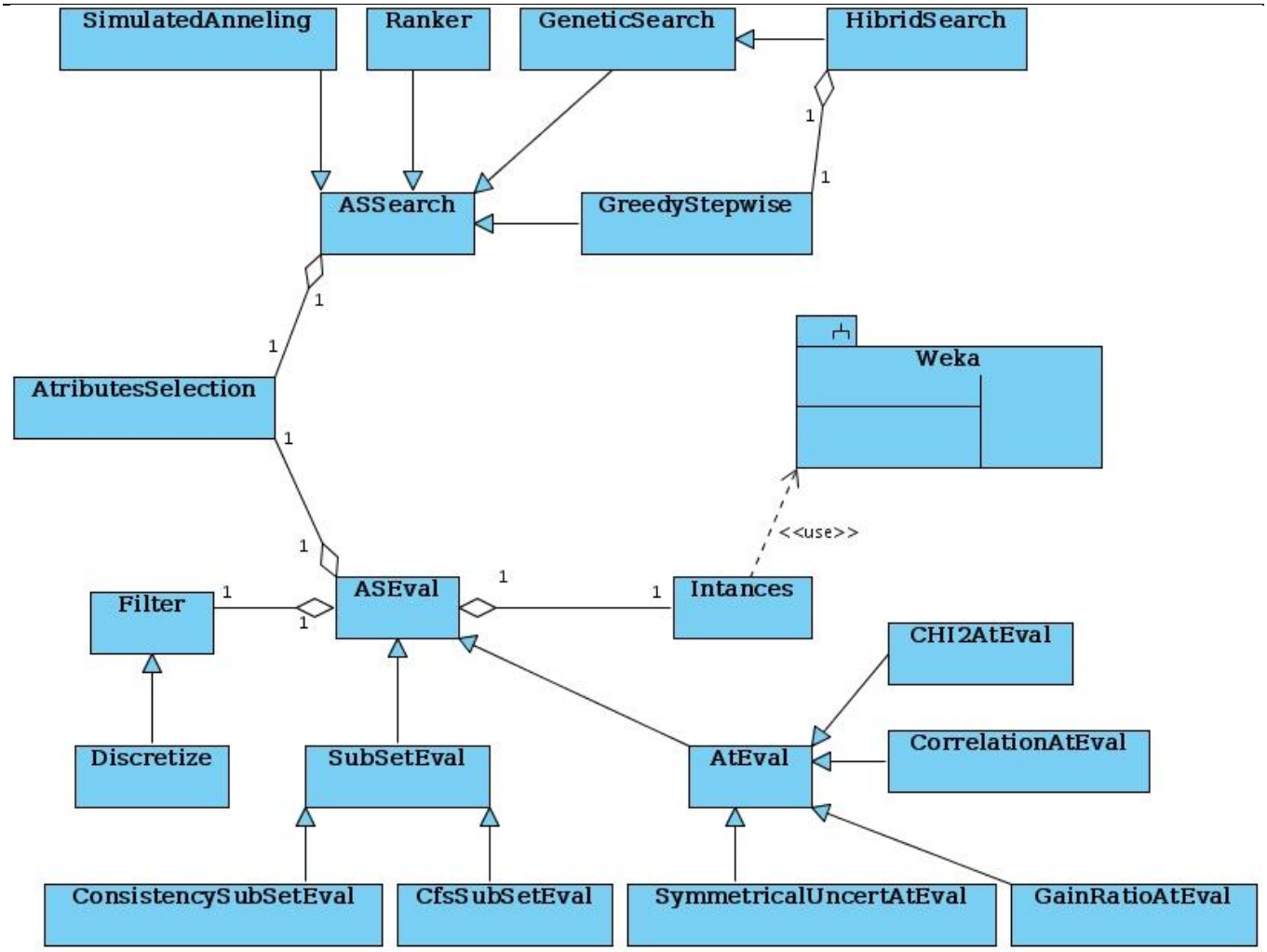


Figura 3.2 Diagrama de Clases del diseño.

El anterior diagrama de clases del diseño muestra las clases más importantes de la aplicación. En ella se presenta como clase controladora la AtributesSelection, la cual se encarga de ejecutar las operaciones para la selección de las variables o reducción de espacio muestral. Esta clase contendrá un método de búsqueda y una medida de evaluación. La clase ASSearch es padre de todas las clases que contiene los métodos de búsqueda implementados como se muestra en el diagrama y la clase ASE val contiene es la clase padre de las clases donde se encuentran las medidas de evaluación implementadas. En el diagrama se muestra también el subsistema Weka, el cual es utilizado por la clase Intances obteniéndose de este algunos de sus componentes. Consideramos ocultar los atributos

y métodos correspondientes a las clases del diagrama, pues son muchos por cada clase y esto afectaría la comprensión del diagrama, al hacerse este muy grande.

3.2.2 Patrones utilizados en la Resolución del Problema.

3.2.2.1 Patrón

En términos generales, un patrón es un conjunto de información que proporciona respuesta a un conjunto de problemas similares, es decir, un patrón es una solución a un problema en un contexto, donde:

- ✓ Contexto son las situaciones recurrentes a las que es posible aplicar el patrón.
- ✓ Problema es el conjunto de metas y restricciones que se dan en ese contexto.
- ✓ Solución es el diseño a aplicar para conseguir las metas dentro de las restricciones.

En resumen un patrón es una solución a un problema de diseño no trivial que es efectiva y reusable.

3.2.3.2 Importancia de utilizar Patrones

Entre las principales importancias de los patrones podemos encontrar las siguientes:

- ✓ Con estos se realiza una producción de software más resistente al cambio.
- ✓ Establecen problemas Pareja-Solución.
- ✓ Ayudan a especificar interfaces.
- ✓ Ayudan a la reutilización del Código.
- ✓ Ayudan al uso de documentación estándar.

3.2.3.3 Patrones GRASP

Patrones de Software para la asignación General de Responsabilidad (General Responsability Assignment Software Patterns).

Los patrones GRASP describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el mismo de una forma sistemática, racional y explicable.

Las responsabilidades están relacionadas con las obligaciones de un objeto en cuanto a su comportamiento.

Básicamente, estas responsabilidades son de los siguientes dos tipos:

Conocer:

- ✓ Conocer los datos privados encapsulados.
- ✓ Conocer los objetos relacionados.
- ✓ Conocer las cosas que puede derivar o calcular.

Hacer:

- ✓ Hacer algo él mismo, como crear un objeto o hacer un cálculo.
- ✓ Iniciar una acción en otros objetos.
- ✓ Controlar y coordinar actividades en otros objetos.

Se pueden destacar 5 patrones **GRASP** Principales que son:

- ✓ **Experto**
- ✓ **Creador**
- ✓ **Alta cohesión**
- ✓ **Bajo Acoplamiento.**
- ✓ **Controlador**

Y 4 patrones adicionales los cuales son:

- ✓ **Polimorfismo**
- ✓ **Indirección**
- ✓ **No hables con extraños**
- ✓ **Fabricación Pura**

Los utilizados en nuestro trabajo fueron:

Experto

La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Hay que tener en cuenta que esto es aplicable mientras estemos considerando los mismos aspectos del sistema:

- ✓ **Lógica de negocio**
- ✓ **Persistencia a la base de datos**
- ✓ **Interfaz de usuario**

Se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento (Bajo Acoplamiento es un patrón GRASP que examinaremos mas adelante).

El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases "sencillas" y más cohesivas que son más fáciles de comprender y de mantener. Así se brinda soporte a una alta cohesión.

Este patrón se ve evidenciado en la clase AttributeSelection, puesto que al poseer un objeto ASSearch como método de búsqueda y un objeto ASEval como medida de evaluación, posee la información necesaria para encontrar subconjunto de atributos solución. Así a esta clase se le asigna la responsabilidad de encontrar subconjuntos de atributos o grupos de subconjuntos de atributos.

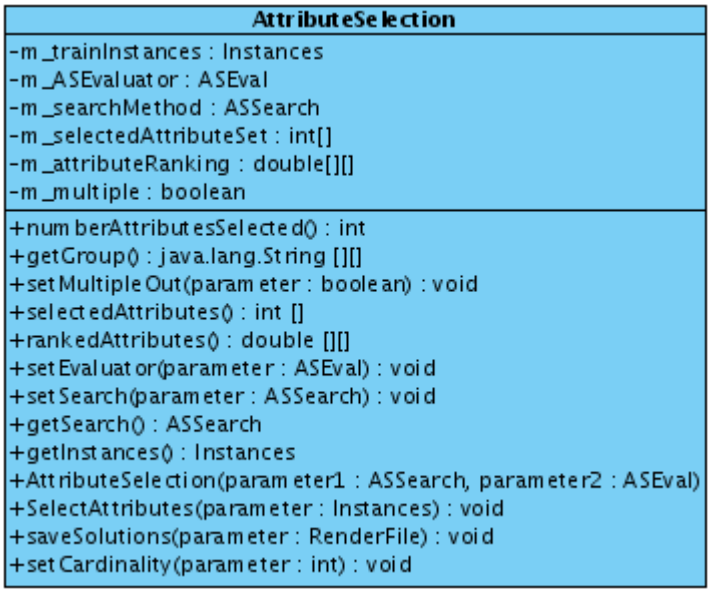


Figura 3.6 Ejemplo de una clase (AttributeSelection) Experta

Creador

Se asigna la responsabilidad de que una clase B cree un Objeto de la clase A solamente cuando:

- ✓ **B contiene a A**
- ✓ **B es una agregación (o composición) de A**
- ✓ **B almacena a A**
- ✓ **B tiene los datos de inicialización de A (datos que requiere su constructor)**
- ✓ **B usa a A.**

Con este patrón se brinda soporte a un bajo acoplamiento (que describiremos más adelante), lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. Es probable que el acoplamiento no aumente, pues la clase creada tiende a ser visible a la clase creador, debido a las asociaciones actuales que llevaron a elegirla como el parámetro adecuado.

La clase ASSearch contiene un objeto Stats; por ello, el patrón Creador sugiere que ASSearch es idónea para asumir la responsabilidad de crear las instancias de la clase Stats.

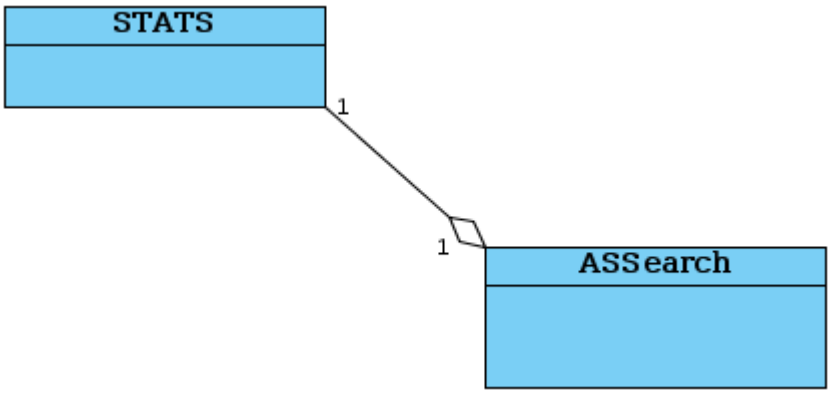


Figura 3.7 Ejemplo de la aplicación del Patrón Creador

Alta cohesión

Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable. En la perspectiva del diseño orientado a objetos, la cohesión (o más exactamente, la cohesión funcional) es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un mayor trabajo.

Debido a las funcionalidades que implementa la aplicación, es importante definir varios métodos de búsqueda y medidas de evaluación. Si desarrollamos una clase que disponga de todos los métodos de búsqueda y atributos necesarios para llevarla a cabo, constituiría para la aplicación un objeto saturado y se tornaría difícil su mantenimiento y reutilización. Lo mismo pasaría si se implementa una clase única, que implemente todos los métodos de evaluación, por ejemplo CHI2, Ganancia de Información, Incertidumbre Simétrica y demás, tal diseño identifica la tendencia de una baja cohesión, asignando demasiadas responsabilidades a una misma clase, lo cual hace que tenga operaciones funcionales muy heterogéneas. Así, para aumentar la cohesión y la fácil reutilización y mantenimiento de la aplicación, se crean entidades específicas para cada Algoritmo de evaluación. Ejemplo:

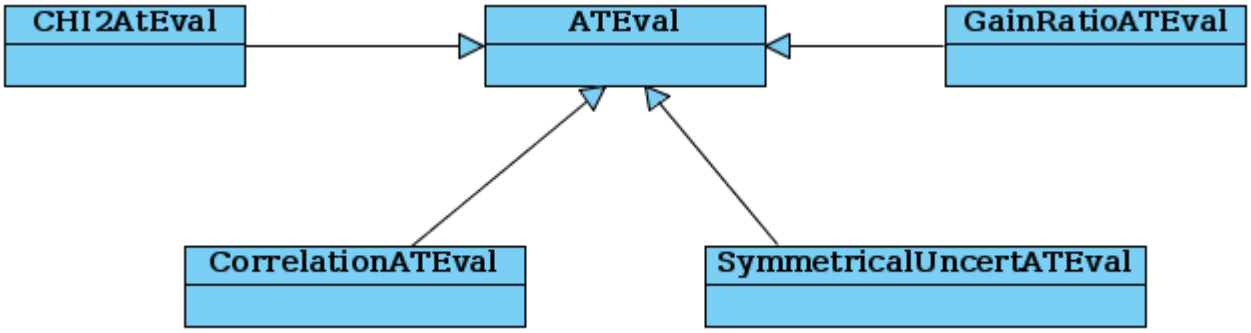


Figura 3.8 Ejemplo de la aplicación del Patrón Alta Cohesión

Bajo acoplamiento

El acoplamiento es una medida de la fuerza con que una clase esta conectada a otras, con que las conoce y recurre a ellas. El Bajo Acoplamiento estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento.

El Bajo Acoplamiento soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. No existe una medida absoluta de cuando el acoplamiento es excesivo. Lo importante es que el diseñador pueda determinar el grado actual de acoplamiento y si surgirán problemas en caso de incrementarlo. En términos generales, han de tener escaso acoplamiento las clases muy genéricas y con grandes probabilidades de reutilización.

Puesto que la entidad AtributeSelection tiene la responsabilidad de conocer los subconjuntos generados durante el proceso de selección de atributos, ésta incluye la clase ASSearch. La clase ASSearch terminará acoplándose al conocimiento de un objeto de la clase Stats para almacenar soluciones durante su ejecución, así para que el experto AtributeSelection pueda conocer la información de los subconjuntos generados, este no necesariamente debe acoplarse a la clase Stats, pues su acoplamiento con ASSearch hace que disponga de información necesaria para hallar las soluciones.

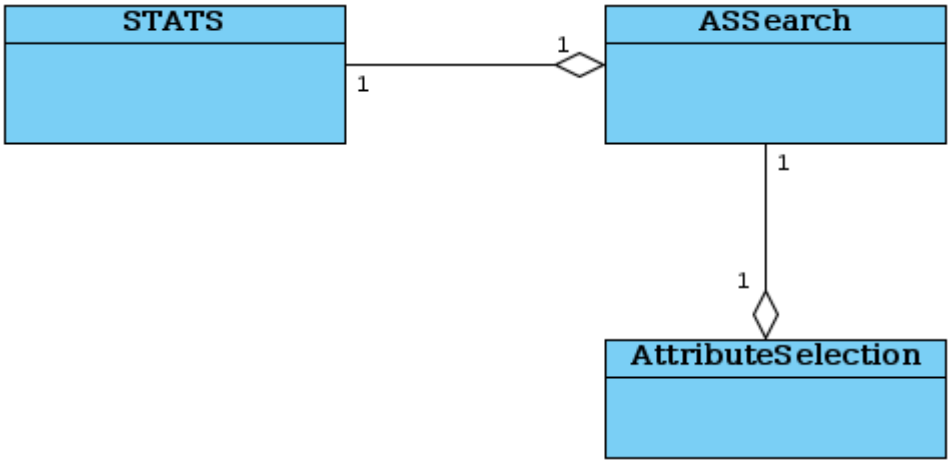


Figura 3.9 Ejemplo de la aplicación del Patrón bajo Acoplamiento

Polimorfismo

El polimorfismo significa “asignar el mismo nombre a servicios en varios objetos”, cuando los servicios se parecen o están relacionados entre sí.

El uso del patrón Polimorfismo está acorde al espíritu del patrón Experto. Si podemos caracterizar Experto como el patrón fundamental táctico, Polimorfismo será el más importante patrón estratégico en el diseño orientado a objetos. Entre sus Beneficios uno de los más importantes es la facilidad para agregar las futuras extensiones que requieren las variaciones imprevistas.

Este patrón se evidencia con facilidad en el diseño, especialmente en las clases que derivan de ASSearch, que tienen funcionalidades independientes y bien centralizadas, todas estas entidades soportan el método:

```
int [ ] search (ASEval ASEvaluator, Instances data) trhows Exception
```

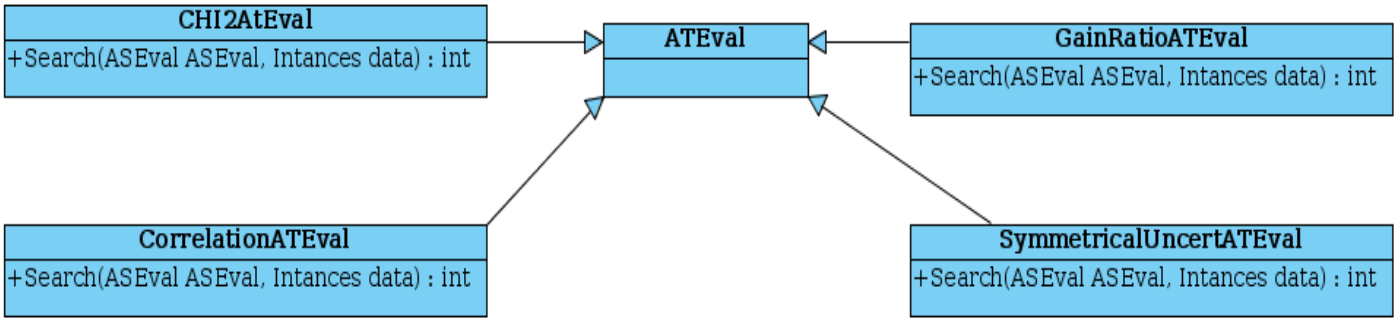


Figura 3.10 Ejemplo de la aplicación del Patrón Polimorfismo

3.3 Descripción de la solución

Para la implementación de la solución se utilizaron algoritmos de búsqueda global, tales como, algoritmos genéticos y de enfriamiento simulado, así como hibridaciones entre un algoritmo de búsqueda global (Algoritmo Genético) y un algoritmo de búsqueda local (búsqueda greedy).

Para realizar la reducción de espacio muestral y ordenamiento de las variables se escoge un método de búsqueda y la medida de evaluación como criterio para la búsqueda. Este método de búsqueda y medida de evaluación están en dependencia del administrador del sistema, pues él escoge con cual de los métodos realizará la selección y que medida o criterio usará para la evaluación. A continuación se muestra el funcionamiento de los métodos de búsqueda más importantes en la solución.

3.3.1 Algoritmos Implementados

3.3.1.1 Algoritmo Genético

Los algoritmos genéticos son procesos de búsqueda globales basados en los principios de selección y evolución natural, así como en el principio Darwiniano de reproducción y supervivencia. Son algoritmos matemáticos de un elevado paralelismo y se basan en factores estocásticos para su ejecución. Para el desarrollo de la solución llegamos a un conjunto de aspectos fundamentales a tener en cuenta:

Diseñar una representación eficiente para optimizar el procedimiento de búsqueda, o sea codificar las posibles soluciones del problema en forma de cadenas binarias, donde cada una de estas

(individuos), constituyen un subconjunto de variables candidato a ser solución. Así, dado un conjunto potencia *U*, tal que $U = \{A, B, C, D, E, F, G\}$ con cardinalidad 7, un subconjunto $U' = \{B, D, G\} \in U$, este se representa por la siguiente cadena binaria $L = [0101001]$ donde el valor '1' representa la presencia de variables, y el valor '0', representa la ausencia de variables en el subconjunto.

Decidir como inicializar una población para garantizar una mayor diversidad de soluciones y el procedimiento no converja prematuramente. En este sentido, se tienen dos variantes de solución, una es inicializar aleatoriamente toda la población, y la otra es sembrar un individuo prefijado en la población inicial, para acelerar el proceso evolutivo.

Diseñar la forma de evaluar un individuo. Se trata de atribuir numéricamente las aptitudes de los cromosomas en una población, asignarle el valor numérico del resultado de la evaluación de la función objetivo. Existen, sin embargo, dos problemas importantes asociados a este método, como son la **competición próxima** (individuos cuya aptitud relativa son próximas numéricamente) y el efecto de **súper individuos** (individuos con evaluación muy superior a la media, capaces de dominar el proceso de selección, haciendo que el AG converja prematuramente hacia un óptimo local). Para resolver estos problemas, se desarrollaron métodos de transformación numérica de evaluación de aptitud de cada subconjunto generado, como son las técnicas de escalado lineal y potencial. Estas garantizan, que un elemento típico de la población contribuya en promedio con un descendiente de la próxima generación, proporcionando a su vez una mayor variedad de esquemas.

Diseñar operadores genéticos que proporcionen un mecanismo estructurado de intercambio de información útil (bloques constructivos para el cruzamiento) entre individuos y a su vez explorar nuevas zonas del espacio de búsqueda y permita escapar de máximos locales.

Decidir cómo seleccionar los individuos (soluciones) para la próxima generación de modo que contribuya geométricamente a la presencia de esquemas aventajados y reducir la presencia de los retrasados. Dada la variedad existente de esquemas de selección, se desarrolla un mecanismo de selección elitista combinado con selección por ruleta, donde las dos mejores soluciones de la

población actual son insertadas en la siguiente generación para mejorar el proceso de convergencia del algoritmo, y los restantes individuos son seleccionados probabilísticamente en proporción a la aptitud que estos posean.

Decidir la condición de parada que puede ser por un número fijado de generaciones o cuando el algoritmo converge a una misma solución, lo cual se explica cuando toda la población posea una misma solución.

A continuación se muestra el seudocódigo del algoritmo implementado:

```
function GeneticSearch(eval)

  t := 0;
  Inicializar P(t) ;
  Evaluar P(t) ;
  Escalar P(t) ;
  Obtener_Mejor_Individuo P(t) ;
  Para t := 1 hasta cantidad Generaciones(max) hacer :
      Seleccionar P(t) desde P(t − 1) ;
      Cruzar P(t) ;
      Mutar P(t) ;
      Evaluar P(t) ;
      Escalar P(t) ;
      converge := Obtener_Mejor_Individuo P(t) ;
      Estadisticas P(t) ;

      Si (i = max) or (converge = true) entonces
        break;
      fin Si
  fin Para
  atributos := Listar(Mejor Individuo);
  return atributos;
fin funcion
```

Donde, *P(t)* es la población en la iteración *t*.

3.3.1.2 Enfriamiento Simulado

Este algoritmo es una metaheurística para problemas de optimización global que se basa en conceptos de la mecánica estadística. La característica principal de este algoritmo es que al buscar una nueva solución Sn+1 dada una solución Sn, acepta en ocasiones una de inferior aptitud a la de Sn por medio de una función probabilística. Para el desarrollo de este algoritmo de búsqueda se tienen en cuenta los siguientes aspectos:

Diseñar una representación simple y eficiente de la solución a través de una cadena binaria, similar al procedimiento de búsqueda genética. De esta forma se discretiza el espacio de búsqueda y contribuye a explorarlo eficientemente.

Otro aspecto es determinar los parámetros que influyen el proceso de enfriamiento, estos son: la Temperatura Final (*Tf*), la Temperatura Inicial (*Ti*), el número de iteraciones por temperatura*(N*), el tipo de coeficiente de enfriamiento, etc., pues estos constituyen un factor primordial para el éxito del algoritmo. Estos parámetros pueden ser cambiados para un fin específico, aunque poseen valores predeterminados en el procedimiento de búsqueda.

Se determina la primera solución aleatoriamente, o sea se genera una cadena binaria que representa el subconjunto de variables solución. En estos casos, la inicialización de la primera solución no constituye un gran peso en la solución para valores altos de temperatura, puesto que es muy probable que esa solución sea rechazada por una de menor calidad.

Decidir cuantas iteraciones hacer por cada temperatura, iterar muchas veces para temperaturas tempranas podría significar un alto coste, puesto que el procedimiento de búsqueda es inestable y no existe un patrón de estados candidatos a ser buenas soluciones.

Generar una solución vecina de la solución anterior, de modo que se explore en gran medida el espacio de búsqueda. Se calcula la diferencia de ajuste entre ambas $Dif = (F_k - F_{k+1})$, donde F_k es la solución anterior y F_{k+1} es la actual, de tal forma que siempre se acepta la nueva solución si se cumple $(Dif < 0)$. Así, si la nueva solución generada tiene menor ajuste $(Dif > 0)$ la probabilidad de

aceptar F_{k+1} como solución para una temperatura (T), esta dada por $N(0,1) < e^{(-Dif/T)}$ donde $N(0,1)$ es un número aleatorio con distribución uniforme entre cero y uno.

Decidir el valor del coeficiente de enfriamiento de temperatura, quien influye en la calidad del procedimiento y en la rapidez de ejecución del mismo. Existe una demostración matemática donde los algoritmos de búsqueda por enfriamiento simulado convergen al óptimo global de un problema, si la temperatura desciende infinitamente lenta [45], por ejemplo, para un valor muy cercano a uno del coeficiente de enfriamiento. Obviamente esto no se puede implementar en una aplicación real en la que el procedimiento termina en valores de tiempo finito.

A continuación se muestra el seudocódigo del algoritmo implementado:

```
function SimmulatedAnnealing (T0, Tf, k, nVecinos)
      T = T0
      Sactual = Genera solución aleatoria;
      Mientras T >= Tf hacer:
          Para i en nVecinos (T) hacer:
              Scandidata = Genera un vecino (Sactual)
              λ = coste (Scandidata) − coste (Sactual)
              Si U (0, 1) < e−λ/T or λ < 0 entonces:
                Sactual = Scandidata;
              fin Si
          fin Para
          Estadisticas();
          T = k*(T);
       fin Mientras
      atributos := Listar(SActual);
      return atributos;;
fin function
```

Donde:
Sactual : solucion actual
Scandidata: solucion candidata
T0 es la temperatura inicial
Tf: la temperatura final
k: es el coeficiente de enfriamiento elegido
nVecinos(T): el número de vecinos generados en cada ciclo según T
U(0,1) :es un generador de números aleatorios uniformemente distribuidos.

3.3.1.2 Hibridación entre Algoritmo Genético y algoritmo de Búsqueda Secuencial

Los algoritmos genéticos son por construcción métodos de búsqueda ciega, el proceso de optimizar es una caja negra que asigna a cada individuo una aptitud. Esta opacidad en la medida que proporciona un algoritmo de propósito general y permite realizar la búsqueda con información mínima, tienen la contrapartida de que son intrínsecamente débiles. Como la debilidad es intrínseca, cualquier intento de mejora cualitativa implica incorporarle al algoritmo un mecanismo de explotación de la solución, después de explorar el espacio de búsqueda.

La idea general de esta técnica de hibridación consiste en utilizar el algoritmo genético para realizar la búsqueda global y encargar la búsqueda local greedy (secuencial) para explotar la solución. Para esto

fue necesario llevar a cabo la hibridación de forma modular, incorporando el procedimiento de búsqueda secuencial como un operador más del algoritmo genético.

El procedimiento de búsqueda local, toma como punto de partida las soluciones brindadas por el algoritmo genético en cada generación después de aplicarles los operadores probabilísticos, así el método de búsqueda secuencial, explota los estados vecinos que generan estas soluciones globales considerando solo aquellas que sean mejores.

A todos estos algoritmos de búsqueda planteados anteriormente se le incorporó un mecanismo de almacenamiento de las mejores soluciones durante su ejecución. O sea, se implementó un proceso de almacenamiento de aquellas soluciones cuya aptitud fuera superior a la aptitud promedio del conjunto de soluciones almacenadas. Para lograr mayor eficiencia en consultas de selección e inserción al conjunto de soluciones, estas se almacenan en Tablas Hash. Este mecanismo de estadística simple, permite obtener un subconjunto de soluciones (ordenadas por aptitud) finales al problema, permitiéndole al usuario escoger cualquiera de estas.

Los métodos de evaluación antes referidos necesitan una medida o criterio de evaluación por lo que fueron implementadas las siguientes mediadas de evaluación:

Para las variables individuales se implementaron las de CHI2, Correlación de Pearson, Incertidumbre Simétrica y la de Gain Ratio.

Para subconjuntos fueron implementadas: las de Correlación de subconjuntos y la de consistencia.

A continuación se muestra un diagrama de flujo (**figura 3.3**) donde se explica de modo general la funcionalidad de reducción de espacio muestral y la de ordenamiento de las características independientes de acuerdo con la relevancia que estas presentan con respecto a la clase o característica dependiente (actividad biológica).

Figura 3.3 Diagrama general del flujo de eventos

De manera general este diagrama muestra es como son cargados al sistema los ficheros de tipo .arff y por los procesos que pasan para ser reducidos u ordenados en dependencia de la orden que el administrador le pase al sistema. Al final del proceso se entrega un fichero en el que se encuentran los datos reducidos, este fichero es de tipo .arff

De manera más específica el siguiente diagrama de flujo (**figura 3.4**) muestra el proceso de la reducción de las características, para tener una visión ampliada del proceso de reducción y lograr un mejor entendimiento de la solución del problema.

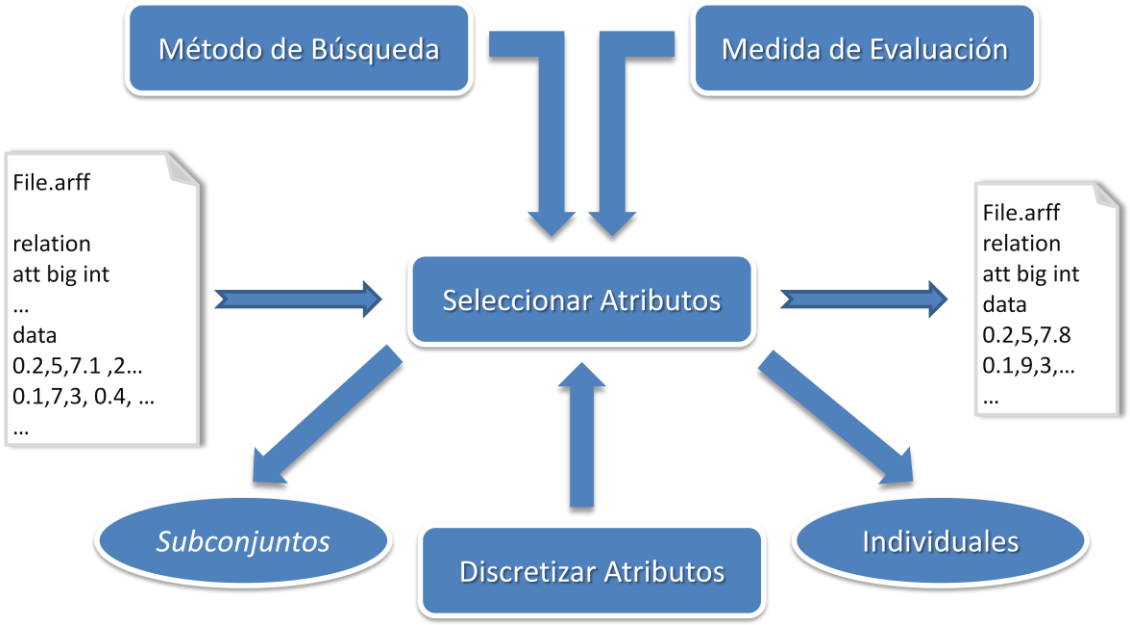


Figura 3.4 Diagrama de flujo para el proceso de selección de atributos o reducción de espacio muestral.

En el diagrama anterior se describe como el fichero de entrada de tipo .arff es incorporado desde una base de datos al sistema, con el objetivo de realizarle una reducción de sus atributos. El administrador selecciona el método de búsqueda y la medida de evaluación, en caso de que este fichero presente datos con valores continuos y la variable dependiente o clase posea valores discretos, entonces, se procede a la discretización de los datos, este proceso de discretización no es más que convertir los valores continuos a valores de tipo discreto, buscando así uniformidad entre los datos de las variables independientes y la variable dependiente (clase).

Las medidas de evaluación son utilizadas para selección de características de tipo individual o de subconjuntos, de acuerdo a la que se haya seleccionado se realiza la reducción correspondiente.

Al terminarse la selección los nuevos datos son guardados en un fichero nuevo de tipo .arff y este es guardado en una base de datos.

Implícito dentro del proceso de selección se encuentra el proceso de ordenamiento de los atributos, el cual se explica de la siguiente manera y basándose en el diagrama anterior:

El administrador selecciona el método de búsqueda y la medida de evaluación, en caso de que este fichero presente datos con valores continuos y la variable dependiente o clase posea valores discretos, entonces, se procede a la discretización de los datos.

Al finalizar este proceso de ordenamiento en vez de crearse un fichero .arff se crea un fichero .txt con los datos ordenados de acuerdo con la relevancia que estos presentan con la variable dependiente o clase. Este proceso es muy importante pues con sus resultados se pueden realizar estudios estadísticos por parte de los especialistas en la parte de predicción y clasificación para de alguna manera tener una visión de la relación que tienen los atributos o variables independientes con respecto a la variable dependiente o clase.

3.5 Análisis de los resultados

Para comprobar la eficiencia y rapidez de los métodos implementados se tomaron muestras de datos reales del compuesto cefalosporina [46]. A continuación se muestran las características del compuesto:

Cefalosporina
11 atributos reales
1 clase (0,1)
105 instancias

En la **tabla 3.1** se muestran los resultados. Contenidos en 5 columnas se encuentran, el identificador de la reducción, el cual será utilizado como referencia en las pruebas de clasificación, el método de búsqueda empleado, así como las medidas de evaluación utilizadas por los métodos de búsqueda, los parámetros iniciales de los algoritmos, en dependencia de la medida con la que se realiza la evaluación se presenta si posee predicción local o no y el % de la reducción de la muestra.

Tabla 3.1 Reducción del compuesto Cefalosporina mediante los algoritmos implementados en el módulo.

Identificador de la Reducción	Método de búsqueda	Medida de evaluación	Parámetros de búsqueda	Predicción local de atributos	% de la reducción de la muestra
AG-Consistencia	**Algoritmo genético**	**Consistencia**	n:10	no posee	63.7
			m:100		
			Pc:0,75		
			Pm:0,15		
AG-CFS-pl	**Algoritmo genético**	**Correlación**	n:10	Si	63.7
			m:100		
AG- CFS-npl			Pc:0,75	No	63.7
			Pm:0,15		
ES-Consistencia	**Enfriamiento simulado**	**Consistencia**	Ti:50	no posee	73.2
			it:100		
			Tf:0,00001		
			a:0,99		
ES- CFS-pl	**Enfriamiento simulado**	**Correlación**	Ti:50	Si	63.7
			it:100		
ES- CFS-npl			Tf:0,00001	No	63.7
			a:0,99		
BH-Consistencia	**Búsqueda híbrida**	**Consistencia**	n:10	no posee	73.2
			m:100		
			Pc:0,75		
			Pm:0,15		
BH- CFS-pl	**Búsqueda híbrida**	**Correlación**	n:10	Si	45.5
			m:100		
BH- CFS-npl			Pc:0,75	No	63.7
			Pm:0,15		

n: tamaño de la población. **m: # de generaciones.** **Pc: Probabilidad de cruzamiento.**
Pm: Probabilidad de mutación. **Ti: temperatura inicial.** **it: iteraciones por temperatura.**
Tf: temperatura final. **a: factor de decremento.**

En el **anexo # 1** se muestra de manera gráfica las mejores reducciones hechas con los distintos algoritmos.

Después de haber realizado la reducción de las muestras con los distintos métodos implementados en el módulo se procede a realizar la comparación de los resultados obtenidos. Las muestras son tomadas y evaluadas a través de las distintas técnicas de clasificación, la evaluación se realiza primeramente en una Red Bayesiana, donde en la **tabla 3.2** se muestran los resultados obtenidos.

Los valores aparecen definidos, según el método de búsqueda y la medida de evaluación para una selección de atributos dada. Así, la primera columna muestra el identificador de selección, la segunda y tercera muestran respectivamente, el porciento de instancias clasificadas correcta e incorrectamente, la cuarta columna la calidad del modelo generado por el clasificador, en términos del ajuste para cada posible valor del atributo clase(en este caso binaria).

Tabla 3.2 Clasificación utilizando una Red Bayesiana.

Identificador de reducción	% Clasificados Correctamente	% Clasificados Incorrectamente	Calidad del modelo	
			Clase 0	Clase 1
Clasificación sin selección de variables previa				
	75.0000	25.0000	0.740	0.759
Búsqueda utilizando Algoritmos Genéticos				
AG-Consistencia	76.9231	23.0769	0.769	0.769
AG-CFS-pl	77.8846	22.1154	0.777	0.781
AG-CFS-npl	76.9231	23.0769	0.769	0.769
Búsqueda utilizando Enfriamiento Simulado				
ES- Consistencia	77.8846	22.1154	0.777	0.781
ES-CFS-pl	77.8846	22.1154	0.777	0.781
ES-CFS-npl	77.8846	22.1154	0.777	0.781
Búsqueda utilizando algoritmo Híbrido				
BH- Consistencia	75.9615	24.0385	0.757	0.762
BH-CFS-pl.	83.6538	16.3462	0.835	0.838
BH-CFS-npl.	77.8846	22.1154	0.777	0.781

En el **anexo #2** se representa de manera gráfica los mejores valores obtenidos por los métodos implementados en la **tabla 3.2**.

Además de evaluar los resultados con este clasificador bayesiano, se evalúan también mediante una Red Neuronal con aprendizaje back-propagation, en cuya capa oculta se definen tres neuronas, un coeficiente de aprendizaje de 0.3 y un momento de inercia de 0.2, mostrándose los resultados en la **tabla 3.3**. Es importante destacar, que a estos clasificadores se les aplica validación cruzada para evitar el sobre-aprendizaje de los mismos, utilizando 10 particiones del conjunto original de datos.

Tabla 3.3 Clasificación Utilizando una Red Neuronal.

Identificador de reducción	**% Clasificados Correctamente**	**% Clasificados Incorrectamente**	**Calidad del modelo**	
			Clase 0	**Clase 1**
Clasificación sin selección de variables previa				
	50.9615	49.0385	0.592	0.386
Búsqueda utilizando Algoritmos Genéticos				
AG-Consistencia	83.6538	16.3462	0.832	0.841
AG-CFS-pl	82.6923	17.3077	0.830	0.824
AG-CFS-npl	74.0385	25.9615	0.722	0.757
Búsqueda utilizando Enfriamiento Simulado				
ES- Consistencia	78.8462	21.1538	0.766	0.807
ES-CFS-pl	82.6923	17.3077	0.830	0.824
ES-CFS-npl	82.6923	17.3077	0.830	0.824
Búsqueda utilizando algoritmo Híbrido				
BH- Consistencia	81.7308	18.2692	0.804	0.829
BH-CFS-pl.	82.6923	17.3077	0.830	0.824
BH-CFS-npl.	83.6538	16.3462	0.832	0.841

En el **anexo #3** se representa de manera gráfica los mejores valores obtenidos por los métodos implementados en la **tabla 3.3.**

Conclusiones Generales

Nuestra investigación fue llevada a cabo desde octubre de 2007 hasta junio de 2008 y al culminar la misma arribamos a las siguientes conclusiones:

- ✓ Se realizó una búsqueda en la bibliografía, referente al tema de la selección de variables, identificándose los algoritmos utilizados a nivel mundial para la resolución de este tipo de problema. Escogiéndose los algoritmos genéticos y de enfriamiento simulado.
- ✓ Se implementaron procedimientos de búsqueda que utilizan algoritmo genético y enfriamiento simulado para la reducción de la dimensionalidad de los datos, así como, las medidas de evaluación basadas en la correlación, consistencia, y Teoría de la Información de Shannon para garantizar la calidad del procedimiento de búsqueda.
- ✓ Se implementó un algoritmo híbrido que combina algoritmo genético con algoritmos de búsqueda secuencial, lo que constituye algo novedoso pues en la bibliografía consultada no se encontró ningún trabajo o publicación de este tipo específico de hibridación.
- ✓ Se evaluaron los algoritmos implementados con los cuales se obtuvieron resultados satisfactorios, pues al reducirse las muestras utilizando algoritmo genético se obtuvo una reducción del 62,7% de la muestra original, al aplicársele enfriamiento simulado se obtuvo una reducción del 73.2 % de la muestra original y al aplicarle el algoritmo híbrido se realizó una reducción del espacio muestral del 73,2% de la muestra original. Se comprobó además la calidad de la solución obtenida en la reducción al aplicarse clasificadores profesionales, provenientes del software Weka. Los resultados obtenidos fueron satisfactorios.

Recomendaciones

Como colofón a esta investigación, recomendamos continuar trabajando en:

- ✓ Crear un módulo con los algoritmos y medidas implementados para que estos sean incluido en la plataforma.
- ✓ Buscar nuevos métodos y medidas de evaluación que contribuyan al mejoramiento de los resultados obtenidos con los algoritmos implementados.

1. Kohavi, R. “Wrappers for Performance Enhancement and Oblivious Decision Graphs”. 1995 Stanford University, Computer Science Department.

2. [Consultado el 1 de noviembre de 2007] Disponible en: http://es.wikipedia.org/wiki/Inteligencia_artificial.

3. [Consultado el 1 de noviembre de 2007] Disponible en: http://www.eumed.net/libros/2006a/rmss/a5.htm

4. Garey, M. R.; Johnson, D. S., "Computers and Intractability: a Guide to the Theory of NP-Completeness", W. H. Freeman and Co., San Francisco, California, 1979. [Consultado el 8 de noviembre de 2007]

5. Kohavi, R.; George H. J., "Wrappers for feature subset selection. Artificial Intelligence", 1997 [Consultado el 8 de noviembre de 2007]

6. Langley, P., "Selection of relevant features in machine learning." In Procedings of the AAAI Fall Symposium on Relevance, páginas 1–5, New Orleans, LA, USA, 1994. AAAI Press. [Consultado el 8 de noviembre de 2007]

7. Dash, M.; Liu, H., "Feature selection for classification. Intelligent Data Analysis", 1(1-4):131–156, 1997.[Consultado el 20 de noviembre de 2007] [Consultado el 20 de noviembre de 2007]

8. Kohavi, R., "Feature subset selection as search with probabilistic estimates." In AAAI Fall Symposium on Relevance, paginas 122–126, November 1994.[Consultado el 23 de noviembre de 2007]

9. S. Davies and S. Russell. Np-completeness of searches for smallest possible feature sets. In AAAI Press, editor, Proceedings of the 1994 AAAI Fall Symposium on Relevance, paginas 37–39, 1994.[Consultado en enero de 2008]

10. Brassard, G.; Bratley,T., "Fundamentos de Algoritmia." Prentice Hall, 1997.[Consultado el 9 de enero de 2008]

11. Liu, H.; Setiono, R., "A probabilistic approach to feature selection - a filter solution". In International Conference on Machine Learning, páginas 319–327, 1996. [Consultado el 9 de enero de 2008]

12. Liu, H.; Setiono, R., "Feature selection and classification - aprobabilistic wrapper approach."[Consultado el 9 de enero de 2008]

13. [Consultado el 1 de noviembre de 2007] Disponible en: http://es.wikipedia.org/wiki/Heur%C3%ADstica.

14 [Consultado el 1 de noviembre de 2007] Disponible en: http://enciclopedia.us.es/index.php/Heur%C3%ADstica

15. Moreno Pérez, J A.; Melián Batista, B "METAHEURÍSTICAS PARA LA PLANIFICACIÓN LOGÍSTICA", Grupo de Computación Inteligente. Universidad de La Laguna, 2005 [Consultado el 3 de noviembre de 2007].

16. [Consultado el 1 de noviembre de 2007] Disponible en:http://es.wikipedia.org/wiki/Metaheur%C3%ADsticas

17. Glover, F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence, Computers and Operations Research" 13, 533-549.[Consultado el 3 de noviembre de 2007].

18. Holland, J. H., "Adaptation in Natural And Artificial Systems", University of Michigan Press, Ann Arbor, 1975.[Consultado el 10 de enero de 2008]

19. Cerny, V., "Thermodynamical Approach to the Traveling Salesman Problem: An efficient Simulated Algorithm", Journal of Optimization Theory and Applications, Vol. 45, 41-45, 1985.[Consultado el 10 de enero de 2008]

20. Kirpatrick, S., Gelatt, C. and Vecchi, M. "Optimization by Simulated Annealing, Science", vol. 220, pp. 672-680, 1983.[Consultado el 13 de enero de 2008]

21. Glover, F., "Tabu Search: A Tutorial", Interfaces, Vol 20, No. 4, pp. 74-94, 1990. [Consultado el 10 de enero de 2008]

22. Durán Acevedo, C. M., "Diseño y optimización de los subsistemas de un sistema de olfato electrónico para aplicaciones agroalimentarias e industriales." pág 40.(2000) [Consultado el 14 de noviembre de 2007]

23. [Consultado el 15 de noviembre de 2007] Disponible en: http://es.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico.

24.Coello Coello, Carlos A., "Introducción a los Algoritmos Genéticos" [Consultado el 14 de noviembre de 2007] Disponible en: http://www.redcientifica.com/doc/doc199904260011.html

25.Quinlan, J. R., "Induction of decision trees. Machine Learning", 1:81–106, 1986.[Consultado el 10 de enero de 2008]

26. METRÓPOLIS, N; ROSENBLUTH, A.; TELLER, A.; TELLER, E. "Equations of state calculations by fast computing machines". The journal of chemical physics, Vol. 21, No. 6.1953.[Consultado el 14 de noviembre de 2007].

27.KIRKPATRICK, S.; GELATT, C. D. and VECCHI, M. P. "Optimization by Simulated Annealing". Science 220, pp. 671-680. 1983.[Consultado el 14 de noviembre de 2007].

28. GUTIÉRREZ, M. Á.; DE LOS COBOS, S. G.; PÉREZ Salvador, B. R. "Optimización con recocido simulado para el problema de conjunto independiente". Revista En Línea. Universidad Autónoma Metropolitana. México, 1998. [Consultado el 12 de febrero de 2008] Disponible en: http://www.azc.uam.mx/publicaciones/enlinea2/3-2.html.

29. [Consultado el 12 de febrero de 2008] Disponible: http://www.cimat.mx/~horebeek/cursus/node40.html.

30. Quinlan, J. R., "Programs for Machine Learning." C 4.5 Morgan Kaufmann, 1993.[Consultado el 12 de enero de 2008]

31. [Consultado el 12 de enero de 2008] Disp en:http://www.gsi.dit.upm.es/~anto/tesis/html/evalits.html

32. Gini coefficient. Disponible en : http://en.wikipedia.org/wiki/Gini index, Feb 2006. [Consultado el 17 de febrero de 2008]

33. Breiman, L., editor. "Classification and regression trees." Chapman & Hall, 1998. [Consultado el 18 de febrero de 2008] Coello Coello, Carlos A.,

34. Kira, K; Rendell, L. A., "A practical approach to feature selection." In Proceedings of the Ninth International Conference on Machine Learning, páginas 249–256, Aberdeen, Scotland, 1992. Morgan Kaufmann. [Consultado el 18 de febrero de 2008]

35. Kononenko, I., "Estimating attributes: Analysis and extensions of RELIEF." In European Conference on Machine Learning, páginas 171–182, 1994.[Consultado el 22 de febrero de 2008]

36. Schlimmer, J.C., "Efficiently inducing determinations: A complete and efficient search algorithm that uses optimal pruning." In Proceedings of the Tenth International Conference on Machine Learning, pages 284–290, New Brunswick, NJ, 1993. Morgan Kaufmann. [Consultado el 12 de marzo de 2008]

37. Almuallim, H.; Dietterich, G.T., "Learning with many irrelevant features." In Proceedings of the Ninth National Conference on Artificial Intelligence, volume 2, pages 547–552, San Jose,CA, 1991.AAAI Press.[Consultado el 12 de diciembre de 2007]

38. Huan Liu, Hiroshi Motoda, and Manoranjan Dash. A monotonic measure for optimal feature selection. In European Conference on Machine Learning, paginas 101–106, 1998. [Consultado el 14 de marzo de 2008]

39. Pawlak, Z. "Rough Sets, Theoretical aspects of reasoning about data." Kluwer Academic Publishers, 1991. [Consultado el 14 de marzo de 2008]

40. Cover, T. M.; Thomas, Joy A. "Elements of Information theory." Wiley-Interscience, 1991. [Consultado el 14 de marzo de 2008]

41 Press, W.H.; Flannery, B.P.; Teukolsky, S.A.; Vetterling, W.T. ".Numerical recipes in C. Cambridge University Press", 1998.[Consultado el 14 de marzo de 2008]

42. Sheinvald, J.; Dom, B.; Niblack, W. "A modelling approach to feature selection." In 10th International Conference on Pattern Recognition, volume i, pages 535–539, 1990. [Consultado el 14 de marzo de 2008]

43. Patrenahalli, M.; Fukunaga, N.; Fukunaga, K. "A branch and bound algorithm for feature subset selection." IEEE Transactions on Computers, 26(9):917–922, sep 1977. [Consultado el 14 de marzo de 2008]

44. **Rosales García, A. R.; Marrero López, Y.** *Propuesta del diseño arquitectónico de la plataforma GRAph-TOol .*

45. Deekers, A.; Aarts, E., "Global Optimization and Simulated Annealing, Mathematical Programming" 50, 1991, pp. 367-393. [Consultado el 12 de febrero de 2008]

46. **Carrasco Velar, R.** "*Nuevos descriptores atómicos y moléculares para estudios de estructura-actividad: Aplicaciones."* Ciudad de La Habana: s.n., 2003.

Bibliografía

Introducción al diseño de fármacos/ Julio César Escalona Arranz, Ramón Carrasco Velar y Juan Alexander Padrón García. -- Ciudad de La Habana: Editorial Universitaria, 2008. ISBN 978-959-16-0647-1.

Garey, M. R.; Johnson, D. S., "Computers and Intractability : a Guide to the Theory of NP-Completeness", W. H. Freeman and Co., San Francisco, California, 1979. [Consultado el 8 de noviembre de 2007]

Kohavi, R.; George H. J., "Wrappers for feature subset selection. Artificial Intelligence", 1997 [Consultado el 8 de noviembre de 2007]

Langley, P., "Selection of relevant features in machine learning." In Procedings of the AAAI Fall Symposium on Relevance, páginas 1–5, New Orleans, LA, USA, 1994. AAAI Press. [Consultado el 8 de noviembre de 2007]

Dash, M.; Liu, H., "Feature selection for classification. Intelligent Data Analysis", 1(1-4):131–156, 1997.[Consultado el 20 de noviembre de 2007] [Consultado el 20 de noviembre de 2007]

Kohavi, R., "Feature subset selection as search with probabilistic estimates." In AAAI Fall Symposium on Relevance, paginas 122–126, November 1994.[Consultado el 23 de noviembre de 2007]

S. Davies and S. Russell. Np-completeness of searches for smallest possible feature sets. In AAAI Press, editor, Proceedings of the 1994 AAAI Fall Symposium on Relevance, paginas 37–39, 1994.[Consultado en enero de 2008]

Brassard, G.; Bratley,T., "Fundamentos de Algoritmia." Prentice Hall, 1997.[Consultado el 9 de enero de 2008]

Liu, H.; Setiono, R. , "A probabilistic approach to feature selection - a filter solution". In International Conference on Machine Learning, paginas 319–327, 1996. [Consultado el 9 de enero de 2008]

Liu, H.; Setiono, R., "Feature selection and classification - aprobabilistic wrapper approach."[Consultado el 9 de enero de 2008]

Durán Acevedo, C. M., "Diseño y optimización de los subsistemas de un sistema de olfato electrónico para aplicaciones agroalimentarias e industriales." pág 40.(2000) [Consultado el 14 de noviembre de 2007]

[Consultado el 15 de noviembre de 2007] Disponible en: http://es.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico.

Coello Coello, Carlos A., "Introducción a los Algoritmos Genéticos" [Consultado el 14 de noviembre de 2007] Disponible en: http://www.redcientifica.com/doc/doc199904260011.html

Quinlan, J. R., "Induction of decision trees. Machine Learning", 1:81–106, 1986.[Consultado el 10 de enero de 2008]

METRÓPOLIS, N; ROSENBLUTH, A.;TELLER, A.; TELLER, E. “Equations of sta69te calculations by fast computing machines”. The journal of chemical physics, Vol. 21, No. 6.1953.[Consultado el 14 de noviembre de 2007].

GUTIÉRREZ, M. Á.; DE LOS COBOS, S. G.; PÉREZ Salvador, B. R. “Optimización con recocido simulado para el problema de conjunto independiente”. Revista En Línea. Universidad Autónoma Metropolitana. México, 1998. [Consultado el 12 de febrero de 2008] Disponible en: http://www.azc.uam.mx/publicaciones/enlinea2/3-2.html.

Breiman, L., "Classification and regression trees." Chapman & Hall, 1998. [Consultado el 18 de febrero de 2008] Coello Coello, Carlos A.,

Kira, K; Rendell, L. A., "A practical approach to feature selection." In Proceedings of the Ninth International Conference on Machine Learning, páginas 249–256, Aberdeen, Scotland, 1992. Morgan Kaufmann. [Consultado el 18 de febrero de 2008]

Kononenko, I., "Estimating attributes: Analysis and extensions of RELIEF." In European Conference on Machine Learning, páginas 171–182, 1994.[Consultado el 22 de febrero de 2008]

Schlimmer, J.C., "Efficiently inducing determinations: A complete and efficient search algorithm that uses optimal pruning." In Proceedings of the Tenth International Conference on Machine Learning, pages 284–290, New Brunswick, NJ, 1993. Morgan Kaufmann. [Consultado el 12 de marzo de 2008]

Almuallim, H.; Dietterich, G.T., "Learning with many irrelevant features." In Proceedings of the Ninth National Conference on Artificial Intelligence, volume 2, pages 547–552, San Jose,CA, 1991.AAAI Press.[Consultado el 12 de diciembre de 2007]

Liu, H; Motoda, H; Dash, M . “A monotonic measure for optimal feature selection.” In European Conference on Machine Learning, paginas 101–106, 1998. [Consultado el 14 de marzo de 2008]

Pawlak, Z. "Rough Sets, Theoretical aspects of reasoning about data." Kluwer Academic Publishers, 1991. [Consultado el 14 de marzo de 2008]

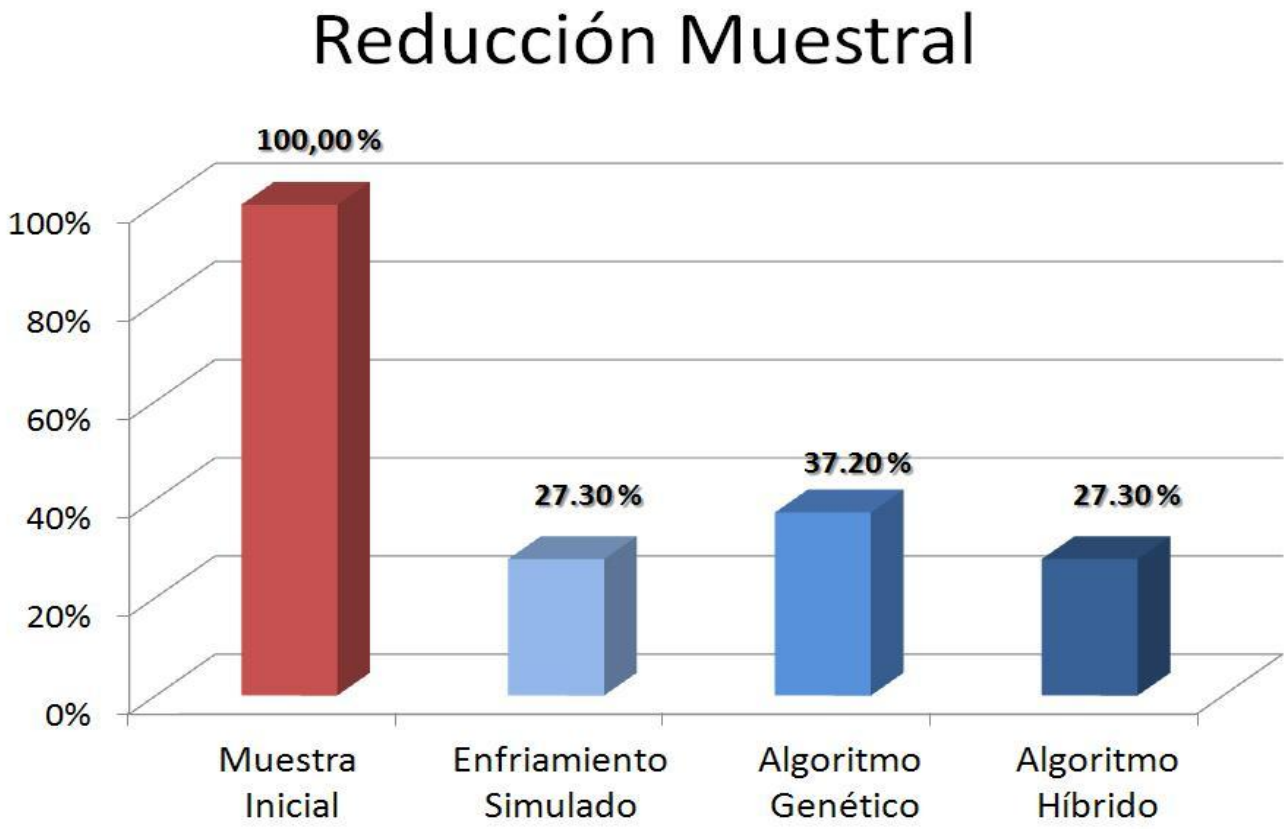
Cover, T. M.; Thomas, J. A. "Elements of Information theory." Wiley-Interscience, 1991. [Consultado el 14 de marzo de 2008]

Sheinvald, J.; Dom, B.; Niblack, W. "A modelling approach to feature selection." In 10th International Conference on Pattern Recognition, volume i, pages 535–539, 1990. [Consultado el 14 de marzo de 2008]

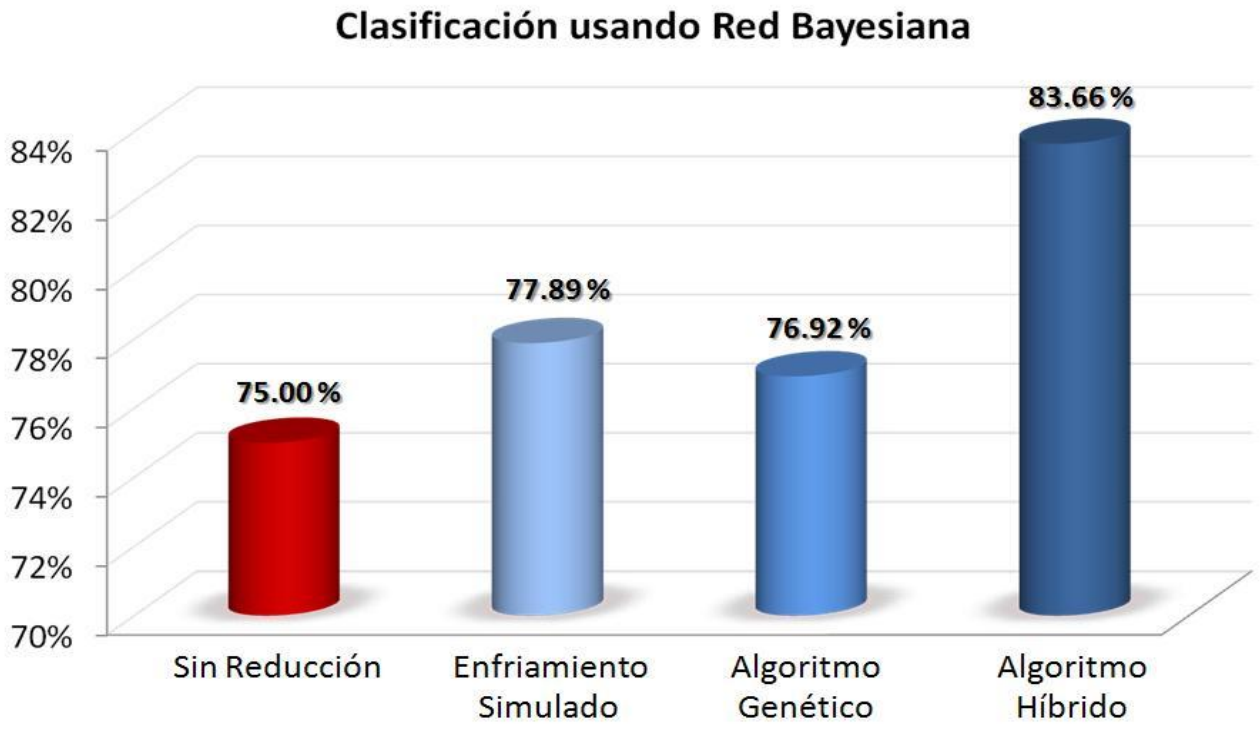
Glover, F. “Future Paths for Integer Programming and Links to Artificial Intelligence, Computers and Operations Research” 13, 533-549. 1986. [Consultado el 3 de noviembre de 2007].

Craiman, L., “UML y Patrones Introducción al análisis y al diseño orientado a objetos” Prentice hall, 1999 [Consultado el 2 de junio de 2007].

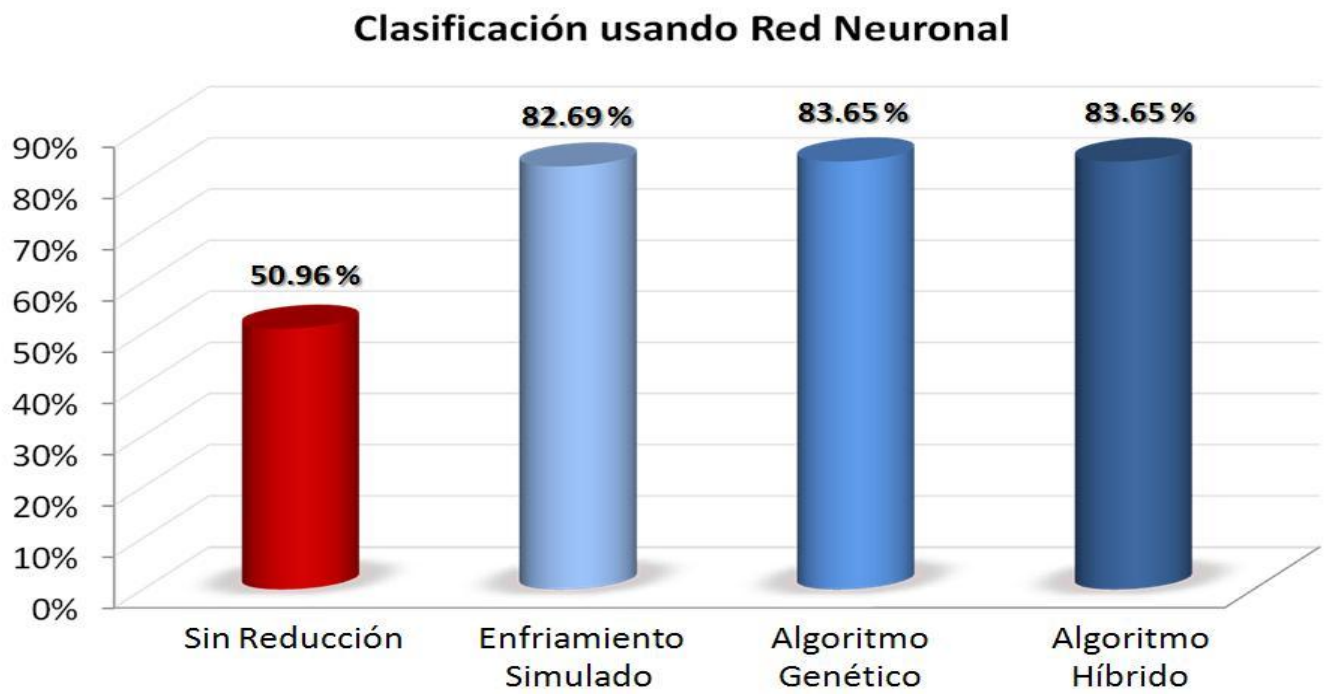
Anexo #1


Reducción Muestral
100,00 %
27.30 %
37.20 %
27.30 %
100%
80%
60%
40%
20%
0%
Muestra Inicial
Enfriamiento Simulado
Algoritmo Genético
Algoritmo Híbrido

Anexo #2


Clasificación usando Red Bayesiana
84%
82%
80%
78%
76%
74%
72%
70%
75.00%
77.89%
76.92%
83.66%
Sin Reducción
Enfriamiento Simulado
Algoritmo Genético
Algoritmo Híbrido

Anexo #3


Clasificación usando Red Neuronal
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
50.96 %
82.69 %
83.65 %
83.65 %
Sin Reducción
Enfriamiento Simulado
Algoritmo Genético
Algoritmo Híbrido

Anexo #4 Código fuente de la búsqueda utilizando algoritmo Genético.

```
public int[] search(ASEval ASEval, Instances data) throws Exception {

    m_best = null;
    int[] attributes;
    if (!(ASEval instanceof SubsetEval)) {
        throw new Exception(ASEval.getClass().getName() + " no es "
                + "un evaluador de subconjuntos!");
    }


    if (data.classIndex() == -1 || !this.m_hasClass || this.m_numAttribs == 0)
    {
        // this.buildSearch(data);
        throw new Exception("Debe inicilizar el metodo de busqueda");
    }

    //CfsSubsetEval ASEvaluator = (CfsSubsetEval) ASEval;

    SubsetEval ASEvaluator = (SubsetEval)ASEval ;

    // ASEvaluator.buildEvaluator(data);

    m_random = new Random(m_seed);
    m_population = new GABitSet[m_popSize];

    // set up random initial population
    initPopulation();
    evaluatePopulation(ASEvaluator);
    populationStatistics();
    scalePopulation(m_scaled);
    checkBest();

    boolean converged;
    for (int i = 1; i <= m_maxGenerations; i++) {
        generation();
        evaluatePopulation(ASEvaluator);

        populationStatistics();
        scalePopulation(m_scaled);
        // find the best pop member and check for convergence
        converged = checkBest();

        // despues que haya caminado el algoritmo
        {
            attributes = attributeList(m_best.getChromosome());
            m_statistic.statisticalSet(attributes, m_best.m_objective);
        }

        if ((i == m_maxGenerations) || (converged == true)) {
            if (converged == true) {
                break;
            }
        }
    }
    // cequear k mi tabla hash no tenga mi solucion final y evitar
    // duplicados
    attributes = attributeList(m_best.getChromosome());
    m_statistic.removeInHash(attributes);
    return attributes;
}
```

Anexo #5 Código fuente de la búsqueda utilizando Enfriamiento Simulado.

```
public int[] search(ASEval ASEval, Instances data) throws Exception {

    int attributes[];
    boolean limit;
    if (m_numAttribs > m_iterates)
        m_iterates = m_numAttribs;

    SubsetEval ASEvaluator = (SubsetEval) ASEval;

    if (data.classIndex() == -1 || !this.m_hasClass || this.m_numAttribs == 0){
        // this.buildSearch(data);
        throw new Exception("Debe inicilizar el metodo de busqueda");
    }

    double new_merit, energy = 0, prob = 0.0, prob_normal = 0.0;
    BitSet new_sol;

    while (m_To >= m_Tf) {
        limit = false;
        for (int i = 0; i <m_iterates ; i++) {
            new_sol = generateSubset();
            new_merit = ASEvaluator.evaluateSubset(new_sol);
            energy = m_fitness - new_merit;

            if (energy < 0) {
                m_best = new_sol;
                m_fitness = new_merit;
                limit = true;
            } else {
                if (Utils.eq(m_fitness, new_merit)) {
                    // busco el de menor attributos
                    int count_old = countFeatures(m_best);
                    int count_new = countFeatures(new_sol);
                    if (count_old > count_new) {
                        m_best = new_sol;
                        m_fitness = new_merit;
                        limit = true;
                    }
                } else {
                    /* probabilidad con media cero varianza 1 */
                    double den = Math.sqrt(3 * Math.PI);
                    double exp = -0.5 * Math.pow(m_gauss.nextDouble(), 2);
                    double num = Math.exp(exp);
                    prob_normal = (double) num / den;
                    prob = Math.exp(-1 * energy / m_To);

                    if (prob_normal < prob) {
                        m_best = new_sol;
                        m_fitness = new_merit;
                    }
                }
            }
            //comenzar estadisticas cuando se posea una buena solucion
            if(m_statistic.count != 0)
            {
                attributes = attributeList(m_best);
                m_statistic.statisticalSet(attributes, m_fitness);
            }

            if (limit) {
                //i = m_iterates;
                // garantizar la primera solucion buena, disminuir poblacion
                if(m_statistic.count==0 && m_To*m_To*m_To*m_To < m_Tf)
                {
                    attributes = attributeList(m_best);
                    m_statistic.statisticalSet(attributes, m_fitness);
                }
            }
        }
        m_To = m_alpha * m_To;
    }
    attributes = attributeList(m_best);
    m_statistic.removeInHash(attributes);
    return attributes;
}
```

Anexo #6 Código fuente de la búsqueda utilizando el Algoritmo Híbrido.

```
public int[] search(ASEval ASEval, Instances data) throws Exception {

    m_best = null;
    int[] attributes;
    m_random = new Random(m_seed);
    m_population = new GABitSet[m_popSize];
    SubsetEval ASEvaluator = (SubsetEval) ASEval;

    if (data.classIndex() == -1 || !this.m_hasClass || this.m_numAttribs == 0){
        throw new Exception("Debe inicilizar el metodo de busqueda antes de efectuarlo.");
    }


    // set up random initial population
    initPopulation();
    evaluatePopulation(ASEvaluator);

    populationStatistics();
    scalePopulation(m_scaled);
    checkBest();

    boolean converged;
    for (int i = 1; i <= m_maxGenerations; i++) {
        generation();
        localSearch(ASEvaluator, data);
        evaluatePopulation(ASEvaluator);
        populationStatistics();
        scalePopulation(m_scaled);
        // find the best pop member and check for convergence
        converged = checkBest();

        // despues que haya caminado el algoritmo
        {
            attributes = attributeList(m_best.getChromosome());
            m_statistic.statisticalSet(attributes, m_best.getObjective());
        }

        if ((i == m_maxGenerations) || (converged == true)) {
            if (converged == true) {
                break;
            }
        }
    }
    // cequear k mi tabla hash no tenga mi solucion final y evitar
    // duplicados
    attributes = attributeList(m_best.getChromosome());
    m_statistic.removeInHash(attributes);
    return attributes;
}
```

Glosario de Términos

A

Actividad biológica: Actividad que caracteriza el comportamiento biológico en compuestos químicos (Molécula o Fragmento).

B

Bioinformática: Es la aplicación de los ordenadores y los métodos informáticos en el análisis de datos experimentales y simulación de los sistemas biológicos.

C

Compuestos Orgánicos: Compuestos cuya composición fundamental es sobre la base del elemento químico carbono.

CQF: Centro de Química Farmacéutica (CQF) es una institución para el desarrollo de investigaciones científico-tecnológicas dirigidas hacia la obtención de sustancias bioactivas para uso humano.

D

Descriptor: Número que caracteriza estructuralmente la molécula.

E

Espacio muestral: Conjunto de todos los posibles resultados individuales de un experimento aleatorio.

G

GPL: Acrónimo de General Public Licence (Licencia pública general de GNU).

N

NP-completo: En teoría de la complejidad computacional, la clase de complejidad NP-completo es el subconjunto de los problemas de decisión en NP tal que todo problema en NP se puede reducir en cada uno de los problemas de NP-completo. Se puede decir que los problemas de NP-completo son los problemas más difíciles de NP(no polinomial) y muy probablemente no formen parte de la clase de complejidad

NP-duro: En teoría de la complejidad computacional, la clase de complejidad NP-hard (o NP-complejo, o NP-difícil) es el conjunto de los problemas solución que contiene los problemas H tales que todo

problema L en NP puede ser transformado polinomialmente *en H.*

P

Polimorfismo: Capacidad que tienen los objetos de una clase de responder al mismo mensaje o evento en función de los parámetros utilizados durante su invocación. Un objeto polimórfico es una entidad que puede contener valores de diferentes tipos durante la ejecución del programa.